BACHELOR'S DEGREE THESIS

Double Degree in Statistics and Economics

Title: Bayesian Neural Networks as a pricing model to reduce information costs in peer-to-peer online marketplaces

Author: Marc Susagna Holgado

Advisors:

- **Xavier Puig Oriol**
- **Salvador Torra Porras**

Departments:

- **Statistics and Operations Research (UPC)**
- **Econometrics, Statistics and Spanish Economy (UB)**

Academic year: 2017/2018

UNIVERSITAT DE BARCELONA

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

SUMMARY III
AMS CLASSIFICATION V
TABLE OF ACRONYMS AND ABBREVIATIONS V
INTRODUCTION VI
METHODOLOGY VIII

CHAPTER I: PRICING MODEL FOR PEER-TO-PEER ONLINE MARKETPLACES 1

1. Theoretical foundations about the benefits of improving availability of information 1
1.1 Overview of the relevance of information in market analysis 1
1.2 Market performance under incomplete information 2
1.3 A specific case: Uncertainty during the ascertainment of the market price 5

2. The information and communications technology revolution and its current reduction of uncertainty 6
2.1 Overview of ICT's positive effects on economies and how they relate to information availability 6
2.2 From the data deluge to the useful information: The rise of Data Science 8

3. Peer-to-peer online marketplaces and their intrinsic potential to reduce uncertainty when ascertaining the market price 10
3.1 Overview of peer-to-peer online marketplaces platforms 10
3.1.1 Definition, relationship with sharing economy and examples 10
3.1.2 Origins, evolution and impact 14
3.1.3 Controversy generated around these platforms and efficient regulation 21
3.2 Explaining their potential as a more efficient framework for transactions 24

4. Pricing model as a tool to ascertain the market price 26
4.1 Rationale of a pricing model in the peer-to-peer online marketplace setting 26
4.2 Implementation as a price recommender: Requirements and considerations 29
4.3 Desirability of a pricing model to ascertain the market price: Choosing a model 32

CHAPTER II: FEED-FORWARD BAYESIAN NEURAL NETWORKS 35

5. Foundations of Bayesian Neural Networks 35
5.1 Overview of the Bayesian framework and its flexibility 35
5.1.1 Theoretical approach of the Bayesian framework 35
5.1.2 Implementation of the Bayesian approach 40
5.2 Rationale of feed-forward Artificial Neural Networks 41
5.2.1 Origins of feed-forward Artificial Neural Networks 42
5.2.2 Explaining their rationale and how they parameters are obtained 42
5.2.3 Settings in which Artificial Neural Networks present a comparative advantage 50
5.3 Motivation to use a Bayesian Neural Network 54
5.4 Bayesian Neural Networks history and literature 55

6. The Bayesian Neural Network proposed in this project **61**
6.1 Model Formulation: Bayesian Neural Network as the bridge between the statistical model and Neural Networks 61
6.2 Fitting the Bayesian Neural Network proposed in this project 64
6.2.1 Consideration about the shape of the posterior distribution 65
6.2.2 Sampling from a complicated posterior distribution with MCMC methods 70
6.2.3 Using Design of Experiments to choose a suitable architecture 79
6.2.4 Designed pipeline of our methodology to fit the proposed Bayesian Neural Network 86
6.2.5 Validation procedure for the proposed Bayesian Neural Network 88
6.3 Going Beyond: Extensions of the proposed Bayesian Neural Network 89
6.3.1 Non-compliance with the basic hypothesis of the model 89
6.3.2 Regularization and Automatic Relevance Determination through hierarchical models 89
6.3.3 Interpretation Layer 92

CHAPTER III: BAYESIAN NEURAL NETWORKS AS A PRICING MODEL FOR AIRBNB IN BARCELONA 94

7. Motivation to use information about Airbnb **94**

8. Fitting the proposed Bayesian Neural Network and validation procedure **95**
8.1 Data preprocessing 95
8.1.1 Adaptation of the dataset to approximate conducted transactions 95
8.1.2 Filtering, transforming and dealing with predictor's missing values 97
8.2 Finding a suitable architecture through Design of Experiments 101
8.3 Validation and comparison with the best linear model 108

9. Analysis of the results **114**

CONCLUSIONS 127

BIBLIOGRAPHY 133

ANNEX 139
Example Box-Behnken design matrix 139
Summary of the most relevant results of our methodology through an example dataset 140
Implementing BNN as a pricing model for Airbnb in Barcelona 159
Code required to fit a BNN with R and STAN 178

Summary

Abstract

The main purpose of this thesis is to disclose the potential of exploiting the synergies between Statistical Science and Machine Learning. In particular, we propose a specific feed-forward Bayesian Neural Network (BNN) as a parametric statistical model able to both yield better punctual predictions than linear models and handle uncertainty through more grounded intervals than the ones offered by bootstrapping conventional Neural Networks. On top of proposing a complete methodology (based on DoE for architecture selection and MCMC to conduct inference) to apply BNNs in real cases, we analyze, using theoretical arguments from Microeconomics, the positive effect on society that it would have to use BNNs as pricing model for peer-to-peer online marketplaces and, moreover, we implement them for the case of Airbnb in Barcelona.

Keywords: Peer-to-peer online marketplaces (P2P OM), Pricing models, Microeconomics, feed-forward Artificial Neural Networks (ANN), feed-forward Bayesian Neural Networks (BNN), Design of Experiments (DoE), Markov Chain Monte Carlo (MCMC) for Bayesian Inference, bootstrap and resampling methods.

Resum en llengua oficial

Xarxes Neuronals Bayesianes com a model de predicció de preus per reduir els costos d'informació en plataformes online de transaccions entre iguals

El principal objectiu d'aquest treball consisteix en demostrar el potencial de combinar el coneixement de l'estadística i de l'aprenentatge automàtic (*Machine Learning)* per tal de proporcionar noves eines que permetin aprofitar les oportunitats que les Tecnologies de la Informació i Comunicació han generat en els darrers anys. Aquestes oportunitats es basen en la creació de tot un univers de dades que està esperant a ser analitzat i convertit en informació útil.

En particular, aquest projecte es centra en les plataformes online de transaccions entre iguals (P2P OM) com ara Airbnb, Uber o Blablacar, ja que estant tenint un gran impacte en la nostra societat al modificar el procés mitjançant el qual adquirim béns i serveis. En aquest projecte, s'argumenta que, construint un model de predicció de preus mitjançant totes les transaccions realitzades en la plataforma, es podria proporcionar als usuaris oferents una eina de recomanació de preus per tal de determinar el preu de mercat del bé o servei que ofereixen,

de forma ràpida i objectiva. L'objectiu és aconseguir que els oferents prenguin decisions més acurades sobre els preus, apropant-los, així, a les preferències de la demanda i incrementant el nombre de transaccions realitzades.

Per tal de construir el model de predicció de preus es proposa treballar amb Xarxes Neuronals Bayesianes (BNN), amb l'objectiu d'oferir millors prediccions puntuals que el model lineal i, sobretot, intervals pel preu de mercat que realment capturin el comportament del mercat, cosa que les Xarxes Neuronals Artificials convencionals (ANN) tenen serioses dificultats per aconseguir-ho. Ara bé, i aquí és on es fan paleses les sinergies entre l'estadística i l'aprenentatge automàtic, en aquest projecte, a diferència del treball d'autors previs, es proposa les BNN com un model estadístic paramètric i, com a conseqüència, es desenvolupa tota una metodologia per tal de poder implementar-les en problemes aplicats, com ara el cas de les P2P OM.

Aquesta metodologia es fonamenta en tres pilars principals que són: La manera d'implementar els mètodes MCMC per tal de capturar la multimodalitat inherent en BNN i com determinar que s'estan obtenint mostres de la a posteriori, l'ús de Disseny d'experiments en comptes de validació creuada per tal de determinar una arquitectura adient per la BNN i, finalment, el desenvolupament de tècniques pròpies i adaptació de aportacions d'altres autors per tal d'entendre com està funcionant la BNN.

Finalment, s'implementa la BNN proposada d'acord amb la metodologia dissenyada pel cas de Airbnb a Barcelona, amb l'objectiu de demostrar tant el major rendiment i capacitats de la BNN respecte al model lineal i les ANN, com la utilitat que tindria un model de predicció de preus a l'hora d'ajudar els usuaris de les P2P OM a decidir un preu. A més a més, i fent ús de les tècniques d'interpretació de la BNN també s'observa com es poden extreure conclusions sobre el nivell de competència en cadascun dels barris de Barcelona i, a més a més, es pot explorar els efectes de canviar algunes característiques de l'apartament sobre el preu de mercat, cosa que pot ajudar als usuaris a decidir canvis i inversions.

Paraules clau: Plataformes online de transaccions entre iguals (P2P OM, sigles en anglès), Models de predicció de preus, Microeconomia, Xarxes Neuronals Artificials Directes (ANN, sigles en anglès), Xarxes Neuronals Bayesianes Directes (BNN, sigles en anglès), Disseny d'Experiments (DoE sigles en anglès), Cadenes de Markov Monte Carlo (MCMC) per inferència Bayesiana, *bootstrap* i mètodes de remostreig.

AMS classification

MSC Primary classification 62F15; Secondary classifications: 68T10, 62F40, 62G09, 62J05 and 62K20.

Table of Acronyms and Abbreviations

Acronym/Abbreviation	Original Term
ANN	Feed-Forward Artificial Neural Network
ARD	Automatic Relevance Determination
B2C e-commerce	Business to customer electronic commerce
BIC	Bayesian Information Criterion
BNN	Feed-Forward Bayesian Neural Network
C2C e-commerce	Customer to customer electronic commerce
CV	K-fold cross-validation
DoE	Design of Experiments
EC	Experimental Condition
GDP	Gross Domestic Product
HDI	Human Development Index
HMC	Hybrid/Hamiltonian Monte Carlo
ICT	Information and Communication Technologies
LM	Linear Model
MCMC	Markov Chain Monte Carlo
ML-hyperparameter	Machine Learning hyperparameter
MLE	Maximum Likelihood Estimator
OECD	Organization for Economic Cooperation and Development
P2P OM	Peer-to-peer online marketplaces
RMSE	Root Mean Square Error
VI	Variational Inference

Table 0.1: Table with the meaning of the most used acronyms and abbreviations.

Introduction

The main purpose of this thesis is to bridge Statistical Science and Machine Learning by proposing some techniques that outline the impressive synergies between those two fields. In particular, we aim to encourage statisticians to delve into Machine Learning and deal with it from the statistical point of view because, thanks to how Statistical Science treats uncertainty, several techniques from Machine Learning can be easily enhanced. With this, we aim to fulfill the underlying ambition of this thesis, which is to help develop Data Science because, due to the recent data deluge, it has become a key leverage to exploit business opportunities and, therefore, develop our society.

The most important contribution of this thesis that connects statistics and Machine Learning is the proposal of a feed-forward Bayesian Neural Network (BNN) as a nonlinear parametric statistical model and, along with it, a complete methodology on how to fit a BNN in applied cases. In fact, we introduce in this proposed methodology the second most relevant contribution of this thesis which, as the first one, aims to link statistics and Machine Learning. In particular, we propose using Design of Experiments (DoE) instead of k-fold cross-validation (CV) in order to find a suitable architecture for Neural Networks because DoE allows us to carry out a set of cost-effective experiments that endow us with much more information than CV.

Since our underlying goal is to help develop Data Science to have an impact on society, instead of just devising our BNN we also implement it to seize a setting-changer opportunity that peer-to-peer online marketplaces have offered to our society. In order to fulfill these two concurrent goals of both devising a BNN and explaining how it can be used to help develop our society, we have divided this thesis in three chapters.

In the first chapter, we explain the opportunity that peer-to-peer online marketplaces (P2P OM) have provided us and, moreover, we employ theoretical arguments from Microeconomics' market analysis to explain that using Data Science techniques, especially the BNN in this project, can reduce information costs by processing secondary-generated data and, therefore, enhance the market allocation of resources.

In the first section of this chapter, we delve into a theoretical discussion about the role of information on resource allocation. In the second section, we explain that ICT has currently reduced information costs and that, thanks to Data Science, a further reduction of them is possible. In the third section we focus on a particular case in which Data Science can help reduce information costs, which is the sector of P2P OM so, in this section we explain how

P2P OM are shaping our society and which is the opportunity that they offer to reduce information costs. Finally, in the last section of this chapter we explain how to seize that opportunity through a pricing model and, moreover, we discuss that the best model for it is the BNN.

In the second chapter we focus on BNNs and, in particular, in the first section, we review the foundations on top of which a BNN is build which are, Bayesian statistics and Artificial Neural Networks, and, moreover, we summarize the most important literature on BNNs. In the second section, which comprises most of this thesis' contributions we formulate the BNN as a parametric statistical model and, moreover, we explain the methodology that we devised to implement it. In particular, the main contributions are justifying that the proposed BNN is a parametric statistical model, proposing a different type of architecture for BNNs, using DoE to find a suitable architecture for Neural Networks and, finally, offering a renewed point of view to justify the use of MCMC methods in order to obtain the posterior distribution in a BNN. In fact, for this last contribution we propose a new complete schema on how to implement MCMC methods for BNNs (i.e. we justify the number of chains and initial values for each chain) and, moreover, we devise a technique that we called Multidimensional Convergence of MCMC methods to assess when our MCMC method is taking samples from the posterior distribution in complicated Bayesian models like the BNN proposed in this project. Finally, in the second subsection of the second section of this chapter we explain some extensions of our BNN that mainly focus on allowing the user to understand what is driving the prediction and, therefore, cast light on Neural Networks to reduce their functioning as a black-box.

In the third chapter we combine what has been explained in the previous two chapters by implementing our BNN as a pricing model for Airbnb (which is one of the most important P2P OM) in Barcelona, to show both how our methodology works and why is it relevant to use BNN as pricing models for P2P OM. Therefore, the purpose of this last chapter is to summarize in an example how Statistical Science can feed on Machine Learning in order to help develop Data Science and have a positive impact on society by seizing the opportunities that automatically-generated data offers.

Finally, we want to remark that we based our methodology on several experiments conducted on simulated and real datasets and, in the annex, we included an example that summarizes the most relevant results from these experiments and that, at the same time, helps to further understand the proposed methodology that we devised to implement BNNs.

Methodology

In order to devise the proposed methodology to apply Bayesian Neural Networks we have used logical arguments both based on previous research conducted by authors of Artificial Neural Networks and own ideas that arose during the undergraduate studies. Moreover, we conducted a whole set of experiments using simulated datasets and real datasets available on Kaggle and R's own repositories in order to empirically test the previous ideas and compose our methodology.

In order to quantify and compare the results from those experiments, we used Bayesian inference to conduct hypotheses test and take grounded decisions about how the proposed methodology would be.

The software used to conduct all the experiments and trials about Bayesian Neural Networks has been R and, specially, its interface to the Bayesian modelling software STAN. Even though conventional Artificial Neural Networks have several developed packages in R to implement them, BNN is a field that is recently awakening and, as a consequence, the code required to fit the proposed BNN in R had to be completely developed in this thesis.

About the infrastructure on top of which this thesis has been developed, the most relevant aspect is that it has been a local computer with multiple cores, in order to allow parallel programming and speed up Bayesian inference through Markov Chain Monte Carlo methods.

Chapter I: Pricing model for peer-to-peer online marketplaces

This chapter comprises the majority of the concepts from the bachelor's degree in Economics that are used for this thesis. In particular, we aim to remark the relevance of peer-to-peer online marketplaces in our society and how processing secondary-generated data from these platforms can enhance the market allocation of resources.

In order to do that, in the first section we review the main theoretical arguments about information costs and uncertainty from Microeconomics that will be used in following sections to explain how these platforms can deliver a more efficient allocation of resources. Afterwards, in the second section, we discuss that peer-to-peer online marketplaces are just a part of all the sectors that have been created by the Information and Communications Technologies and, therefore, there are other sectors in which processing secondary-generated data can reduce information costs.

In the last two sections of the chapter we focus on peer-to-peer online marketplaces and, to do that, we offer an overview about their history and their potential. Moreover, in the last section, we propose a pricing model to process the secondary-generated data in order to effectively reduce information costs and we introduce the Bayesian Neural Network as the most suitable model for this purpose.

1. Theoretical foundations about the benefits of improving availability of information

1.1 Overview of the relevance of information in market analysis

It is well known that availability of information plays a crucial role in market's efficiency, since it allows economic agents to decide optimally and, therefore, avoid misallocation of resources. In fact, the effects of information on a market's performance have been studied since, at least, Adam Smith time which means that has accompanied economics since its constitution as a science (Roncaglia, 2006).

In order to assess the relevance of information in market analysis, one needs to start with the most basic framework which is used as a benchmark: perfect competition. Even though there are different theories on perfect competition, as explained by Mas-Colell (Mas-Colell, 1998), they all agree that in order to obtain efficiency in a market, it is required, along with a set of other conditions, the existence of perfect information, which means that each economic agent

knows, at any moment, the exact market price of the good (Stigler, 1957). In this theoretical framework, the fact that each agent knows the price at any time, allows the market to set a unique price which is the one that ensures that the quantity sold is the maximum possible according to both the cost structure of the producers and the customers preference. Therefore, the market reaches its optimal equilibrium. In fact, that price is the minimum possible according to the cost structure of the firms because, otherwise, economic agents would enter the market and increase the supply due to the fact that perfect information allow them to know that they will obtain extraordinary profits by entering the market. For a more complete definition of perfect competition, please delve into (Mas-Colell, 1998) or (Mankiw, 2011), since it is thoroughly documented there.

As said before, perfect competition was the first model developed and, afterwards, several economists contributed to market analysis by studying the effects of eliminating, one by one, the different conditions that requires the perfect competition framework, aiming for a more realistic model that would deliver honest conclusions about the market performance. The main conclusions that are extracted from these theories is that, the further from the perfect competition framework, the worse is the market performance in terms of overall welfare, which mainly means that less transactions than it should are carried out and that they are completed at inefficient prices, meaning that a suboptimal amount of resources are allocated to this market and, therefore, there is a reduction on the overall welfare of society.

In this project, the focus lies on the effect of removing perfect information, which means that we want to assess the performance in a market with imperfect information and, specially, how reducing uncertainty (or increasing information) enhances the optimality of resources allocation. This is an uppermost topic, since the majority of transactions are executed in markets where uncertainty is common when taking decisions, as stated by Stiglitz (Stiglitz, 1989) and Mankiw (Mankiw, 2011).

1.2 Market performance under incomplete information

A market with information failure performs, by definition, worse than it would if the information was perfect. For instance, the consumers do not know how much they must exactly pay for the good, since they are not able to correctly determine how much they will benefit from it since they do not know the exact characteristics of it or, for example, if the competitors are offering a more attractive price for the same product. On the other hand, the producers also do not know at which price they should sell their goods, because, for example, they do not acknowledge the exact demand that they will face. Therefore, in this kind of markets, which are almost every in a real economy, the consumers end up paying too much

or too little for the products and, simultaneously, the firms supply too much or too little, causing a misallocation of resources.

There are several examples that allow to fully understand that this lack of information exists and that it has consequences on the transactions. For instance, in the labor market, if the interview on a candidate fails to determine that he/she does not master a skill that will be required in the job, and the applicant is hired, then there is a misallocation of resources, because the paid price, wage in this case, does not correspond to the productivity that the firm expects of the candidate. In another situation, it can also be that the candidate does not know which is the exact wage that is being paid for his/her position and, as a consequence, ends up with a lower salary for his/her effort.

In a more common setting, when buying groceries, the consumer does not know exactly how much fruit he/she must buy and maybe too much is bought and some of them perish before being consumed, which represents a waste of money for the consumer. In this same example, it can be also that the customer has several possible stores and is not able to choose because he/she does not know in which the relation price-quality relationship is more suitable for him/her and, as a consequence, does not buy at the most convenient one. Even though this example can be too trivial, imagine the same setting but with different investment options instead of groceries stores, the uncertainty may cause the investor to choose a suboptimal option due to the fact of finding the best (i.e. the cost of information) is too high. Back to the case of the groceries store, there is also lack of information for the seller, since he/she does not know exactly at which price he/she should establish the products according to both the underlying costs and the price of the competence.

Another example is the market of apartment rental in which the landlord can conceal some deficiencies of the accommodation and, therefore, make the tenant pay an overpriced rent for it. This same idea, could be applied for different cases such as second-hand cars. In these cases, the information failures come from the fact that some agent enjoys of a higher position in terms of information and uses it to sell the goods in a suboptimal price. The problem in these situations is that the consumer usually suspects that the seller may be trying to overprice the good and, as a consequence, they are prone to pay less for it, damaging those sellers that do not conceal the deficiencies of their good. As a consequence, the number of transactions diminishes both because the sellers have to diminish the price and also because the buyers are afraid of being scammed.

An example related to the situation of asymmetric information explained before is the cigarette manufacturer (or the financial agent) that does not appropriately inform the

consumers about the injuries that smoking can cause (or the possible costs if the financial product collapses) and, as a consequence, the buyer consumes too many of that product (or buys an inadequate financial product for him/her).

In all these cases explained before there is misallocation of resources, which means that either there are transactions that should not have happened at the established price, but they happened or, the other way around, that transactions that should have happened did not materialize. The two reasons that explain this "bad" decision-making are, according to Stigler (Stigler, 1961):

1. There is a limitation of information, which means that the buyer or the seller is not able to retrieve more information.
2. Obtaining more information requires an extra cost that the agent is not willing to bear.

Therefore, is not that the agents are taking bad decisions but more that they are choosing the best they can with the available information. For instance, the tenant decides to rent the apartment because looking for flaws could take too much time. Similarly, the firm decides to hire the candidate because it would be too costly to maintain the selection process for new candidates. In this same situation, the firm could improve the tests that must be held during the interview in order to better assess the skills of the candidates, but it could be too costly to design it, so the firm prefers to take the risk.

What it is clear from the examples before, is that if there was more information available or the costs of retrieving more information were smaller, then the overall uncertainty would diminish. As a consequence, the number of suboptimal transactions would decrease, because the price would be more accurate and the market would be closer to efficiency, increasing, this way, the market welfare. In other words, there is not a misallocation of resources because only the optimal transactions are conducted and, moreover, they are conducted at the price that fits both the costs of the seller and the preferences of the buyer.

Since this cost of retrieving information is diminishing the potential society welfare, many Public Sectors have instituted several interventions, through regulations, in those markets. For instance, in order to avoid the smokers' misinformation, the cigarettes manufacturers must indicate, in a visible and explicit way the effects on health that consuming that product can cause. Another example is the mandatory appearance of the expiration date in perishable products or the legal requirement of some certifications to enter specific jobs.

1.3 A specific case: Uncertainty during the ascertainment of the market price

As it can be seen from the examples before, the cases with imperfect information are manifold and, in this project, we will focus in a special case: The ascertainment of the market price, as Stigler (Stigler, 1961) proposed, which is part of the decision on the price made by the seller. This case is one of the most relevant in the literature both because appears in almost every market and because it summarizes the idea of decisions under uncertainty. For more information about the topic, please refer to (Stigler, 1961).

In this situation, the different sources of uncertainty that the seller faces when is deciding the price for his/her product are:

The first one is uncertainty about the own costs. Usually, the seller only knows an estimate of the costs that he/she has to bear, because the inputs' price fluctuates as the supply and demand for them are not constant. Moreover, the seller is aware that some unexpected changes can have an impact on the costs, but he/she is not able to predict them, so he/she has to assume a risk in his/her decisions.

The second one is the uncertainty about the competition costs. If it is hard to know the own costs, it is clearly more difficult to estimate the costs of the other sellers and, as a consequence, one must take decisions under uncertainty about the rivals' costs.

The third one is the uncertainty about the demand that will exists for the product. Even though in the theoretical framework of market analysis the demand is fixed and known, in real markets the seller must estimate it and, therefore, the seller must take a risk when deciding the price, because he/she has uncertainty about the demand that his/her product will have.

Finally, the last relevant source of uncertainty is the market price. Even though a firm can explore the price that the competitors are fixing for their products, it is costly to search for them and, since it is prohibitive to take into account the price fixed by each competitor, usually the firms reduce their search to a limited number of competitors, the ones that are closer to them. According to Stigler (Stigler, 1961), this source of information is the one called "ascertainment of the market price" and it is the one on which this thesis will focus. In fact, the relevance of this source of uncertainty is that it can summarize both the second and the third cause and, as a consequence, it is crucial for the seller to acknowledge it while deciding the price of the product.

Going through all those sources of uncertainty and having in mind that are others that exist and that were not included in that list, one concludes that the economic agents rely on their intuition when taking decisions, since they do not have enough information to know the optimal decision. Lots of influential economists have discussed about this topic and, one of the main contributors to the economic science, John Maynard Keynes, named this expectation-based decision-taking process under the term of Animal Spirits (Keynes, 1936), which in a few words (and in a very rough approximation), means that decisions are taken more by emotions (which create expectations) than rationality and, as a consequence, it is difficult to predict the human behavior. The literature on this topic is extensive and several authors like (Farmer & Guo, 1994) and (Akerlof & Shiller, 2010) have thoroughly analyzed, accounting both for the reasons of its existence and the implications that it has on the economy.

As it was discussed in subsection 1.2 (*Market performance under incomplete information*) and, applying it to the ascertainment of market price, if the sellers faced smaller costs of searching for the market price then they would be endowed with more information and, even though their decisions would still be somewhat unpredictable because there are still other sources of uncertainty, those decisions would be less based on instinct and more build on true facts. As a consequence, those decisions would be closer to the optimal and the market welfare would increase. Some sources of this enhancement of the market welfare would be that more transactions would be carried out, the dispersion of price would reduce, it would be easier to become a seller (so the price would reduce), the time of materializing a transaction would greatly diminish (because the decision would be easier to take), etc.

2. The information and communications technology revolution and its current reduction of uncertainty

2.1 Overview of ICT's positive effects on economies and how they relate to information availability

In the section before we explained the foundations of the benefits obtained from reducing the costs of retrieving information and, in particular, we analyzed the case of the ascertainment of the market price.

A clear example of the improvements made by the increment of available information has been the technological upheaval that our society has been experiencing from the last quarter of the 20th century. As it has been exposed by many authors like (Bauman, 2000) and (Castells, 1996), Information and Communications Technology (ICT) has deeply modified the way that

different agents interact with each other in a society and, in particular, it has transformed our economies into a new paradigm called the "information economy". The implications of this revolution are manifold, and in this project, we assume that the reader is familiar with them. For an in-depth analysis of the "information economy" please refer to (Porat, 1998).

In order to evaluate the positives effect of this major change in our economies, many authors like (Lee, Gholami, & & Tong, 2005) and (Doucek, 2010) have studied the impact that it had on the Gross Domestic Product (GDP). Even though there are lots of controversy about whether the GDP is a good indicator of the performance of an economy (see a complete discussion on (Wilson, Tyedmers, & Pelot, 2007)), other indicators like the Human Development Index (HDI) are not completely accepted as a solution by the literature as (McGillivray & White, 1993) discusses. As a consequence, establishing the exact effect of ICT on the development of our economies has become one of the most grueling tasks that applied economist face. Even though the exact effect is difficult to assess, it is clear that, as the Organization for the Economic Cooperation and Development (OECD) states in its Digital Economy Outlook (OECD, 2017), ICT has significantly enhanced our economies, and, at the same time, it has increased our quality of life. This assessment of the effect has arisen lots of debates and some empirical studies have been carried out, like the one conducted by Jensen (Jensen, 2007).

There are many reasons that explain this positive impact of ICT and most of them can be analyzed from the perspective of the reduction of information costs, as explained below.
For instance, it has increased the size of the markets and, as a natural consequence as it was explained by Adam Smith (Smith, 1776), this has increased the markets' efficiency because specialization has increased, the costs have been reduced and the volume of transactions has raised. All of this translates to a higher number of easily obtainable products and, furthermore, at a significantly smaller price. In order to envision that the reduction of uncertainty is behind this market expansion, imagine the situation of importing goods from another country. Thanks to the capabilities that ICT offer, shipping a cargo from another country is easier nowadays because the firm can contact, at any time, the counterpart of the transaction (so negotiation is faster and easier and there is not as much uncertainty as before) and it can improve its stock management because the firm can even track by GPS the exact position of the ship.

In fact, this reduction of uncertainty in international trade has made viable one of the most relevant sectors developed during the *Information Economy*: e-commerce. Only analyzing the effects of e-commerce, one can understand the relevance that the reduction of uncertainty

has on economies, not only because it facilitates more transactions but because it opens opportunities to constitute entirely new sectors.

This reduction of uncertainty has altered many markets and created many more, not only e-commerce. Other sectors that have been enhanced are travelling (because now the tourist can easily find an accommodation at a reasonable price, without spending too much time searching for it), learning (because there are lots of material online to learn languages, techniques with different online courses… it was harder before to find the exact academy that would suit the consumer needs), entertainment (before going to the cinema now the consumer can read reviews and watch trailers about the movie, so the choice of which movie to see is close to the optimal) and on and on. The common denominator of this changes is that now, since information is easily available, more transactions are completed and the overall welfare increases.

Going back to the main topic of this thesis, which is the ascertainment of the market price, it is obvious that the amount of information that Internet has made available has had a direct impact on the decision carried out by sellers about the price of their products. For instance, imagine a restaurant that is wondering about the price of the daily menu. Before Internet arrived, the owner had to visit the restaurants of the competition or, at least, pass by them and look for the price of the menu and, since this process was costly in terms of time, usually he/she would only pay a visit to the closer ones, even though it is clear that they are not the only competition that the restaurant face. In contrast, now the owner can retrieve a large amount of information in much less time, since he/she can look up the competitors' websites and obtain the price that they are fixing. Since now the cost of retrieving this information is smaller, the owner obtains information of more restaurants and processes it to take a more accurate decision about the price.

2.2 From the data deluge to the useful information: The rise of Data Science

In this subsection, we revise the main bottleneck that faced the reduction of uncertainty offered by ICT: The existence of too many data on Internet slowed down the absorption of useful information. Furthermore, we analyze how society has tried to overcome it and how much is yet to be done, knowing that a further reduction of uncertainty would yield significant improvements in welfare.

As the reader knows, one side effect of digitalizing processes and letting ICT take control of different activities is the amount of data that is automatically generated, which has great potential but is not currently completely exploited by our society. The main reason of its

potential is that information can be extracted from data and, if there is an increment of information then uncertainty reduces. Note that data itself is not information, because in order to be information, it needs to be useful, as clearly explain Bellinger, Castro and Mills (Bellinger, Castro, & Mills, 2004) in their straightforward article.

As said before, a part of this amount of secondary data generated would be able to reduce uncertainty if correctly treated and there are some examples of firms that have exploited their possibilities by using the stack of opportunities that Statistics and Data Science offers.

During the digitalization process, several hotels made available reservations through their websites, as a way to enhance demand of their rooms. This reduced the uncertainty of the customer about the price, so, following the rationale exposed in the subsection 2.1 (*Overview of ICT's positive effects on economies and how they relate to information availability*), it increased the number of reservations. After a time, since most of the hotels did the same, the customer needed to spend too many time to retrieve the best option for him/her, so he/she decided to only look for reservations in some hotels. Even though the reduction of uncertainty was important, the best was yet to come, and it arrived when some firms decided to exploit the secondary data generated. Those firms that encountered that niche are now successful companies like Trivago, TripAdvisor, Kayak… and the origin of their success is that they offered a reduction of the cost of retrieving information in exchange of a small surcharge for the reservation. It is obvious that is not the only reason of their success because after they earned some reputation they incorporated special offers made by the hotels with those rooms that, otherwise, would not have been sold. The main positive consequence is that the overall welfare increased, because booking a hotel room nowadays is much more easy and straightforward, because the customer receives only relevant information, which translated into a significant increment of worldwide reservations. This example is not unique for hotels, since it expands also to flights, car rental and many other sectors.

Another successful example of the exploitation of this secondary data are recommender systems, which have greatly increased the transactions of the e-commerce platforms because they reduce the uncertainty of the customer. With recommender systems, the customer has easy access to those goods that is likely to buy, which means that the expenditure of time searching for the desired product diminishes. Again, like economic theory exposed, when uncertainty is reduced, market efficiency increases.

Data Science has offered other opportunities to take advantage of those amounts of secondary data generated. For instance, some consultancy firms perform text mining on all the opinions posted on websites and summarize them for some clients, so they can

understand which is the feedback not only of his/her business but also the competitor's one. Another example is performing sentiment analysis on tweets, which is usually used by investment firms to predict the evolution of financial products.

However, as it was said at the beginning of this subsection, we believe that the whole potential of this data is yet to be discovered and the aim of this project is both to reveal an opportunity related to the ascertainment of the market price and to propose statistical tools to extract the data's usefulness in that setting. In order to explain this opportunity, we need to delve into peer-to-peer online marketplaces platforms.

3. Peer-to-peer online marketplaces and their intrinsic potential to reduce uncertainty when ascertaining the market price

One of the main changes that ICT has caused in our economies is the energetic spread that online peer-to-peer marketplaces platforms have had in the last years. This has been a paramount change because the market, which is the key instrument to satisfy our needs as proposed by Adam Smith (Smith, 1776), has migrated in many sectors from a physical place to a digital one, in which the information flows at extraordinary high rates and the costs of retrieving it are almost insignificant. In fact, the efficiency of these digital markets is overcoming traditional ones which, as a consequence, are becoming more and more obsoletes as time passes by. However, this revolution is still starting and traditional markets still maintain a huge share of the transactions, since there are layers of population that, because of their education, age and other characteristics, are reticent to shift to those online marketplaces.

Before discussing the potential that these online peer-to-peer marketplaces platforms, (or also called customer to customer -C2C- e-commerce) offer to reduce uncertainty when ascertaining the market price, let's focus first on defining and analyzing, in a deeper way, which is their nature, origins, projection and also controversies that have arisen around them.

3.1 Overview of peer-to-peer online marketplaces platforms

3.1.1 Definition, relationship with sharing economy and examples

As it has been explained when introducing this section, one of the characteristics that define a peer-to-peer online marketplace (from now on, P2P OM), is that they function in a digital environment in which buyers can easily explore the market. However, this feature is also held

by other sectors, like e-commerce, so it is not sufficient to define a P2P OM. Please note that when we refer to e-commerce, we mean business to customer (B2C) e-commerce and, with P2P OM we are speaking about C2C e-commerce.

The second characteristic of a P2P OM, also contained in the name, is that the user or in other terms, the economic agent, can be, at the same time, both a buyer and a seller in the sense that he/she can obtain some products through this marketplace but, at the same time, can offer some others which are related to the ones bought. Therefore, there is not the same hierarchy and relationship between agents than in traditional markets.

Even though we have defined what is a P2P OM, in order to fully understand the concept, it is often useful to define what is not a P2P OM because, with it, we avoid some misunderstandings of the idea. Before, we commented that P2P OM should not be confused with e-commerce (i.e. B2C e-commerce), because in B2C e-commerce the transactions are not peer-to-peer, as the roles of firms and customers are clearly defined.

Another important distinction that must be taken into account is that P2P OM is not the exactly same concept as sharing economy, even though the boundaries between these two concepts are very fuzzy as most of the sharing economy platforms are based on marketplaces platforms. Therefore, this relationship between P2P and sharing economy could be summarized with the following statement: P2P OM are an online framework to carry out transactions between equals (in a sense that the user can be both buyer and seller), which is a tool largely used by sharing economies platforms. After this, the reader must be wondering about what exactly is the sharing economy if its definition is not that is a P2P OM.

The literature on sharing economy has vigorously flourished during the recent years and, even though many authors share some scents of the concept behind it, the exact definition has not been ascertained yet. While some authors use it as a broad concept to refer to online transactions between equals (which resembles to a P2P OM) (Taeihagh, 2017), others deliver a narrower definition which relates to Collaborative Consumption harnessed through online platforms, like (Hamari, Sjöklint, & Ukkonen, 2016) and (Belk, 2014). In this project we treat sharing economy from the perspective of the second definition.

The discussion on the perimeter of sharing economy would be too extensive for the purpose of this project, in which we analyze P2P OM platforms. Therefore, the most suited definition of sharing economy for this purpose, based on Rachel Botsman explanations from both her book (Botsman & R., 2010) and Ted-Talks (Botsman R. , 2010), is that a good is not only consumed by one individual but for a whole group so the social benefit of that good is

maximized, because the costs of maintaining, using and consuming are shared among the group. Thanks to the easiness that ICT has endowed our society to connect people that want to share a good or service, the number of sharing economy platforms have substantially increased in the last 10 years.

Therefore, in order to distinguish between P2P OM and sharing economy, one needs to have in mind two concepts: The end and the means. In sharing economy, the end is to reduce the cost of consuming an underused asset by employing it jointly with other individuals and, in order to do that, it needs a framework to connect those individuals, which is usually a P2P OM. However, the mean for this end is not always a P2P OM, so there are sharing economy platforms that do not include P2P OM (like Couchsurfing). Likewise, there are some P2P OM in which the users do not interact for the purpose of reducing the cost of consuming one underused product, instead, their end is solely to obtain profits from a product, so these platforms are not part of the sharing economy environment. With this rationale, secondhand marketplaces would not be part of the sharing economy, as Beck states (Beck, 2017), but they are a P2P OM platform.

Following the argument above, the broader concept is P2P OM, while sharing economy is a more limited one. In fact, the explanation above helps to understand why the whole sharing economy paradigm has arisen so many controversies, because a platform is not, by definition, from the sharing economy, since it depends on the purpose that user gives to it. Depending on the product sold in the market, the user has a different tendency to have a “pure sharing economy attitude” and, as a consequence, some platforms are considered more from the sharing economy environment, while others are not.

In order to assess this idea, imagine three cases: Airbnb, Uber and Blablacar. In the case of Airbnb, imagine that there is a user that lives in an apartment and has an empty room. Moreover, he/she is not looking for a long-term roommate. If this user posts this room on Airbnb, then it will be part of the sharing economy. Otherwise, imagine an Airbnb user that has an empty apartment in which he/she does not live and, in order to make the most of it, he/she decides to rent it through Airbnb instead of selling it. In this case, the transactions derived from this apartment should not be deemed sharing economy. The case is similar with Uber, imagine the user that has to drive to his/her job and decides to pick up some other users along the way, then all the payments should be accounted as sharing economy. On the other hand, if the driver decides to spend all afternoon doing rides, then it would not be part of the sharing economy. Since in Airbnb and Uber the second type of users are common, lots of controversy have arisen around them, because those firms catalogue those second type of transactions as if they were part of the sharing economy. Even though this controversy exists,

the reader cannot ignore that both Airbnb and Uber are P2P OM and, as explained before, they have increased the market welfare due to the reduction of uncertainty that they have caused.

In the case of Blablacar the situation is the same, there are users that need to make a long ride and find people to carry out the trip in a cheaper way, while there are others that can decide to execute several trips on a week just for the purpose of earning some income from it. However, in this case, the amount of transactions like the second are a very small part of the whole, so the situation is different to Airbnb and Uber and, as a consequence Blablacar is usually considered a sharing economy platform or, at least, more than Airbnb or Uber.

The common denominator of these examples is that, all of the transactions, even if they are performed under the concept of sharing economy or not, all were performed through a P2P OM. Of course, some sharing economy platforms, like Couchsurfing, conduct transactions through a different type of platforms than P2P OM.

Finally, in order to ensure that the difference between the two concepts has been ascertained, a Venn diagram of all peer-to-peer online transactions in Figure 1.1 is presented. In order to delve into the controversy of sharing economy platforms, please refer to (Laurell & Sandström, 2017), (Malhorta & Van Alstyne, 2014) and (Schor, 2016).

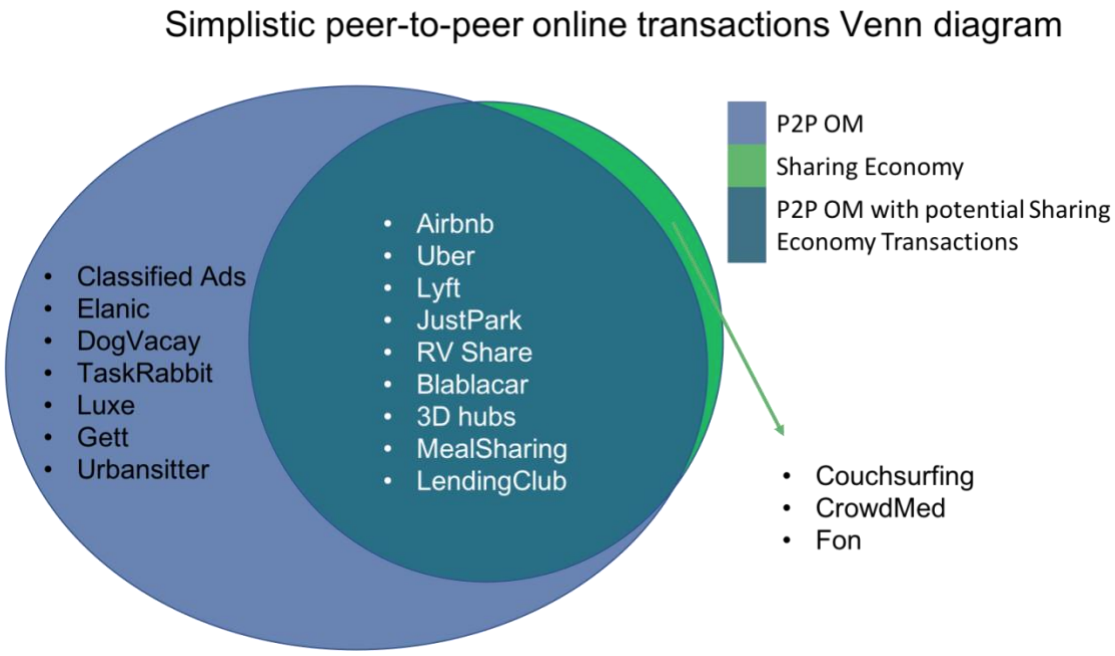


Figure 1.1: Venn diagram of all the peer-to-peer online transactions, grouped depending on if they are considered P2P or Sharing Economy

3.1.2 Origins, evolution and impact

Even though P2P OM and B2C e-commerce are not the same concept, the evolution of both types of platforms has been almost similar for many years, since they both use the same tool: an online platform to perform transactions. Because of the inherent relationship of P2P OM and B2C e-commerce during the first years of their development, we we'll use examples of both to explain the origins of P2P OM platforms, because without some B2C e-commerce firms it's impossible to fully understand P2P OM origins.

The first P2P OM ever created was the pioneer Boston Computer Exchange, founded in 1982 and in which second-hand computers could be traded (Moraru, 2008). However, due to the fact that computers were becoming obsolete at a fast pace and, also, to the fact that this platform was not able to constitute a sufficiently big community, it closed during the 1990s. What is remarkable of this platform is that it released summaries about the prices at which the most famous computer models were sold because the founders knew that they had a comparative advantage of information in comparison to traditional stores, so they wanted to exploit it (Banks, 2008). Another online marketplace, but in this case it was a B2C e-commerce, was the Videotex marketplace, also founded in 1982, and which arose lots of attention because its capability to offer TV entertainment to users, as explained in both (Campbell & Thomas, 1981) and (Talarzyk, Widing, & Urbany., 1984).

Even though those were the first firms, the online marketplaces, P2P or not, made a huge advancement when a spearheading firm in the ICT sector from the United States, CompuServe, launched their e-commerce platform in 1984, called Electronic Mall. This is a relevant event in the evolution of online marketplaces because CompuServe, through the many resources invested in marketing, made the society aware of the possibilities that e-commerce could offer (in fact, they spread some easy to follow tutorials to maximize their impact (CompuServe, 1984)).

During the next 10 years, the number of new online marketplaces was very small because, even though they had lots of potential, they faced an important bottleneck: Connecting to them was difficult because customers needed to log into a terminal and access the Bulletin Board System (BBS) which hosted the database with all the products of the marketplace. In terms of information costs, which is the main pillar of this thesis, the cost of accessing those online marketplaces to ascertain the market price and the quality of the good was too costly compared to retrieving the goods from the traditional market.

The innovation that allowed online marketplace to overcome that bottleneck is the one that has also contributed to the creation of our society as it stands now: The World Wide Web (WWW) browser. The history of Web browsers is a very interesting one and we encourage the reader to delve into that topic, but it will not be covered in this project because it is too extensive for the purpose of it. What is relevant for this thesis is that the idea of a Web Browser was considered, at the time of their appearance, a revolution itself and many books, tutorials, courses about it were released like (December & Randal, 1994) or (Catledge & Pitkow, 1995). Researchers, CEOs, politicians and influent people in general started to truly envision the network connected society.

In fact, the rise of WWW browsers started in 1992 and, with Mosaic (a graphical web browser that launched in 1993), the spread of WWW and Internet was implacable. Thanks to the easiness that WWW browsers offered to the population to connect to the internet, online marketplaces started to flourish and several setting-changer firms were created, like eBay (firstly called AuctionWeb) and Amazon (firstly called Cadabra), both in 1995. Apart from the development in the US, which was the core of all the developments explained before, the online marketplaces spread rapidly along the world. In fact, in 1996, online marketplaces (B2C e-commerce ones) where opened in places like India (IndiaMart) and Korea (ECPlaza).

Even though WWW was the innovation that allowed online marketplaces to be a success because users could access them easily, there was still another technological innovation that had a crucial role in the development of online marketplaces, because it made it even easier to operate through these online platforms. This innovation was the possibility to execute payments online, so transactions could be accomplished within seconds and with complete certainty that they would be executed. The firm that spearheaded this innovation was PayPal and some variants of it were adopted by competitors and even financial companies.

After this major technological innovation, several online marketplace firms were created and the pioneers that were already established, like Amazon and eBay, instituted the path to their extraordinary success. As we explained before, when we say online marketplace we refer to both B2C and C2C e-commerce because, until now, there was not much difference between the development of both sectors.

Once we have explained the technological tool that P2P OM uses (which is the same that e-commerce), now we can delve into the exact evolution of the P2P OM platforms which, as we defined at the beginning of this section, are online frameworks to carry out transactions between equals, in a sense that the user can be both buyer and seller.

The key factors that allowed the rapid spread of P2P OM during the 2000s and, specially, 2010s are, mainly, three and they will be discussed in the following paragraphs. The first one is that the access to P2P OM platforms became easier at an incredible pace and, the second one, is that ICT modified our society into a networking-based one, in which the individuals are prone to share information about them with others, specially through social networks. Finally, the third main reason is that the price of the goods was relatively cheap compared to traditional markets, even though this reduction of price heavily depends on the sector.

One of the main reasons why P2P OM platforms are easy to access is because of the business model that they adopted. Since those platforms need a sufficiently big number of customers and buyers in order to correctly function, because if not the demand disappears as it is not able to find the goods that it is looking for, they opened the access to everyone, without extra costs than having internet connectivity, so searching and retrieving information in those markets is costless. Another reason why the access to P2P OM was easy during the 2000s, and maybe a more important one, is because of the third crucial innovation: The smartphone. The important date on the timeline is 2007, because even though before that smartphones existed, it was with Apple's iPhone when smartphone started to exhaustively spread in our society (Sarwar & Soomro, 2013).

About the second key factor, is clear that P2P OM were able to find a sufficiently big number of users, in part, because of social networks sites, which have altered the way individuals in a society connect with each other. As explained in (Ellison, 2007), there are many reasons of the social networks sites success and, of course, lots of consequences so, for an in-depth analysis of the social network sites, please refer to this magnificent work done by Ellison (Ellison, 2007). In order to focus theses consequences to the topic of this thesis, the two most important consequences that social networks had on P2P OM are, firstly, that they modified our society in a way that contacting some unknown person is common, so dealing with strangers through P2P OM became less of a problem and, secondly, that social networks sites became a cheap and efficient channel for advertisement and spread of P2P OM platforms, specially through the users' opinions and impressions that they were willing to explain freely. As explained by Lu, Zhao and Wang (Lu, Zhao, & Wang, 2010), it has created the perfect environment for the booming of P2P OM platforms.

Finally, the smaller costs of maintaining the marketplace infrastructure, which is basically a website and a constantly updated database, the fact that in most of P2P OM intermediaries disappeared, the fact that taxes are difficult to be applied at a user level on those platforms and, finally, the fact that most users use P2P OM as a side tool to earn some extra income, but

it is not their main source of earnings, has caused the price of P2P OM to be low and, as a consequence, has seduced several consumers of the traditional marketplaces.

As said before, the number of P2P OM that appeared during the 2000s was huge compared to the period from 1995 to 2000, when pioneers were instituted. However, the fastest pace of P2P OM creation was during the 2010s thanks, as explained before, to the developments of smartphones. In terms of reduction of uncertainty, as we have focused during this project, smartphones allowed a further reduction of the costs of retrieving information. In particular, one of the main characteristics of P2P OM is that searching for information only has the cost of the time spent (which is a small amount because in less time one can visit more products than in traditional markets) and, with smartphones, now users can retrieve information of the products posted on P2P OM during the spare time (like waiting for the subway, at queues…) that they have, in which opportunity cost is significantly smaller.

In order to envision the evolution of P2P OM we have taken an almost exhaustive sample with the most famous platforms, because they are the ones that actually had an impact in society as they were able to build a sufficiently big community. In Figure 1.2 we have plotted the number of platforms created per year and also the year of the three main innovations discussed during this subsection.

In order to decide which platforms should be incorporated in the sample, we first looked online on several websites about sharing economy and P2P OM and, afterwards, we corroborated the information found there through a very influential website[1] that tracks almost any startup, with a worldwide scope. However, assessing whether a platform has been influential or not is a grueling a task, since there is not many information about them and, as a consequence, the decision on that was the size of related forums and the amount of information about the platform in this influential website[1], because the more influential, the more users post information in that website. Therefore, we want to warn the reader to avoid using the graph from Figure 1.2 as an exact estimate of the number of platforms created per year, because it is not exhaustive. However, the graph serves the pursued purpose: to take an approximated glance at the evolution, in terms of number of platforms, of the P2P OM sector.

Some other considerations that the reader should have in mind when interpreting the graph is that we took, as maximum, firms founded in 2015, because it is too difficult to assess which of the firms founded in 2016 and 2017 will became influential in the next years so, including

[1] https://www.crunchbase.com

all of them would have added some bias, because we only took the influential ones that were founded before 2016.

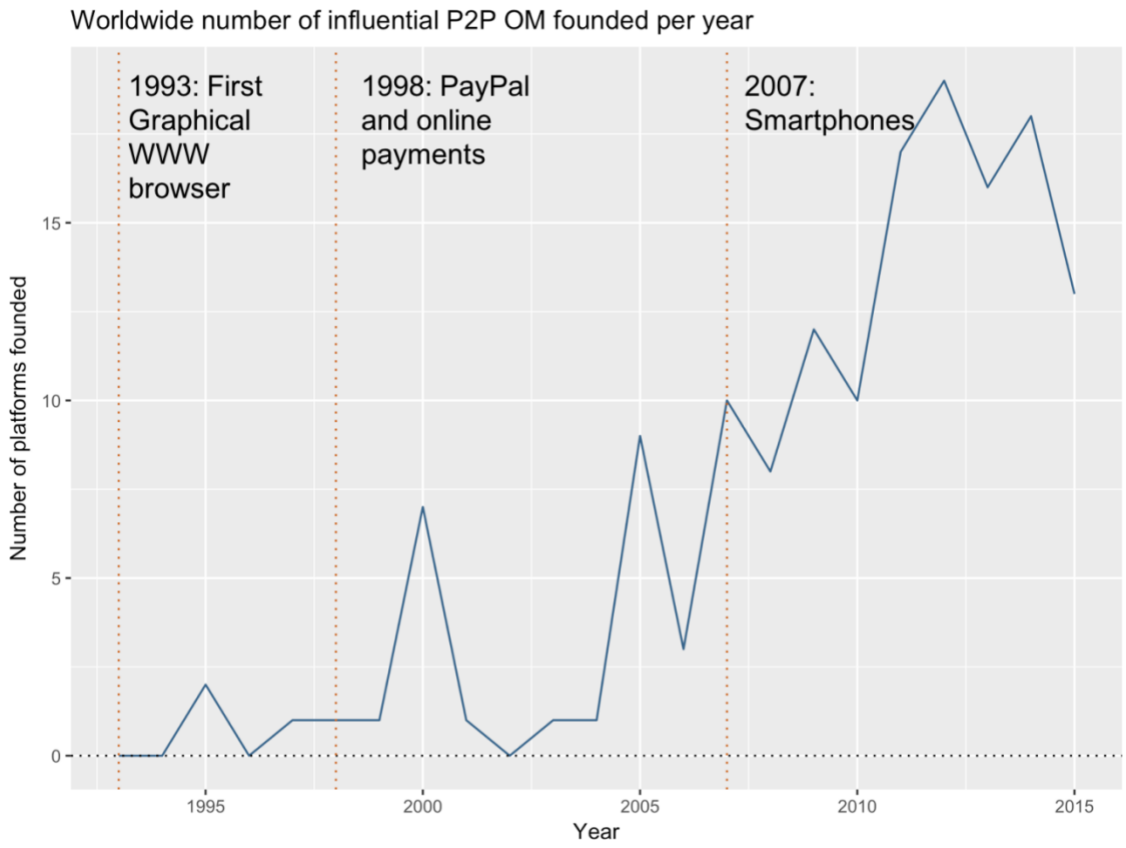


Figure 1.2: Number of P2P OM founded each year until 2015. The focus is mainly on the US and EU since they were pioneers in this area.

As we can see in Figure 1.2, each of three technological innovations which are WWW browsers, online payments and Smartphones has had a relevant impact in the creation of P2P OM platforms. Moreover, this graph shows that every year more and more P2P OM, usually startups, are brought to the market and, as a consequence, one should expect a huge role of them in the economy that is to come in the following years. Therefore, extracting as much social profit from them as possible should be a top-priority task if we want to fully exploit the benefits of ICT revolution.

Now that we have defined the reasons of P2P OM flourishment and also their evolution, it would be interesting to take a glance at their overall impact on our economy. However, as it happened with ICT, assessing it is a very difficult task. First, we will approximate the volume of their impact and, afterwards the expected evolution of this impact.

The first main problem to assess the volume of P2P OM impact is that usually it is accounted only under the term sharing economy because, as we explained in this section, almost all sharing economy, which is a trendier topic than P2P OM, usually operates through a P2P OM.

In fact, if one does some research about the performed studies on the effect of P2P OM, he/she will only be able to find ones for sharing economy, since are the projects that are being demanded by firms and governments due to the attention that the topic has generated.

However, these studies on sharing economy, like (European Commission, 2017), usually use all transactions made by "sharing economy" platforms, because it is impossible to analyze their effect otherwise. For instance, they use all the value generated by firms as Uber and Airbnb, even though, as we explained before through examples, a great part of their transactions is not from the sharing economy concept. As a consequence of this, these studies could deliver a rough estimate of the impact of P2P OM. According to the Venn diagram of Figure 1.1, these studies aim to estimate the intersection between the concept P2P OM and sharing economy and, as a consequence, can be used as a lower bound to estimate the impact of P2P OM.

Even though studies like (European Commission, 2017) cannot be accurate, because they leave out some P2P OM and because the task of assessing the impact is truly grueling and complicated as stated in both (Hall & Pennington, 2016) and (Coyle, 2014), it will serve the purpose of sketching out the impact of P2P OM on our economies. According to (European Commission, 2017), which used P2P OM based on accommodations, transportations, household services, professional services and finance, states that, for 2015 in Europe, these platforms generated revenues of nearly 4 billion euros and, the important part, that facilitated transactions that add up to a total value of 28 billion euros. Moreover, this study estimates that during 2016 those firms have almost doubled their revenues, which clearly indicates the thriving development that has been explained during this thesis.

Apart from the fact that this study only focuses on some P2P OM firms, so it underestimates its impact, one has to have in mind that, in order to assess the net increment of welfare that these platforms have produced, one should discount the value of the transactions that migrated from traditional markets to P2P OM.

In order to assess the future impact of P2P OM firms, it is useful to check the evolution, in terms of turnover, that those firms have experienced during the last year. However, we cannot use studies developed on sharing economy, because there are few of them and the concept is different among them. Furthermore, there is a more reliable source of information that could be used to estimate the growth of P2P OM firms: The Ecommerce Europe association[2], which is an association that represents more than 75000 companies that sell online in Europe.

[2] https://www.ecommerce-europe.eu/about-ecommerce-europe/

The main reason that this information is more reliable is because, in comparison to data on sharing economy, all transactions are officially registered, since they are subject to different taxes as VAT. However, we will not be able to assess the exact evolution of P2P OM (also named C2C e-commerce), because this institution only keeps track of what we have been calling e-commerce in this project, which, in their jargon it is called B2C e-commerce. Therefore, we will use B2C e-commerce behavior as an approximation of C2C e-commerce.

The main argument that supports the use of B2C e-commerce growth rates to estimate C2C e-commerce growth rates is that, as it has been explained during this section, that both sectors use the same technological tool: an online marketplace. Therefore, the growth of both sectors is subject to the increment and spread of this tool among society, because it will increase the number of users. In other words, the main reason that B2C and C2C e-commerce are expanding is because the population is becoming more Internet literate (in the sense that they use it with more frequency) and the mass of users is constantly enlarging, therefore, both sectors should growth at a similar pace. Of course, this is an important assumption and the estimate will not be accurate, but it is useful for the purpose of this subsection: to sketch out an idea about the development of C2C impact, not to assess its exact evolution, because it would be too extensive for the purpose of this project, in which we focus on further exploiting the possibilities of C2C through the opportunities that Data Science offers.

According to the report offered by the Ecommerce Europe association in 2015 (European Ecommerce, 2015), the turnover of B2C e-commerce in Europe increased a 13.6% respectively to 2013 and a 13.3% from 2014 to 2015. Furthermore, they forecast that the turnover of 2016 will be a 12% higher than the one accomplished during 2015. It is impressive the growth that this sector is experiencing and, even though rates are slowly diminishing, an increment of more than 10% per year is completely astonishing. As a consequence of this, it is clear enough that e-commerce, both B2C and C2C will further shape our society because every day more and more transactions are executed through these platforms.

As a conclusion, in order to estimate the size of the C2C e-commerce (P2P OM), we had to use an instrumental variable, which is the size of the sharing economy in Europe. However, since the concept of sharing economy and P2P OM is fuzzy, the data that has been used, even though it is said that comes from the “sharing economy”, actually is a suitable metric for P2P OM, as explained before. When it comes to assessing the expected growth in the future through the momentum accumulated during the last years, we were not able to use data about the sharing economy, because it was too scarce and the fuzziness of the concept complicated it. However, we were able to retrieve valuable information from B2C e-commerce and use it to envision an approximation of C2C e-commerce’s future, basing these conclusions

on the assumption that both B2C and C2C e-commerce depend on the same leverage: The growth and expansion of Internet use.

From this information, the main conclusion is that C2C e-commerce will be an uppermost factor in the future of our society and, any activity aiming to improve its efficiency, as the one that this thesis proposes (use Data Science and Statistics to make the most of secondary data generated), will greatly benefit the welfare of our society.

3.1.3 Controversy generated around these platforms and efficient regulation

From their beginning, many P2P OM have been involved in many debates about their nature and, particularly, their ethics, suitability as fair competition to firms established in traditional markets and their input to society.

However, according to what has been explained during this section, the reader should be able to understand that most of these debates are based on sharing economy concept, instead of P2P OM. In fact, as it has been explained before, the uppermost reason that is arising all these conflicts is that many users employ some auto-denominated sharing economy platforms to execute transactions that clearly do not fit the definition of sharing economy. This practice of disguising transactions under the concept of sharing economy has a relevant impact on the development of this sector, because it reduces the society trust on these platforms and, as a consequence, the user community growth is slower than it should.

Since this controversy was explained before in this section and the fact that this thesis focuses on P2P OM, we will discuss the controversy arisen around P2P OM, and not around sharing economy platforms.

The main controversy generated around P2P OM is that they are usually considered unfair competition by the firms that operate in the same sector but through a traditional market. The pillars that sustain this claim come from many sources, for instance, players in traditional markets need to fulfill some legal requirements that causes to increase the final price, while in P2P OM these conditions are usually not met, and the price is smaller. Another reason that sustains the idea of unfair competition is that, like before, transactions conducted in these P2P OM platforms are not taxed and, as a consequence, the price offered is significantly lower. The fact that many transactions are not taxed and monitored by any authority has a wide range of negative effects on economy. Mainly, the tax collection falls as transactions that were taxed in traditional market, like a taxi ride, shift to the grey economy.

Furthermore, other usually accounted downsides of P2P OM are that they reduce the work standards (as anyone is able to become a seller, without protection), and that they downgrade consumer security (because scams are more possible).

Even though all downsides of P2P OM stated before (which are unfair competition, tax impact, reduction of work standard and consumer security) have grounded reasons to exist, some of they are not accurate, which means that the magnitude of the negative impact is smaller. Moreover, as it has been exposed during this section, P2P OM offer a great deal of positive effects on economy both because they reduce uncertainty so more transactions are conducted and because unused assets are brought to the market, so there is a reduction of the prices and an increment of society welfare. Furthermore, these P2P OM platforms have given the opportunity to several layers of population to find sources of complementary income that otherwise would not have been possible.

Since there are both positive and negative impacts, the decision about how to regulate P2P OM is troublesome. As both Jericho (Jericho, 2016) and Minifie (Minifie, 2016) discuss, both of the extreme decisions are economic failures. On the one hand, forbidding these platforms prevents our society to endorse technology and the modernization of different sectors, being the main consequence that improvements on society welfare are avoided because innovation is ignored. On the other hand, letting them operate without restraints and an ad hoc regulation will cause both social conflict, endorsed by firms that operate in traditional markets, and an unfair acquisition, by the P2P OM platforms, of the complete welfare increment generated, which means that profits that should be shared among all economic agents would pile up in these private firms.

Therefore, an efficient regulation on P2P OM should be implemented, even though its development is complicated and costly (European Commission, 2017). There are basically two reasons why establishing adequate regulation is difficult in our society. The first one is that these platforms are not completely socially accepted because traditional markets are spending resources to guarantee its survival by biasing the public opinion. The second one is that all these platforms as Figure 1.2 shows have developed in recent years so there is small knowhow about how to deal with them.

In order to become acquainted with these platforms and, therefore, solve the second main reason that complicates regulation, the public sector should track these firms by increasing the amount of information about them and their impact because, as seen during this thesis, the quantitative information about this sector is scarce and, as a consequence, yielding

efficient and accurate regulation is impossible if the magnitude and nature of the sector is unknown.

On the other hand, in order to prevent overregulation of P2P OM platforms (like forbidding them), the public sector needs to reduce the negotiation power that traditional markets have, since they are the main economic agents that are slowing down this innovation wave through the bias that they impose on public opinion. As we have said before, there are grounded reasons to claim that P2P OM platforms require more regulation which are, specially, the fact that they present a threat to fair competition and that they are a risk for work standards. However, other reasons like the erosion of tax collection and reduction of consumer security are not so realistic.

For starters, even though it is true that there is a reduction on VAT income because many transactions conducted in P2P OM are not subject to it, it is not proven that tax collection has been reduced. The main reason is that P2P OM yield outstanding profits (because it allows a great deal of transactions that before would not be conducted), so there is an increment of another source of income which compensates the reduction of the other. Even though this argument states that, in a worldwide perspective, the public income generated by taxing P2P OM platforms can outbalance the reduction caused by VAT, it is true that in a country-based perspective this does not necessarily apply, because P2P OM platforms usually tribute in tax heavens. Therefore, regulation should focus on designing an international agreement to avoid the existence of tax heavens, rather than focusing on dwindling the development of these platforms within a country.

Apart from that, it is also arguable that these platforms reduce consumer security. In particular, and as we have explained, in P2P OM the customer enjoys a higher amount of information about the transaction. For instance, the opinions of other users endow them with the relevant information to determine whether he/she should conduct the transaction. Moreover, in P2P OM users are more secure about the price of the good or service that they are consuming, since it is usually set before the transaction is conducted and any sunk costs are supported. In the case of city rides this is a clear example because the price is shown upfront and does not modify during the ride, but it also holds for accommodation services. For instance, in traditional markets, after the customer has visited the place usually emerge other fees that increase the transaction price and, since the customer has supported the sunk cost, in terms of time spent visiting the apartment, then he/she is prone to pay the overprice, while through a P2P OM the price is set before incurring in any sunk cost, so it lowers the seller's power and allows the buyer to take a more accurate decision. Even though it is true that scams can be instituted in these platforms, there is usually a quick mechanism to fill out

complains on any transaction and, since all of the transactions are registered in a database, not like in traditional markets, it is easy to deliver a rapid solution based on facts.

Of course, the reasons presented above that defend the idea that the negative impact of P2P OM is small can be debated with counterarguments, but the problem is that, both the ones that were stated above and these counterarguments would be based on some rationale and qualitative opinions, rather than data and real facts. Therefore, as stated before, the first goal that should be set would be to collect more data about these platforms in order to establish a fact-based debate that would lead the regulation needs.

As a conclusion of this subsection, we encourage the reader to be skeptical and to doubt any explanation that is not sustained with data and real facts. With that, we expect the public opinion to endow decision-makers with the possibility to regulate the P2P OM sectors without biased ideas and, as a consequence, to ensure that they will be able to constitute an efficient and fair regulation about P2P OM that will let our society benefit from all the positive opportunities that P2P OM offer.

3.2 Explaining their potential as a more efficient framework for transactions

At the beginning of this section we explained the great benefits that P2P OM have which are, mainly, that since they endow users with lots of information, it is easier to conduct transactions at an efficient price and, as a consequence, the welfare increases. According to what has been exposed at the end of the previous subsection, in order to ensure this positive impact, quantitative studies must be carried out and, in fact, both (Oram, 2001) and (Fraiberger & Sundararajan, 2017) support the idea, using data, that P2P OM are more efficient platforms to conduct transactions than traditional markets. However, we noticed that the whole potential of P2P OM has not been fully exploited yet and, through Data Science, we can further enhance the efficiency of these platforms, especially in terms of decreasing uncertainty when ascertaining the market price.

In particular, even though an extremely large number of transactions are carried out thanks to the easiness to retrieve information that these platforms offer compared to traditional markets, the reduction of information costs, as explained in subsection 2.2 (*From the data deluge to the useful information: The rise of Data Science*) has encountered a serious bottleneck: The users are flooded with lots of data and, because of the size of it and the necessarily amount of time required to analyze it, they tend to ignore it and focus only on a ridiculously small subset.

In terms of ascertaining the market price, the user, even though in P2P OM he/she has access to almost all products offered by the competence, will only fetch the price of some competitors, just to extract an idea about the market price. For instance, a user that is willing to publish a spare room of his/her apartment in Airbnb, will first search the price of three, four or five rooms in the same area and, afterwards, decide the price of his/her room according to his/her expectations. Even though this decision is not the optimal in terms of information, because more could be extracted if the user scanned all the market, it is the optimal according to both the costs of searching for this information (measured by the opportunity cost of the time invested on it) and the expected profit. The main consequence of this behavior is that usually the price is biased and, since the majority of users act the same way, the consequence is a high variability of prices and, therefore, a suboptimal market equilibrium is found. This equilibrium is characterized by the fact that potential transactions are not conducted because of many reasons such as that there is uncertainty about whether the price is fair or not, there are idle products that never find a buyer (because the price does not meet the demand preferences) or that there are less offerors (it is difficult to fix a correct price to sell the good, so they are discouraged to participate).

What we are saying with this is that, even though P2P OM increased the efficiency of the market equilibrium when compared to traditional marketplaces, there is room for improvement within these P2P OM platforms or, in other words, they have potential to further reduce the uncertainty when ascertaining the market price. In particular, these firms maintain a database with all the executed transactions and the key idea is that it could be used to summarize the market price of the product. As explained before, their potential is in the secondary data generated.

Therefore, if the P2P OM platform summarized the market price for the user, then he/she would decide with almost complete information without having to bear any cost of retrieving it and, as a consequence, the price would be closer to the optimal which translates into both benefits for the user (less time to assess the market price and an increment of the expected profit from the good because the price meets the demand needs) and for the market, because there is a reduction of the price variability and more transactions are conducted.

Moreover, it is crucial to understand that the majority of P2P OM users are individuals that use these platforms as a secondary source of income and, since it is not their main activity, their opportunity cost of ascertaining the market price is high and, as a consequence, the potential gains of assessing the market price for them are even larger than in a traditional market. With this, we state, for example, that offering a summary of the market price to a real estate agency has a smaller impact than delivering it to a user of a P2P OM, because the

second one is not a professional and bears a higher level of uncertainty, so his/her decision is further from the optimal.

In fact, we are proposing for P2P OM what firms like TripAdvisor or Kayak did on the secondary data generated by hotel webpages: Analyze all the available information to further reduce the user uncertainty and enhance the market performance. Moreover, we state that this practice not only has positive outcomes for the users and society, but also for the P2P OM platform, so there are incentives for them to implement it.

In particular, the main incentive is that they will increase their profits, both because they will be able to monetize a new service, the price summary, and because, since it will be easier for the user to publish a product as he/she does not have to spend so much time ascertaining the price, more users will enter the market so the P2P OM will benefit from more transactions.

The conclusion of this section is that migrating from traditional markets to P2P OM has reduced the uncertainty of ascertaining the market price, because in a P2P OM the seller can easily check the prices of the competition. However, the comparative advantage of P2P OM has not been fully exploited, because the user usually employs only a small subset of the competition price, so they assess a biased estimate of the market price. Therefore, if the P2P OM made available a price recommendation, based on all the transactions conducted on the market, the user would establish the price of his/her product basing it on the behavior of the whole market, so the decided price would be close to the optimal, which is the one that maximizes the expected profit from the product (the perfect balance between price and probability of being sold) and that fits the demand preferences which translates into a higher market welfare.

4. Pricing model as a tool to ascertain the market price

4.1 Rationale of a pricing model in the peer-to-peer online marketplace setting

As we explained in the previous section, implementing a price recommendation that summarizes all the competition price would yield a higher market efficiency and, also, the market enlargement that it would cause would benefit the firm, so there are incentives to implement it. In fact, the leading P2P OM platforms offer a price recommendation to their users, but they are rather descriptive, and in this project, we propose a sensible model able to capture not only the straightforward information that data about transactions offer, but also the inherent relationship between transaction features and, at the same time, allowing

space for the intrinsic variability that exists around human decisions, which endows us with the possibility to better encapsulate the market behavior.

In particular, in our pricing model, we assume that there are mainly two causes of variability within the price that is set by sellers in P2P OM and they relate to the signal and the noise. In fact, what we are saying is that, using some information about the product features and the transaction setting, we will be able to predict a part of the price, which is the signal. However, at the same time, we will not be able to perfectly predict the price of a product, because there are other reasons than the features of the product that drive the decision of the seller and, since we are not able to measure them, all these reasons will be accounted as the noise.

This rationale of dividing the variability into signal and noise is a general approach for any statistical model and, in order to integrate this rationale into a particular analysis, like the product price as we are conducting in this project, one needs to fully understand the different reasons that feed the two sources of variability. In order to accomplish this purpose, we need to delve into economic theory, to extract main ideas about what is influencing the price of a similar product to be different.

The first reason that explains why a product that serves the same need does not have a unique market price is because there is product differentiation, which means that even though the products fulfill a similar need, they do not do it the same way, so the experience of consuming the product is not exactly the same for the customer and, as a consequence, the demand is willing to pay different prices for very similar products. In order to assess the set of features of a product in our model we will use all the measurable characteristics that are able to modify the customer utility (experience), and with that, we aim to integrate this product differentiation as the signal part of our model. In terms of a statistical model, all these features will be predictors of our model and the price will be the response variable. For instance, in Airbnb there are apartments with and without air conditioning and, even though they serve the same need, which is the accommodation one, one offers the customer a better experience than the other and, even though those apartments will compete with each other, is reasonable for the more equipped apartment to have a higher price. Since it is both reasonable and measurable, we will input this information as our predictors in our model, so their will constitute part of the signal. The same example could be used for Uber (for instance if the seats are comfortable or not) or Blablacar (if the car is small or not), in RV Share (if the RV has all kitchen equipment) or in Liquid Space (the square meters of the space) or, in fact, in any P2P OM. Like we said before, this is why we focus in pricing models for P2P OM, because there are lots of these platforms and they have the required data to build the model.

In fact, another potential of P2P OM data is that they track many features of the transactions conducted through their platform and, as a consequence, allow us to measure some concepts of product differentiation that in traditional marketplaces is not possible to measure. For instance, the reputation of a seller is easily assessed in P2P OM, through the opinions that the community has published about the seller. Moreover, the exceptional possibilities that Natural Language Processing is deploying (Lewis & Jones., 1996) will cause these opinions to have an extraordinary added value to analyze P2P OM.

Product differentiation is one characteristic that is repeated in almost every market in our society and, as Rosen (Rosen, 1974) states, it is possible for an equilibrium to exists in those markets. Therefore, as we know that product differentiation exists, we use it as an input for our model, because only with it we will be able to simulate the market behavior through the model and, as a consequence, we will have the possibility to yield accurate predictions.

A second reason that explains why the price of a similar product is not unique is because usually economic agents do not operate in single markets. Instead, they take decisions having in mind that there are several markets, which are strongly related between them. In particular, when two products are too different (but they satisfy a similar need) they are part of different markets, but they are still connected because they are substitutive goods. For instance, it is not the same product an Airbnb apartment (in which the customer is alone in the apartment) that an Airbnb private room (where the customer shares the flat) but, even though they are in different markets, the two products are substitutive of each other. In order to integrate this concept in a pricing model, one needs to analyze all products jointly, but have some predictors that distinguish the products that are from different markets. In fact, thanks to the amount of information that P2P OM offer, we are able to input those market-tracker variables in our model. Therefore, this source of variability, which is the effect that several similar markets have on each other, is reflected as a signal part of our model, since it is captured by some predictors.

A third reason that affects the product price and, as a consequence, is a source of variability is the transaction setting. This refers to how the environment has conditioned the price of the transaction and, even though a great deal of this environment effect is fuzzy and unmeasurable, P2P OM offer the possibility to get a glance at it. In particular, P2P OM register the exact time and position of both the buyer and the seller at the moment when the transaction was conducted, so it can be used as an input for the model, as another predictor. However, even though that we present this as a possibility for our model, we do not have access to all the transactions (only the P2P OM firm does) so in the example that is attached at the last part of this thesis does not capture these time and location variables. However,

there are other setting variables, like the seller experience, that have an effect on the transaction price and that are also captured by P2P OM, because the platform is able to know for how long the user has been a seller and also how many transactions has he/she performed. Therefore, those setting variable that we are able to measure will enter the model as signal, through some predictors, and those others that we have not been able yet to measure, will enter as the noise part of the model.

Finally, the last source of variability, which is obviously unmeasurable, is based on the seller expectations. As we explained in subsection 1.3 (*A specific case: Uncertainty during the ascertainment of the market price*), the final price of the product is a reasonable decision carried out by economic agents and, since they face lots of uncertainty about the market price, they set a price according to their intuition, to their expectations, the animal spirit as Keynes stated. A pricing model has this condition in mind and, in order to track it, it is attached to the noise part of the model. Remember that the noise part is telling that even though two products have the exact same characteristics they do not have to present the same final price, because there are lots of reasons that are not measured that drive the decision of the final price. What we are saying, is that using all the transaction data that P2P OM offer, even though we will not be able to measure those "noise" reasons, we will be able to assess their overall impact, through the noise part of our model. In fact, this is where our pricing model as a price recommender overcome the current descriptive model used by some P2P OM, because it is able to estimate how unsure it is about the prediction, because it has in mind the unpredictability of human behavior.

After revising the different sources of variability of the market price one can conclude that, a pricing model, is a statistical model that uses the measurable characteristics of the product and transaction in order to yield an accurate prediction about the price and, moreover, it harnesses through the noise all those reasons that explain the product price but are not measurable, which are, mainly, the seller expectations. In simpler words, we are connecting some X (features of the transaction) to a Y (product price) and, at the same time, we are allowing some room for the idea that not everyone decides with the same arguments.

4.2 Implementation as a price recommender: Requirements and considerations

Now that we have described the rationale of a pricing model and the different parts that compose it, we need to specify how they should be implemented. The basic idea is that the model would recommend an interval of prices for the user, so that he/she immediately is aware of the market price for a product or service like the one that he/she is offering. As we have long discussed during this thesis, this would reduce the amount of time retrieving

information, which translates into smaller costs of information (in terms of opportunity costs) and, as a consequence, and increment of market efficiency (more transactions, more users, reduction of the number of products that do not meet the demand, reduction of price variability, etc.).

In other words, the main added value of the price recommender is divided into two pillars. The first one is that the predicted price is able to harness the information of all possible competitors for the user product, and not only the information from the closest competitors, so there is a reduction of the user bias when deciding the final price. On the other hand, the second main pillar is that the prediction is offered through an interval, which endows the user with more information to enhance the decision that he/she will take, because a punctual prediction in many cases is not useful because it does not say anything about how sure it is of the prediction.

For instance, imagine that Uber has a price recommender for all the drivers or, in terms used until now, the sellers of a service in this case. In particular, if the price recommender told a user that the price per kilometer set by all the users that have similar characteristics (so, the competition) is 0.3 €, then he would not know what would happen if he/she increased the price to 0.7€, because he/she is not able to evaluate if it is too much or not for the demand preferences. However, if the price recommender stated that, for characteristics like the ones that the user service presents (car, amenities…), the price is between 0.2€ and 0.5€ then, surely, the user would know that pricing it at 0.7€ would likely place him/her out of the market, so he decides a lower price. As a consequence, more transactions are performed, because there are not idle products on the market waiting for the demand to increase, because with a lower price they are able to meet the demand expectations.

Therefore, the main benefit is that they allow users to take accurate decisions about the price of the product, without the need of sweeping the market to ascertain the price. As a consequence that all users take better decisions (because there has been an increment of information), the whole market performs better and the society welfare increases.

In order to build a pricing model that will feed the price recommender, there are several requirements that must be fulfilled.

The first main condition is that the pricing model needs a database with all the conducted transactions and, for each of this transaction, as much information as possible should be retained. In fact, the more information about the product and the transaction, the more important will be the signal part of our model and, as a consequence, both the punctual

prediction and the interval will be better. The punctual prediction will be enhanced because it will be more accurate and, on the other hand, the interval of prediction will be narrower, because the noise will only reflect the animal spirits part (since all the other reasons of variability will be captured through the signal).

This first requirement is met by almost all P2P OM firms, since all of them track the transactions conducted through their platforms and, as a consequence, are endowed with a database that stores the potential predictors and the price. However, the amount of data (or predictors) collected will vary among P2P OM firms, so the outcome of the pricing model will depend on the platform's effort to build an exhaustive database. It is relevant to state that building this transaction-level in traditional markets is too costly (manually tracking the features, adding up the transactions of all sellers, overcoming errors and purposely biased information…) while in P2P OM firms it is automatically generated, without extra costs. There are still two more considerations that should be taken into account when dealing with this database generated by P2P OM firms.

The first one is that not all products and services posted by sellers should input the pricing model, instead, only the goods that caused a transaction should be accounted, because we want to extract the market behavior (both supply and demand), and not the supply attitude. In other words, if we were to track all posted products then our model would tend to overestimate the market price due to those goods that are overpriced and that never find a buyer and, as a consequence, the added value of the price recommender would diminish.

The second consideration is that, even though the aim is to analyze all conducted transactions within the platform, this would not be enough to exhaustively determine the market behavior because we are not considering the transactions performed in competitors P2P OM that serve a similar need and, also, we are leaving out the transactions performed in traditional markets. Therefore, the P2P OM firm that implements the pricing model will be working only with a sample of all the transactions that are performed in a market and, as a consequence, we need to build a statistical model able to perform inference about the market price. For instance, if Airbnb were to develop a pricing model as the one that we propose in this thesis, it would not take into account the transactions conducted by other P2P OM platforms like Badi or Home Away nor the transactions instituted by traditional real estate agents. However, we assume that transactions developed in Airbnb itself take into account the prices of these other platforms (if not, they would not have demand) and, as a consequence, are a valid sample to extract conclusions on the overall market price and, therefore, perform well as a price recommender.

The second requirement that must be fulfilled in order to enable the pricing model to yield accurate price recommendations is that the product sold within the P2P OM platform must be as much homogeneous as possible. As explained in the previous subsection (*Rationale of a pricing model in the peer-to-peer online marketplace setting*), product differentiation is common in our markets and, through the product and transactions features, we are able to harness the effect of this differentiation on the final price. However, the more heterogeneous the products are, the more difficult will it be to harness this differentiation and, as a consequence, it could cause the pricing model to yield recommendation with small added value. For instance, in platforms like Airbnb, Uber or Blablacar the products within each platform are similar: Accommodation, city rides and long rides. However, in TaskRabbits, since the offered services are manifold and of different nature, building a pricing model would be more difficult. Moreover, this complexity becomes overwhelming in P2P OM where different markets, without almost no relationship between them, are in the same platform, such as Classify Ads firms like eBay. In this case, the different transactions conducted may not be competing between them and, as a consequence, the pricing model would have it harder to assess the market behavior.

4.3 Desirability of a pricing model to ascertain the market price: Choosing a model

In order for the recommender to be useful two main goals should be pursued by the underlying the model and, the better those objectives are fulfilled, the better recommendation will yield the pricing model. As a reminder, the better recommendation, the more enhanced will be the decision about the price of the user and, as a consequence, the higher will be the increment on social welfare.

The first goal is that a pricing model should deliver exact punctual predictions of the price, given some features about the product that the user wants to publish. However, we acknowledge that it is not possible, because as we explained, there are untraceable sources of variability (like human expectations). Therefore, the best model will be the one that makes the most of the predictors and, as a consequence, delivers a punctual prediction with the smaller error. The smaller the error, the closer the recommended price will be to the market price and, as a consequence, the decision of the user about the price will be closer to the optimal.

It is relevant to state that the model needs to yield a small out-of-sample error, because it will be recommending a price for a user that wants to publish a product (it will not be in the training dataset), so we need the model that better generalizes the market behavior.

The second goal that must be fulfilled refers to how much the user can rely on the prediction of the market price offered by the recommender. In fact, this relates to the idea that exists some unaccounted sources of variability: there is a market price but there is some variability around it, due to the fixed prices come from human decisions which are based on expectations. Therefore, the punctual prediction must be accompanied by an interval-based prediction because only with it the user will be able to take a better decision than before. This interval must be both realistic, in the sense that it captures the market variability and, at the same time, narrow, because if it is too wide then it becomes useless for the user.

Since a pricing model, as the one that we presented during the subsection 4.1 (*Rationale of a pricing model in the peer-to-peer online marketplace setting*), takes into account both the signal part and the noise, it is able to deliver substantiated intervals and, as a consequence, it is the natural approach for a problem like the ascertainment of the market price. Nevertheless, the types of pricing model are infinite, since they all depend on how the predictors of the model relate with each other and with the response variable which, in this case, is the price.

A straightforward solution would be to use a linear function of the predictors in order to assess the signal part and capture the noise through a Gaussian random variable. As the reader knows, this would be the linear model used for regression. Moreover, and in order to enhance the capabilities of this model, some interactions between explanatory variables and polynomial functions of the predictors can be included in the signal part.
Even though this model has lots of virtues, like the fact that the effect of each predictor can be easily assessed, it also has downsides for the goal that we are considering in this thesis, which is building a price recommender. As we said before, the two main objectives that must be fulfilled by a pricing model is that the punctual prediction must have a small error and, also, that the prediction is endowed with intervals. Even though the second goal is fulfilled by the linear model, the punctual prediction yielded by it is often not as good as the one offered by spearheading machine learning techniques and, because of that, in this project we aim to develop a model that is able to enhance the prediction offered by the linear model.

However, those machine learning algorithms are, as the name suggest, only algorithms which means that they aim to minimize some objective function but, in fact, they are not built under the setting established by statisticians: the signal and the noise. Therefore, even though they deliver, in many examples, better punctual prediction than a linear model, they would be useless for the purpose of building a model pricing, because they are not able to offer substantiated intervals for the prediction.

With this idea that the linear model is not the best tool for the first goal (accurate punctual prediction) and that machine learning algorithms fail to fulfill the second goal (offer intervals), in this project we propose a more suitable tool for a price recommendation which is, in fact, a combination of both the statistical model and the new machine learning techniques, able to overcome the downsides of both approaches.

As the reader knows, machine learning offers a wide range of algorithms and in this project, we will focus only on Artificial Neural Networks because is the one that has captured the most attention due to the flexibility that it offers and the fact that every day new approaches and techniques derived from Artificial Neural Networks are being developed and used, so it is the algorithm with higher potential. Moreover, we believe that Artificial Neural Networks (ANN from now on) are the best example to show that, actually, the statistical model and machine learning can be combined.

In fact, we believe that the two concepts (i.e. statistical model and ANNs) are naturally connected in a straightforward way, even though the current literature does not offer this perspective. Therefore, during the next section of this project we aim to convince the reader that the only thing that is needed to connect ANNs and the statistical model is a little bit of abstraction. However, in order to envision and materialize the bridge between the two concepts it is fundamental to work within a more abstract and flexible framework to perform statistical analysis: The Bayesian framework.

In the next section of this project we will explain this connection and we will also materialize it in an example. As an introduction, we envisioned a non-linear statistical model where the signal part is the result of an ANN and, apart from it, there is a random noise that captures the variability that could not be explained by the predictors. In this model, the likelihood becomes complex and extracting from it a true maximum likelihood estimator is too complicated, while using the Bayesian framework its treatment is possible, even though it requires some further abstraction that will discussed during the next section. With this model, called Bayesian Neural Network (BNN), we aim to enhance the punctual prediction with respect to the linear model and, moreover, endow our prediction with intervals.

Chapter II: Feed-Forward Bayesian Neural Networks

The purpose of this chapter is to present Feed-Forward Bayesian Neural Networks (BNN) and, in particular, propose a specific Bayesian Neural Network and explain the methodology that we devised in order to use it to deliver interval-based predictions.

The first section aims to establish the basis of a BNN. It is divided in four subsections in which we review Bayesian statistics, conventional Artificial Neural Networks, an explanation about the reasons why BNN can be a superior model and, finally, an overview about the main research conducted about BNNs by previous authors.

In the second section we delve into the specific BNN that we propose, dividing our explanation into three sensible subsections. While in the first subsection we formulate, from a theoretical point of view, the proposed BNN, in the second subsection we explain the devised methodology to implement the proposed BNN in any applied case. In this second subsection we start by analyzing the posterior distribution of a BNN given an architecture and, moreover, how to use MCMC to sample from it. Afterwards, we analyze how to select a suitable architecture using DoE and we summarize the whole methodology in a very specific pipeline with all the steps that must be taken. Finally, in the third subsection we focus on possible extensions for the BNN in the case that the model is not validated and also, we propose two techniques (one based on contributions from previous authors and another completely devised in this thesis) that help to understand what is driving the prediction of a BNN.

5. Foundations of Bayesian Neural Networks

5.1 Overview of the Bayesian framework and its flexibility

5.1.1 Theoretical approach of the Bayesian framework

In order to be able to understand the functioning of a Bayesian Neural Network it is required to have some notions about the Bayesian approach, which is what we are going to explain during this subsection. However, our goal is to provide a brief introduction, so for an in-depth explanation of the Bayesian framework please refer to (Bolstad & Curran, 2016) and (Kruschke, 2014), which are the books that were used for this introduction.

In parametric statistics, the foremost concept is the statistical model in which we assume that the variability of what we are measuring (Y) behaves according to a probability distribution,

which depends on some parameters (θ) that live in a parameter space (Θ). According to the approach explained during the previous section, the parameters would be related to the signal and the variability that casts the probability distribution around that signal would be the noise.

For each θ contained in Θ the statistical model offers a specific probability distribution and the goal is to find the θ that better explains the behavior of what we are measuring, using all available information. However, since there is intrinsic variability of what we are measuring (Y), we will only be able to have an estimate of the best θ and, as a consequence, we will have uncertainty about which is the best θ.

One source of information to estimate the best θ is, obviously, what we have observed (y). In fact, the concept that captures the relationship between θ and what we have measured (y) is the likelihood, which is the jointly probability distribution (according to the selected statistical model) of Y evaluated at the values that we have observed: $P(Y = y|\theta)$. The likelihood is a function from Θ to R^+ in which those θ that are more likely to have generated the observed data are those with a higher value in R^+ and, in fact, since the only relevant part is the ordering of the parameters, it can be scaled to a part of R^+. Since the likelihood orders all Θ, it is often said that the likelihood is what summarizes all the information that the observed data has about the parameters.

However, this is not the unique source of information that we can use to assess the best θ or, in other words, to assess the behavior of what we are measuring. In fact, there is a latent source of information that is influencing our estimate: our prior information or, in other words, our knowledge about what we are measuring. This prior information is crucial in statistics and plays a major role during the creation of the model. For instance, in Design of Experiments, one must decide which variables he/she thinks that are influencing a response variable and, as a consequence, is using some knowledge before even collecting the data. Moreover, in the statistical model that we explained we input prior knowledge through the likelihood that we establish, i.e. through the probability distribution that we assume for our statistical model.

Therefore, since it is obvious that there is prior information, in the Bayesian setting we do not ignore it and, in fact, we decide how much prior information we want to pass to our model. In other words, in the Bayesian setting we are empowered to take the decision about how much should weight our previous knowledge and we harness it through a probability distribution on Θ called the "prior probability distribution" $(\Pi(\theta))$.

As a direct consequence of assessing a probability distribution over Θ, we are considering that θ is a random variable, instead of a fixed value. The underlying reason that explains this abstraction is the fact that, even though the true parameter can be one unique value, from our perspective we will never be able to be completely certain about the parameter, because we are only able to observe y, the realization of a random variable Y which is assumed to follow a probability distribution with a particular θ^*. Due to the randomness of Y it is credible that the realization observed y comes from probability distributions with a θ different than θ^* and, since we cannot observe θ^* we will have to believe that all θ capable of having generated y are the possible true parameter, even though the true parameter is only θ^*. Therefore, this uncertainty about which value of θ has generated y causes that, for us, θ is a random variable.

In order to convey this idea, imagine walking along a park and hearing a squawk but, since the park is very leafy you are not able to see which kind of bird has emitted it. In this case, the true bird species that has emitted the squawk is θ^* and let's name it species A. However, the squawk that you have heard is y and, of course, since every bird of the species A is different, not all squawks emitted by individuals of species A are equal, so the variable Y is random depending on which individual of the species A emits it, because it can have different duration and pitch. Moreover, since there is this variability in pitch and duration of the squawk it can be possible that for some individuals of the species A, its squawk is similar to the one emitted by individuals of the species B. Having this in mind, after hearing the squawk y, you will have to choose whether if the individual that emitted it was from the species A or B and, for you, the true species of the individual will be a random variable, since you'll assess some probability of being from each species, depending on the pitch and duration of the squawk. Therefore, for the subject that has to take the decision, the true species of the bird is a random variable because, even though the squawk y was generated by the species A (θ^*), the subject only is able to hear the squawk y and he/she is not able to see θ, the species.

Another reason that explains why θ is a random variable relates to the fact that we are using a model to summarize what we are observing, even though it is obvious that reality does not behave as the model states. In other words, and quoting George Box: "All models are wrong, but some are useful" (Box & Draper, 1987). Therefore, θ is a random variable because aims to capture some artificial concept that, in reality, does not exist so it cannot be fixed to a value. For a more in-depth analysis of the foundations of assuming that θ are random variables, please refer either to (Cox, 1946) or Jaynes (Jaynes, 1986).

As stated above, our goal is to find the θ of our statistical model that better reflects the behavior of what we are measuring and, in order to do that, we will use both our prior

knowledge and the information that the observed data has provided us. The first source of knowledge is harnessed through the prior distribution $\Pi(\theta)$, while the second is controlled by the likelihood $P(Y = y|\theta)$ and, both of the functions order Θ in the sense that θ with higher values of both functions are more probable to correctly explain the behavior of what we are measuring (Y).

Since both θ and Y are random variables, we use Bayes' theorem to combine the two mentioned sources of information and summarize all the knowledge that we have about θ. Of course, we will not obtain a value for θ, instead, we will obtain a superior concept: a probability distribution on Θ that will reflect all that we know about θ, uncertainty included. This probability distribution, $\Pi(\theta|y)$ is called the posterior distribution and is obtained through

$$\Pi(\theta|y) = \frac{P(Y = y|\theta)\Pi(\theta)}{P(Y = y)} = \frac{P(Y = y|\theta)\Pi(\theta)}{\int_{\Theta} P(Y = y|\theta)\Pi(\theta)d\theta} \propto P(Y = y|\theta)\Pi(\theta). \qquad (2.1)$$

As it can be seen from the formula above, the posterior distribution is just a compromise between what we knew and what we have learned, and it reflects all the knowledge, with the inherent uncertainty, that we have about θ or, in other words, about the specific probability distribution that we assume that has generated the data that we have observed.

This whole framework endows us with an enormous flexibility to envision almost any model and, moreover, it allows us to take decisions in a more suitable rationale than the one offered by the frequentist approach.

In the frequentist approach, whenever we carry out a hypothesis test, what we decide is whether if the null hypothesis is refutable or not according to what we have observed and an assumed risk. As a consequence, in the frequentist setting one does not decide whether if the hypothesis selected is correct or not and, in fact, only the null hypothesis is actually tested. On the other hand, in the Bayesian framework we associate each hypothesis to a part of Θ and, since we are able to represent all our knowledge on Θ, through the posterior distribution, we can assess a probability to each of the different hypothesis, because the only thing required is to integrate the posterior distribution over the subspace of Θ corresponding to each hypothesis. Therefore, in the Bayesian setting we can test multiple hypothesis simultaneously and each of them will have an attached probability, which represents the probability of being the true hypothesis conditioned to all of our knowledge.

As a consequence, in the Bayesian setting instead of stating that one hypothesis is refutable or not, we state that all hypotheses are potentially correct with a certain probability and, like in the frequentist setting if we select one hypothesis we can incur an error. In fact, decision-making through hypothesis test in the Bayesian setting is easier to understand thanks to the fact that it resembles decision-making in our daily life, in which we have to decide between different options and, even though we know that all of them could be the best, we select the one that has a higher probability, according to what we know, of being the best.

Another further abstraction and, as a consequence, source of flexibility of the Bayesian framework is the fact that Bayesians do not have to assume that only one model, the chosen one, is the correct. In fact, Bayesian can take into account several models (each model refers to a different likelihood) and combine them to extract accurate predictions. In order to envision this idea, one has to realize that whenever we obtain the posterior distribution, we obtain it according to a particular model, so the equation above should be

$$\Pi(\theta|y, M_k) = \frac{P(Y = y|\theta, M_k)\Pi(\theta|M_k)}{\int_\Theta P(Y = y|\theta, M_k)\Pi(\theta|M_k)d\theta} = \frac{P(Y = y|\theta, M_k)\Pi(\theta|M_k)}{\mathrm{P}(Y|M_k)}. \qquad (2.2)$$

Therefore, we can take into account several models $(M_1, \dots M_K)$ and, for each of them we would obtain $\Pi(\theta|y, M_k)$ which must, then, be combined in order to obtain $\Pi(\theta|y)$. However, not all models have the same relevance, because some of them are more likely to capture the behavior of what we have observed and, as a consequence, we have to hierarchize them according to what we have observed. In order to do that we compute the posterior model probability for each model:

$$\mathrm{P}(\mathrm{M_k}|Y) = \frac{P(Y|M_k)P(M_k)}{\sum_{k=1}^{K} P(Y|M_k)P(M_k)}. \qquad (2.3)$$

After obtaining these probabilities, one could decide to take the model with the highest posterior probability, which would lead to work with only one model (like in the frequentist case) or, otherwise, one could decide to combine all the models weighting them according to their posterior probability. Referring again to George Box and its statement “All models are wrong, but some are useful”, in the Bayesian framework we can actually work with **some** models, instead of only one so maybe we are more capable to capture the behavior of what we have observed. All this theory around considering several models simultaneously is called Bayesian Model Averaging and, for a more in-depth analysis please refer to (Fragoso, Bertoli, & Louzada, 2018).

5.1.2 Implementation of the Bayesian approach

In the Bayesian framework, the object to perform inference is the posterior distribution $\left(\Pi(\theta|y)\right)$ and, in order to obtain it an integral over Θ must be conducted, even though it is true that in some simplistic examples it is possible to obtain the posterior distribution without carrying out any integration. However, in almost every applied study it is required to conduct the mentioned integral over Θ but, since θ is often multidimensional extracting and analytical solution is intractable. Therefore, one should refer to numerical methods in order to solve the integral, but it is also discarded due to the fact that the error accumulated would lead to incorrect results and, also, since Θ can have lots of dimensions, the time to obtain those results could be prohibitive.

As a consequence, different approaches to avoid those intractable integrals have been developed and the two salient ones are Markov Chain Monte Carlo (MCMC) and Variational Inference (VI). In this project we will sketch out an intuition about the two topics, but we will not delve into them because it would be too extensive for the purpose of this project.

The main idea behind MCMC methods is that, through an iterative simulation of the different parameters we simulate values from a Markov Chain which has, as stationary state, the posterior distribution that we need to conduct inference in our model. Therefore, if we take several samples from the stationary state of this Markov Chain, we are endowed, then, with random samples of the posterior probability distribution and, as a consequence, we can describe (variability, shape, location, percentiles…) that posterior probability distribution.

In VI the approach is different, since we try to find a probability distribution that can approximate the posterior distribution. In order to do that, we define a family of distributions able to be an approximation of the posterior distribution and, through optimization, we obtain the member of the family that is closer to the posterior distribution, as stated in (Blei, Kucukelbir, & McAuliffe, 2017). As a consequence, we are endowed with the parameters of a probability distribution that approximates the posterior distribution, while in MCMC we had samples of the exact posterior distribution.

Even though MCMC is more general, since we do not assume any kind of distribution to approximate the posterior distribution and because we know that the Markov Chain will eventually reach the stationary state (so we will eventually sample from the posterior), the computation time of MCMC is significantly greater than the one from VI.

Whether MCMC or VI methods are used, the fact is that Bayesian analysis is endowed with powerful tools to approximate the posterior distribution and, as a consequence, inference can be conducted while the intractable integrals are avoided. As it was explained during the theoretical discussion of the Bayesian approach, this framework is more flexible and abstract than the frequentist setting and thanks to the implementation easiness and robustness of these techniques, specially MCMC, this generalization that characterizes the Bayesian framework is also able to be harnessed in real and practical scenarios.

For instance, changing the probability distribution of the response variable in a linear model in the frequentist setting would require to analytically extract the Maximum Likelihood Estimator, and also the Fisher Information matrix to obtain an estimate of the parameters' variability. On the other hand, the only thing required in the Bayesian setting, if MCMC are used, is to specify the new distribution from which simulations (i.e. random draw) at each iteration will be obtained.

Another example, which is the one that we will focus in this project, is that if the likelihood becomes complicated because, for instance, we are using a nonlinear model, analytically obtaining the equation for the Maximum Likelihood Estimator can be intractable and we have to rely on iterative optimization procedures and, as a consequence, work with local optima of the likelihood as our solution. However, as it will be explained during this thesis working with MCMC allows us to deal in a more natural way with multimodal likelihood, which places the Bayesian setting as a more suitable framework to deal with complicated models.

The main conclusion is that not only exist theoretical foundations that explain the capabilities of the Bayesian framework, but also there are tools that allow its implementation, which means that the flexibility of working with almost any kind of model is possible in the Bayesian framework. In fact, MCMC field is in constant development and new algorithms appear at an extraordinary fast pace, which means that the implementation capabilities of the Bayesian framework are being constantly enhanced.

5.2 Rationale of feed-forward Artificial Neural Networks

In the previous subsection, we explained the Bayesian framework, which is one of the main pillars of the pricing model that we propose in this project: The Bayesian Neural Network. This subsection tackles the second pillar of our pricing model and, therefore, its goal is to describe how an Artificial Neural Network (ANN) predicts a response variable according to some predictors having in mind that, in our case, the response variable will be the price of a transaction and the predictors all the information about the product and transaction.

5.2.1 Origins of feed-forward Artificial Neural Networks

Before describing how an ANN works, it is relevant to explain the origins of ANNs, so the reader can envision their purpose. During the 1940s several researchers as McCulloch (McCulloch & Pitts, 1943) and Hebb (Hebb, 1949) started to investigate a way to model and structure the functioning of human neurons and, due to the development of computers during the 1950s, many research projects aimed to embody the human neurons functioning in a machine. In fact, one of the most successful application was the Perceptron by Rosenblatt (Rosenblatt, 1958), which yielded an algorithm able to distinguish examples of two different classes. Due to this early success, the hype around Artificial Neural Network increased significantly and several researchers envisioned that Artificial Intelligence implementation was imminent, so many resources were allocated to fund projects about Artificial Neural Networks. However, due to the fact that the Perceptron was not generalizable, the fact that other algorithms did not present the expected outcome and the technological constraints, caused the development of Artificial Neural Networks to freeze for almost two decades. Even though this was the first approach to ANN, the most important development of ANN started during the late 1980s and 1990s, when the interest in ANN was reborn.

During the late 1980s and the 1990s, researchers prioritized developing useful techniques rather than strictly emulating the human neurons functioning and, as a result, several "neuron-based" approaches emerged, being the feed-forward Artificial Neural Network with a hidden layer the most important. Even though some authors envisioned this type of ANN, its actual development was enabled by the influential concept of back-propagation presented by Rumelhart and Williams (Rumelhart & Williams, 1986).

The reader should have in mind that feed-forward ANN or Multi-Layer Perceptron may have different names in the literature and that we used feed-forward ANN because is the one offered by Bishop in (Bishop, 1995), which is considered the uppermost manual to understand the rationale of Artificial Neural Networks. In the following part of this subsection we will explain the concept and capabilities of a feed-forward ANN so, for a more in-depth review of ANN history please refer to (Anderson & McNeill, 1992).

5.2.2 Explaining their rationale and how they parameters are obtained

From now on, we will refer to feed-forward ANN only with ANN because it will be a common concept used during the whole project and it will simplify the explanation. An ANN uses several observations to estimate some parameters that, combined with the predictors (or input), yields an accurate prediction for the response variable. The added value of an ANN is,

mainly, that is able to capture nonlinearities with flexibility and, therefore, has the potential to better approximate the function that links the response variable and the predictors which means that, in some situations, out of sample prediction performance in ANN is greater than in other methods like a linear regression.

The basic concept behind an ANN is that, in order to capture nonlinearities, an ANN calculates a set of latent variables, which are a nonlinear function of the predictors and, afterwards, computes a linear combination of those latent variables in order to predict the response variable. Using the graphical representation of an ANN with one hidden layer in Figure 2.1 we will explain how an ANN works.

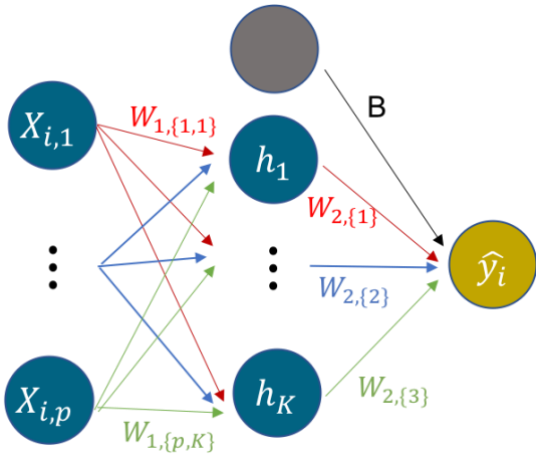


Figure 2.1: Graphical representation of an ANN with one hidden layer.

As it can be seen in the graphical representation, an ANN is structured in three parts: the input layer, the hidden layers and the output layer.

Input layer:

The starting point is characterized by the fact that we have N observations, so we have a vector Y of dimension N that contains the response value for each observation and, associated to each observation i of the vector Y we have a vector Z_i that stores the values of the predictors of that particular observation. Therefore, we have a matrix Z with N rows and a specific number of columns, each one of them relating to a particular predictor that we are considering. Afterwards, we can convert Z into a model matrix X, which will have: a constant column, a column for each numeric predictor and a column for each dummy variable generated by the categorical variables of Z (we used contrast treatment in this approach), without having in mind quadratic terms nor interactions in this model matrix.

As a result, X will be of dimension Nxp, being p the total number of columns of X. If we take the row i of this matrix X, we can place a node in the input layer with the value that the

observation i takes in each column, so the input layer is, in the graphical representation, just a row of the model matrix. For instance, imagine that we want to predict the price per kilometer of an Uber ride and we work with two predictors: average reputation of the driver and the type of car. Therefore, in this case Z would be (if we represent the observation number 1 and N):

Average_Reputation	Car_type
4.5	SUV
…	…
3.8	Sedan

Table 2.1: Example of Uber dataset used to explain how to fit an ANN.

Afterwards, we would create the model matrix associated to this Z, which would be (with Cart_type_SUV being the baseline category and considering that there are only three types):

Intercept	Average_Reputation	Car_type_Sedan	Car_type_Hatchback
1	4.5	0	0
1	…	…	…
1	3.8	1	0

Table 2.2: Model matrix associated to the dataset in table 2.1.

Therefore, the input layer for the observation N would be:

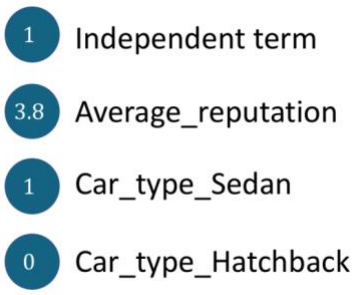


Figure 2.2: Input layer for the example of how to fit an ANN.

Hidden layer:

In the graphical representation it can be seen that each node of the hidden layer, except for the gray one that will have a special consideration, is connected to each input node or, in other words, to all of the matrix model columns. What this connection represents is that each node of the hidden layer comes from a linear combination of the predictors, so each line represented in the graph refers to the weight or coefficient that each input node has in this mentioned linear combination. After obtaining the value of this linear combination a

differentiable nonlinear function f, called activation function, is applied and the value obtained will be the final value that the individual i will have in the node k of the hidden layer.

For simplicity, all nodes in a hidden layer have the same activation function and the usually employed ones are *tanh* and *logistic*, which are also the ones used for the examples of this project.

Since the value of an observation i for a node of the hidden layer is just the image of an activation function on a linear combination of the model matrix, we can parametrize its value with the following equation

$$h_{i,k} = f\left(x_i W_{1,k}\right), \tag{2.4}$$

where x_i is a rowvector of dimension p representing the row of the model matrix associated to the observation i, $W_{1,k}$ is a vector of weights with dimension p and, as a consequence, $h_{i,k}$ is a real scalar representing the value of the observation i in the node k of the hidden layer. If we have more than one node in the hidden layer, since all the nodes have the same activation function, we could compute

$$h_i = f(x_i W_1), \tag{2.5}$$

where W_1 is a matrix of p rows and K columns, being K the number of nodes in the hidden layer. Therefore, h_i is a rowvector of dimension K that stores the value of observation i in each node of the hidden layer. Each node of the hidden layer will represent a latent variable, as it was explained before, and it can be calculated for all the observations. In fact, we can obtain a matrix that collects all h_i by using X instead of x_i in the equation above.

Output layer

Once we have obtained all the latent variables or, in other words, values of the hidden layer, then we compute a linear combination on them. The weights of this linear combination are stored in W_2, which is a matrix of dimension $Kx1$. The number of columns will always be 1 in our case, because we are predicting a continuous variable (price of the transaction) so, in the output layer, there will be only one node, which will refer to the value predicted. Since the value of the output layer is a linear combination of the latent variables (or nodes in the hidden layer) it is useful to specify an independent term, which is represented by the line B (stands for “Bias”) in the graphical representation, which connects the gray node of the hidden layer

with the output layer. Since it is the independent term, the grey node is not connected with the input layer.

Since the node of the output layer is a linear combination on the latent variables, we could obtain the following equation to connect the predicted value of y for the observation i:

$$\hat{y}_i = h_i W_2 + B = f(x_i W_1) W_2 + B. \tag{2.6}$$

As a consequence, we have obtained a function that, in order to predict a response variable y, uses a function of some predictors x and some parameters W_1, W_2, B. Once the functional relationship between y and x is defined, the next step consists on finding those set of parameters which minimize a chosen error function, such as the sum of squared errors, between the observed value y and the predicted value $\hat{y}$.

However, extracting the optimum analytically is impossible, due to the high dependence between parameters or, in other words, the fact that a derivative of the error function with respect a parameter depends on almost all other parameters. As a consequence, in order to obtain an optimum, iterative optimization algorithms are applied and since the error function to minimize is non-convex, we will only be able to extract local optima.

The basic idea is that this implementation of an iterative optimization algorithm is not straightforward and depending on which algorithm is used and how it is applied, one obtains a different type of ANN. However, explaining all the different kinds of ANN is beyond the scope of this subsection, in which we only wanted to explain the rationale of ANN in order to be applied in the Bayesian setting so, for a more in-depth analysis please refer to (Bishop, 1995).

Even though this was the first approach for an ANN, the fact that this technique was prone to overfit the training set caused several new applications to flourish, being regularization the most famous one. Regularization, which is a widely used technique in Machine Learning, consists of including a term in the objective error function which penalizes the complexity of the model (i.e. the number and value of the parameters) in order to find a simplistic set of parameters that avoids overfitting the training set and, as a consequence, is able to yield a better out-of-sample performance when predicting. In fact, this is just a way of implementing regularization but there are different methodologies and, since we only aimed to sketch out the existence of regularization, in order to find out the foundations of its implementation and some discussion about its effects, please refer to (Larsen & Hansen, 1994).

Apart from regularization, other techniques like including a dropout layer, were further developed and, as a consequence the scope of possible ANN is wide, so referring to ANN one does not specifically talk about a particular algorithm but a family of algorithms with the same purpose: yield accurate predictions of a response variables by applying linear combinations and nonlinear functions on the predictors.

In order to explain the rationale of an ANN, we have used an example with only one hidden layer but, of course, an ANN with more than one hidden layer can be implemented. When we allow the ANN to have more than one hidden layer, what we are imposing is a complication on how the latent variables are obtained. According to the rationale explained in this thesis, the latent variables will be the nodes from the last hidden layer or, in other words, the nodes of the layer the precedes the output layer.

In Figure 2.3 there is a graphical representation of an ANN with L hidden layers, each one of them with K_l nodes. Just as we did with the one-hidden-layer ANN, we will extract the functional relationship between the response variable and the predictors, because it is the best way to understand how they work.

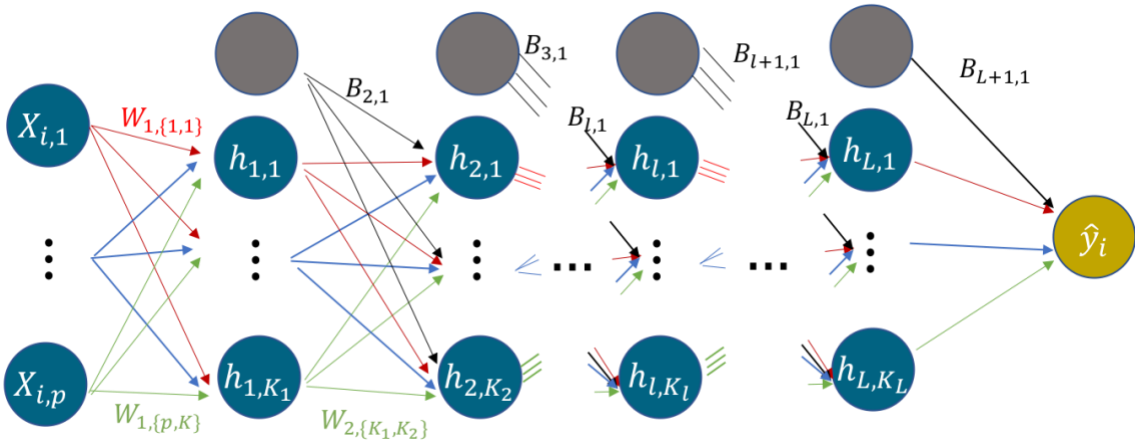


Figure 2.3: Graphical representation of an ANN with L hidden layers.

In an ANN with L hidden layers, we have L matrices of weights (W_l) and a final vector of weights (W_{L+1}) that connects the last hidden layer, i.e. latent variables, with the output layer or, in other words, the predicted value. Each hidden layer has K_l nodes, being l the index that refers to a specific layer and also an activation function f_l because, as we explained before, within a hidden layer all nodes have the same activation function, but different hidden layers can have different activation functions.

Therefore, and according to what we explained in the one-hidden-layer ANN, the nodes of the first hidden layer for all individuals could be obtained through

$$h_1 = f_1(XW_1). \tag{2.7}$$

Since X is the model matrix, with dimensions Nxp and W_1 is a matrix of weights of pxK_1 then h_1 is a matrix of NxK_1, which contains for each individual, the value in each node of the first hidden layer.

As it is shown in the graphical representation, each node of the second hidden layer is connected to each node of the first hidden layer, meaning that a node of the second hidden layer comes from a linear combination of all the nodes in the first hidden layer. The weights of these linear combinations are stored in W_2, which is a matrix that has dimensions K_1xK_2 and, the independent term of these linear combinations is stored in the vector B_2, of dimension K_2. Therefore, one could calculate the value of each individual in each node of the second layer with

$$h_2 = f_2(h_1W_2 + B_2) = f_2(f_1(XW_1)W_2 + B_2), \tag{2.8}$$

where, h_2 will be of dimension NxK_2 and it will store the value that each individual has in each node of the second layer.

If we follow this rationale we would obtain the value of each individual in the last hidden layer, which would be stored in h_L and by applying the last linear combination stored in W_{L+1} (dimension K_Lx1) and the independent term B_{L+1} (a scalar), we then would get the predicted value or, in other words, the output node. Therefore:

$$\hat{Y} = h_LW_{L+1} + B_{L+1}, \tag{2.9}$$

and if we recursively substitute h_L then we obtain the following functional relationship between X and Y

$$\hat{Y} = (f_L(\dots[f_2(f_1(X_iW_1)W_2 + B_2)\dots]W_L + B_L)W_{L+1} + B_{L+1}. \tag{2.10}$$

Once we have obtained this functional relationship, we can apply an iterative optimization algorithm that minimizes and arbitrary error function in order to find the values of all weights matrices W and independent terms B that yield the best prediction, according to the chosen error function. However, since the error function is non-convex we will only be able to find a local optimum so, once we have obtained an ANN by any established method (remember that the number of techniques is extraordinarily wide) we are not able to state whether if the set of obtained parameters is the best or not.

The reader should have in mind that during this subsection we have explained the rationale of how the parameters of an ANN are found once the number of hidden layers, the number of nodes per hidden layer and the activation function per hidden layer are defined. In other words, if we want to fit an ANN, first we need to specify an architecture (number of hidden layers L, number of nodes per hidden layer K_l and the activation function of each hidden layer f_l) and, afterwards, we can obtain a local optimum set of parameters through the rationale explained during this subsection.

The key point behind the architecture is that the performance of a specific architecture heavily depends on the particular problem that one is analyzing and, therefore, empirical methods have been developed in order to find a suitable architecture for the problem, so a satisfactory prediction is obtained. In fact, one of the most used techniques is k-fold Cross-Validation, in which several architectures are tested and the one that yields a smaller out-of-sample prediction error is considered the most suitable one. Of course, this definition is too brief to describe k-fold Cross-Validation but since in this thesis we propose a new approach, when we propose our technique, we will explain the details of k-fold Cross-Validation. However, one can review the foundations and implementation of k-fold Cross-Validation in (Kohavi, 1995). Therefore, the conclusion is that in order to find a suitable architecture, k-fold Cross-Validation is vastly used and, in this project, we will use it as a baseline to compare our proposed method.

Even though the performance of an architecture depends on the particular problem, there has been research projects that aimed to discover if a particular architecture is systematically superior to another in order to reduce the possible architectures when implementing k-fold Cross-Validation. The most relevant result obtained is the Universal Approximation Theorem, demonstrated both in (Cybenko, 1989) and (Hornik, 1991), which, in a few words, states that if the activation function is sufficiently smooth (functions like *logistic* and *tanh* fulfill this condition), then an ANN with only one hidden layer is sufficient to approximate any kind of function and, therefore, it is capable to approximate the function that relates the predictors X and response variable Y and, as a consequence, yield accurate precisions.

As a consequence, one could think that maybe multilayer ANN as the one presented in this subsection should not be considered, because they are just a complication and a similar result could be obtained with one hidden layer. However, this would be a wrong intuition because, even though the Universal Approximation Theorem states that with only one hidden layer we could obtain a suitable ANN, it does not say anything about the number of nodes in the hidden layer. Therefore, if we are not close to the correct number of nodes, a one-hidden-layer ANN will yield an unsatisfactory performance and finding the correct number of nodes would be

too costly. Furthermore, what is relevant is that even though if we are away from the correct number of nodes in the first hidden layer, if we add a second hidden layer the generalization capabilities are enhanced and, as a consequence, it is likely that a two-hidden-layer ANN will yield a more accurate prediction than an ANN with only one hidden layer in which we are away from the correct number of nodes. In other words, it is easier and quicker to find a better ANN by adding layers than by finding the optimal number of nodes with only one hidden layer.

5.2.3 Settings in which Artificial Neural Networks present a comparative advantage

In order to understand the situations in which an ANN is preferable, we will extract some pros and cons of ANN with respect to a baseline model, which will be the linear model with Gaussian response since it is the basic tool for statistical analysis.

The major advantage of an ANN with respect to a linear model is that the former is able to flexibly harness nonlinearities and, since in almost all real studies the relationship between the predictors and the response variable is fuzzy and not linear, causes the ANN to yield enhanced predictions if we compare it to the linear model. Of course, the linear model is able to capture nonlinearities both by transforming the variables, which is complicated because the suitable transformation should be implemented, or including polynomial terms of the predictors, which clearly restricts the functional relationship between the predictors and the response variable and, as a consequence, is not able to generalize as much as the ANN.

Another advantage of ANN is that it is easier to implement than a linear model when there is a large set of predictors. In particular, when lots of predictors exists, in a linear model we need to focus on finding the correct interactions and polynomial degree of the numeric predictors which translates into a vast amount of time finding the optimal model. On the other hand, in an ANN the predictors are inherently combined due to the rationale of an ANN and, therefore, the user does not have to bother about deciding how explanatory variables should be introduced into the model. Of course, depending on the objective of a study, this feature would be considered an advantage or a disadvantage. In particular, when our goal is to predict a response variable, this ability of easily combining the predictors would be desirable, because we would avoid spend time finding the optimal model. However, if our goal is to find the effect of some predictors on a response variable then, of course, this automatic combination of predictors would be a downside of an ANN because we would not be able to assess the effect of each predictor and how they relate with each other.

In fact, this last consideration leads us to one of the uppermost disadvantages of ANN with respect to a linear model: it is a black box. In an ANN the effect of each predictor is diluted

into many coefficients, since for each input node there are several parameters associated. Moreover, even though if we could summarize those coefficients, that would not be enough to extract the effect of a predictor, because its effect is combined with the effect of other predictors through the hidden layer, in what we have been calling the "latent variables". On the other hand, in a linear model, assessing the effect of a predictor is a straightforward task, since it is harnessed by a parameter, or a small set of parameters if there are interactions. Even though this is a clear disadvantage of ANN when the goal is to assess the effect of the predictors on the response variable, it is not important when our goal is to simply predict a response variable, because in that case what is relevant to us is to obtain a small error when predicting.

If we focus on studies where prediction is the main goal, ANN still have some disadvantages which can cause it to yield worse predictions than the ones obtained by the linear model. The first disadvantage is that ANN are prone to overfitting (Lawrence & Giles, 2000), which means that they present an extraordinary predictive performance within the observations used to estimate the parameters, but they do not perform well when predicting the response variable for new observations. There are two main reasons that can cause the ANN to overfit: the fact that there are too many parameters (that is why we need to specify a suitable architecture) and the fact that since we are obtaining the values through an iterative optimization algorithm the weights can take values too fitted to the training data because only then the algorithm finishes (that is why techniques like regularization were devised in Machine Learning). Therefore, if a wrong architecture and regularization constant is used, then the performance of an ANN can be smaller than the one obtained by the linear model, so the user needs to spend time finding a suitable architecture and regularization constant to avoid overfitting.

In the statistical linear model, the fact of being overfitted only comes from including too many parameters/predictors in the model, because the weights are not obtained from an iterative optimization algorithm on an error function. Therefore, it is easier to avoid overfitting because we have a higher control on the model parameters, since they are directly related to only a particular predictor and, as a consequence, they are not as depending with each other as in an ANN. Therefore, we can easily remove those parameters that are causing our model to overfit by removing from our model matrix those variables that are not useful, usually using some indicator like information criteria.

Finally, the last disadvantage of ANN, which affects both studies where the goal is to assess the effect of the predictors on the response variable and studies where the objective is to yield accurate predictions, is that they are not able to yield interval-based prediction for the response variable nor interval-based estimation of the parameters. Instead, they only deliver

a punctual prediction for the response variable according to a punctual estimate of the parameters. In fact, this is the major disadvantage that an ANN has, because predicting without an interval greatly diminishes the added value of statistics in decision-making because the user of the model's output does not know how much sure he can be about the prediction offered.

In order to envision the relevance of interval-based prediction, we will use an example about the topic of this project: a pricing model. Imagine that a user of Airbnb is provided a pricing model based on an ANN, which is not able to deliver interval-based prediction, and, after introducing the different explanatory variables the ANN yields a price prediction for an apartment like the one that he/she is publishing of 22 € per night, which would be an estimate of the market price. After having this information, the user would not be able to know what would happen if he/she set the price at 30 €: would it be too high for the demand and he/she would be placed out of the market or is it close to 22 € and maybe he/she is losing an opportunity of setting a higher price? However, if the pricing model was a linear model, which is able to yield interval-based prediction, then it would tell the user that the market price for an apartment like the one he/she is publishing is between 17 and 35 euros, with a punctual prediction of 22 €. With this information, the user would know that if he/she places a price of 30 € he/she should expect to get some customers, even though, of course, his/her apartment is over the expected market price (22 €). Therefore, the user would be able to take a better decision and, for instance, if he/she does not want to have the apartment completely occupied due to the fact that he/she is not available to receive so many customers, then the user could fix, without doubt, a price of 30 € and still receive some clients.

Now that we have stated the uppermost role that interval-based prediction plays, the following point would be to establish why we are not able to extract interval-based prediction from an ANN. If we follow the rationale established by the Maximum Likelihood Estimator (MLE), then we would need to extract the likelihood, apply logarithms and, afterwards, find the set of parameters that maximize it. If we, for instance, assume a normal distribution, then our goal would be minimizing the sum of squared errors, because that would maximize the likelihood. However, the derivatives with respect to each parameter of this likelihood is too complicated because each derivative depends on almost all other parameters, since some parameters are multiplying a linear combination of other parameters. As explained during this subsection, iterative optimization algorithms are applied so, in fact, even though we set the error function as the sum squared error, we would not obtain the MLE because we would only be able to extract a local optimum of that function. Even though we supposed that we extracted the MLE, then we would need to specify the Fisher information matrix in order to

define the asymptotic variability of the parameters but, of course, obtaining this matrix is also intractable.

Therefore, obtaining real confidence intervals directly from the MLE approach is out of discussion but, of course, one could use Bootstrap in order to obtain intervals of the predicted value (Dybowski & Roberts, 2001). Even though there are studies, like (Paass, 1993) and (Franke & Neumann, 2000), that propose Bootstrap as a valid method to obtain interval-based predictions for ANN we will not delve into them because our goal is to offer intervals for the response variable of new observations, i.e. the possible value that this new observation can take, because we want to capture the real market price, not the expected one. Moreover, and speaking only about intervals of the predicted value (i.e. expected response variable), according to Dybowski and Roberts, the intervals obtained through the Bayesian framework may be better to capture uncertainty and, therefore, deliver better interval-based predictions.

Nevertheless, since Bootstrap is a widely implemented method, we compared the results from the Bayesian framework with those yielded by an implementation of the Bootstrap method that we devised in order to allow bootstrap to offer intervals for the response variable. A full explanation on how we applied Bootstrap in our case is explained in the annex (*Comparing uncertainty treatment in BNN against Linear Model and bootstrapped ANN* from the section *Summary of the most relevant results of our methodology through an example dataset*) but, before delving into it we recommend the reader to review section 6 (*the Bayesian Neural Network proposed in this project*). The main key of this technique is that we relied on a Gaussian distribution to offer the intervals and, since we do not know the MLE estimators in the case of a Neural Network, we had to use approximate estimators, which caused our intervals to be highly unreliable.

After reviewing the uppermost pros and cons of ANN, it is clear that ANN is a suitable tool when our goal is to predict a response variable and the number of predictors is too high. The main reason is that ANN is able to yield more accurate predictions than the linear model and, moreover, the ANN implicitly relates all variables so the effort finding the best model is smaller than in linear model even though, of course, a correct architecture must be found. On the other hand, it is obvious that ANN shortcomings allow the linear model to be a preferable tool when we aim to assess the effect of the explanatory variables on the response variable, basically because even though ANN is able to generalize to a large set of functions, its interpretability can never be as straightforward as the one offered by the linear model.

However, a final consideration should be that, even though the linear model is able to show the effect of each explanatory variable, we need to have in mind the fact that the easiness of

interpreting a linear model diminishes as the number of predictors increases, since complicated interactions are originated and, as a consequence, one has to interpret a large number of parameters in order to understand the effect of a variable. Moreover, another problem that arises when fitting a linear model with a large set of explanatory variables is that, usually, several models present a similar value of an information criteria and, even though, we would choose the one with the smaller value, we would be discarding several plausible models and, therefore, it could be the case that the selected model was not the correct one and, as a consequence, the conclusions about the effects of the variables would not be correct. Moreover, it could be the case that the used distribution for the response variable was not exactly correct, which would mean that the chosen model by information criteria would be wrong and, therefore, the conclusions on the effects would also be wrong. With this discussion, we want to state the difficulty of explaining the effect of some predictors on a response variable and, even though the linear model is the most suitable tool for that, it may not be sufficiently good enough. In particular, in P2P OM we expect to be endowed with a dataset with lots of explanatory variables, since the information is automatically registered by the platform and, therefore, aiming to explain the effect of each variable would be a grueling task, that will not be pursued because, as explained during the section 4 (*Pricing model as a tool to ascertain the market price*), our goal will be to deliver the P2P OM user a prediction of the market price so he/she can take a better decision about the price.

5.3 Motivation to use a Bayesian Neural Network

In the previous subsections we described the two foundations of a feed-forward Bayesian Neural Network (from now on: BNN), which are the Bayesian framework and feed-forward Artificial Neural Networks (ANN).

Through what has been explained in those subsections, one can start to perceive the motivation to develop a BNN as a pricing model. In particular, the main reason is that thanks to the inherent flexibility to generalize any function that ANN have, one can obtain more accurate predictions for the market price than the ones yielded by a linear model and, as a consequence, deliver a more realistic and useful information to the user of P2P OM. Therefore, through ANN we aim to obtain an accurate punctual prediction and, with all the opportunities that the Bayesian framework offers, we expect to be able to endow that accurate punctual prediction with some bandwidths to enhance the amount of information offered.

However, those are not the unique reasons why we think that BNN would be a more suitable model to predict a price in the P2P OM setting. In particular, the fact that the number of

predictors is large, a BNN seems to be a better option than a linear model, because we reduce the amount of time committed to find the best linear model (interactions, polynomial degrees…), just as it was explained with ANN in the previous subsection.

Moreover, since in a BNN the weights and biases will be estimated through their posterior distribution, which naturally considers generalization, instead of an iterative optimization algorithm that excessively focus on fitting the training data[3], there are also reasons to believe that BNN will avoid a part of the overfitting problem of ANNs (the one that appears when estimating the weights) and, therefore, will yield better predictions than ANN. However, as explained in subsection 5.2.3 (*Settings in which Artificial Neural Network present a comparative advantage)* the fact that an ANN overfits the training data is not only due to how the weights are obtained, but also how many weights exists (i.e. the architecture) and, as a consequence, we will still need to deal with this to avoid overfitting in BNNs.

Therefore, there are grounded reasons to believe that the performance of a BNN to predict a market price will be better than the one offered by the linear model and, the goal of this project is to devise a Bayesian model that encapsulates the rationale given by a basic ANN, so we can obtain a basic BNN. However, before entering into this relationship between ANN and a statistical model, we will offer a brief description of BNN history and how they have been developed.

5.4 Bayesian Neural Networks history and literature

During the restoration of the interest on ANN in the late 1980s and 1990s, some research about the relationship between the Bayesian framework and ANN was conducted. In particular, one of the first authors that tried to connect the Bayesian setting with Neural Networks were Opper and Haussler, who aimed to estimate a Perceptron through an algorithm related to the Bayesian framework (Opper & Haussler, 1991). Other authors connected the Bayesian setting with Neural Networks but, instead of trying to develop a predictive model based on the Bayesian framework, which would be a BNN, they used Neural Networks as a way to approximate a probability distribution (Baum & Wilczek, 1988), (El-Jaroudi & Makhoul, 1990). Finally, another interesting point of view relating the Bayesian

[3] For instance, if a linear model is fitted using an iterative optimization algorithm on an error function, then it is likely that there will be overfitting, and, in fact, that is the reason why Ridge regression was devised. In contrast, if we fit a linear model according to a model and using the posterior distribution, then we avoid overfitting, just because of how the weights are obtained. However, in that case if the model overfits will be because there are more predictors than it should, rather than how the weights have been obtained.

framework and ANN was the probabilistic interpretation of a Neural Network given in (Tishby, Levin, & Solla, 1989).

Even though these articles were the first to link the Neural Networks and the Bayesian setting, the research focused on BNN started with MacKay (MacKay, 1991a) and (MacKay, 1991b). MacKay was the first influential author to devise a Neural Network from the Bayesian setting but, instead of treating it just as another statistical model as we propose in this thesis, he focused on describing a type of BNN that would resemble an ANN. In fact, he translated the ANN into the Bayesian framework in order to find a founded and sensible approach to determine the architecture of an ANN and the regularization constant that penalizes the model complexity of a Neural Network in a more natural way than cross-validation.

One of the most interesting results of his magnificent work is that, if we assume, through the prior distribution, that all the weights of a BNN follow a normal distribution centered at the origin and with its variability being ruled by a common hyperparameter (α) and, moreover, we assume a Gaussian distribution for the response variable, it is the same setting as an ANN in which we minimize the sum of squared errors adding a regularization term in this error function. Moreover, he demonstrated that the hyperparameter α is, actually, the regularization constant that is obtained through cross-validation in ANN so, since we are in the Bayesian setting, α will have a posterior probability distribution and we will be able to decide its value according to it. Apart from demonstrating this, MacKay also emphasized that, in the Bayesian framework, on top of interval-based prediction, we could obtain the evidence that the observed data gave to each architecture which, in probabilistic terms is $P(Y|H_j)$, where H_j stands for a particular architecture. As a consequence, we would be able to choose between different architectures in a natural way, using only the training data.

In order to fit the described BNN, MacKay devised a particular framework. First of all, an iterative optimization algorithm, just like in ANN, was implemented to find an optimum and he called the found set of parameters the maxim probability weights (W_{mp}). Afterwards, MacKay used a Gaussian approximation for the posterior distribution, which endowed him with some simplifications that allowed him to use W_{mp} to analytically integrate and obtain the posterior distribution of the BNN's parameters, the evidence associated to each architecture and also a posterior distribution for the regularization constant. The conclusion was that the regularization constant would be automatically set to the mode of its posterior distribution and, in order to choose the architecture, the one with the highest evidence would be the selected.

However, the framework devised by MacKay had several shortcomings which were, basically, the fact that by approximating through a Gaussian the posterior probability distribution we are assuming that there is a unique optimal solution for a Neural Network which, by definition, is wrong because there are several optima. As a consequence, MacKay proposed conducting this same analysis for different starter points and different architectures so multiple W_{mp} could be obtained and, instead of using just a Gaussian, a whole mixture of Gaussian, each relating to a different local optimum, would be employed.

Even though that new approach was promising, an important drawback of the MacKay's evidence framework was found by Thodberg, which kind of invalidated MacKay's rationale (Thodberg, 1996). In particular, Thodberg applied MacKay's BNN in different cases and he found that the correlation between the model evidence and the out-of-sample predictive performance was too scarce, basically due to the fact that the Gaussian approximation used by MacKay yielded poor approximation to the posterior distribution and, therefore, a poor estimate of the model evidence.

Therefore, the main conclusion that the reader should extract from MacKay's work is that he focused in envisioning the ANN into the Bayesian setting, instead of building a Bayesian model with the same rationale as an ANN. Moreover, even though his approach was theoretically grounded and ambitious, its implementation was not as brilliant, causing his BNN to yield poor performance, non-reliable decision procedure about the Neural Network architecture and, moreover, not credible intervals, because they were extracted from an erroneous posterior distribution.

After MacKay proposed the first BNN, the next relevant contribution was done by Buntine and Weigend (Buntine & Weigend, 1991). These two authors, pondered about different ways of introducing prior knowledge in a BNN, since the management of it is crucial in the Bayesian setting and it can help to interpret the results obtained. However, and as it happened with MacKay contributions, the key point of all these techniques reviewed by Buntine and Weigend is that, instead of offering an up-to-bottom Bayesian approach to obtain a Neural Network, they adapted the ANN concept in order to encapsulate probabilities and, therefore, be able to give some interval-based predictions. Moreover, in all these mentioned articles the authors base their explanations in analytical demonstration and test their techniques in simplistic simulated examples, but they do not extend the analysis to applied cases.

Even though those were the first approaches to introduce the Bayesian rationale into ANN, the next author that revolutionized the field of BNN was Radford Neal. In his first articles (Neal, 1992), (Neal, 1993) and his book (Neal, 1995), which has become one of the most important

reference handbook for BNN, Neal aimed to fit a BNN through Markov Chain Monte Carlo (MCMC) methods, instead of relying on approximations of the posterior distribution by a Normal distribution, which is what MacKay proposed in his project.

In Neal's work, the proposed MCMC method to simulate from the posterior distribution of the BNN parameters was the Hybrid Monte Carlo (HMC), which was devised by Duane (Duane et al., 1987). In a summarized way, the HMC uses gradient information in order to speed up the exploration of the parameter space and, as a consequence, is able to simulate from the posterior probability distribution with a significantly smaller amount of time than established MCMC methods such as the Metropolis algorithm.

Even though Neal's uppermost contribution is that he proposed HMC as an alternative procedure to obtain the posterior distribution in a BNN, he further analyzed the framework developed by MacKay in which all weights are random Gaussian variables the variability of which is subject to a hyperparameter α that controls the model complexity, as a way to introduce regularization into BNN. His main conclusion about it was that MacKay focused too much on replicating the ANN structure in the Bayesian setting and that, if the Bayesian approach is completely implemented (which is accomplished by working with MCMC instead of Gaussian approximations), one should not be concerned about the model complexity and regularization, since the posterior distribution tends to automatically penalize overfitted models. As a consequence, and following the Universal Approximation Theorem, he devised his own type of BNN in which the number of nodes in the hidden layer is large and the weights that connects the hidden layer with the output layer follow a Normal distribution the variability of which negatively depends on the number of nodes in the hidden layer.

In a summarized way, the main contribution of Neal is that he generalized the approach of BNN and made visible the potential that this model comprises, without the requirement of strictly emulating an ANN. However, his research also had its downsides, which mainly were that he only worked with small simulated examples and, moreover, he only focused on one-hidden-layer BNN. Apart from that, even though working MCMC was able to better generalize the BNN and yield better performance, it had its own shortcomings which were, basically, the difficulty on how to assess the convergence to the stationary state of the Markov Chain and, moreover, the fact that evidence could not be extracted, which basically meant that model comparison was no longer possible with this approach.

After this work, Neal focused his career on developing faster and reliable MCMC methods, that could work not only for BNN but also for other complicated Bayesian Models. Moreover,

in order to spread the use of these furtherly-developed MCMC methods, he produced a software called Flexible Bayesian Modelling, which aimed, basically, to fit BNN.

During the last part of the 1990s and first years of the 21st century the literature on BNN was based on reviewing and implementing in real cases MacKay's evidence framework and Neal's contribution, being (Thodberg, 1996) previously discussed, (Lampinen & Vehtari, 2001) and (Vivarelli & Williams, 1997) the most relevant contributions.

If we focus on (Lampinen & Vehtari, 2001) there are two important conclusions extracted. The first one is that the less restrictive the prior was (i.e. the less informative it was) the better was the generalization of the BNN, meaning that the BNN obtained smaller out-of-sample error metrics. The second conclusion was even more interesting, since it stated that, for some cases, BNN is able to outperform ANN.

In (Vivarelli & Williams, 1997) the authors compare the performance of MacKay's evidence framework against the performance obtained by Neal's contribution to train BNN and the conclusion is that, for relatively small datasets, the Hybrid Monte Carlo solution (i.e. Neal's contribution) seems to be superior than MacKay's evidence framework.

Due to the relevance that Neal's contribution took, further research about using MCMC to simulate from the posterior distribution of neural networks weights was conducted, being (Müller & Rios, 1998) and (Vehtari, Sarkka, & Lampinen, 2000) the most influential contributions. Both authors discuss how MCMC should be implemented in order to efficiently simulate from the posterior probability distribution of the BNN's parameters. While Müller and Rios focused on devising a simulation schema to capture multimodality, Vehtari focused on how many chains should be used and, moreover, how to choose the starting values for those chains. Other authors, like Freitas aimed to implement new MCMC algorithms, based on the Hybrid Monte Carlo method (Freitas et al., 2000).

Even though MCMC inference, based on Neal's contribution, was relevant, other Authors like Hinton (Hinton & van Camp, 1993) and Barber (Barber & Bishop, 1998) aimed to use Variational Inference in BNN instead of MCMC methods in order to obtain the posterior distribution of the BNN's parameters. Since MCMC methods required too many time to simulate from the posterior distribution, the Variational Inference approach started to gain relevance in the field of BNN, even though its generalization capabilities were smaller than the ones offered by the MCMC approach.

Even though the results were encouraging, especially the ones found in (Lampinen & Vehtari, 2001), during the first decade of the 21st century, the literature on BNN greatly diminished, mainly due to the fact that the proposed methods until the date to approximate the posterior distribution in complicated Bayesian models, as BNN, were not able to scale to real problems easily. In order to overcome this bottleneck, two main approaches to obtain the posterior probability distribution, both based on MCMC methods, were developed.

The first approach was based on Online Machine Learning, which is a batch-based learning procedure in which one does not need to load on memory all the observations because the algorithm can learn from sequentially given observations. Successful authors that implemented this rationale into Bayesian learning (i.e. obtaining the posterior distribution) were Welling and Teh (Welling & Teh, 2011) and they named "Stochastic gradient Langevin Dynamics" this MCMC method. Afterwards, their contribution was applied to BNN.

The second most important approach was based on further developing the Hybrid Monte Carlo (HMC) method, which was the one proposed by Neal for BNN. In particular the goal was to speed up the process, like in (Freitas et al., 2000), and, at the same time, obtain more robust algorithms than the original HMC. One of the most important development of HMC was done by Hoffman and Gelman, where an adaptive HMC sampler, called the No-U-Turn Sampler (NUTS) was proposed (Hoffman & Gelman, 2014). Currently, there are no published articles about the implementation of NUTS in BNN and, in fact, one of the goals of this project is to apply this MCMC method in BNN and test its performance. Moreover, we aim to overcome one of the main drawbacks that the MCMC methodology has in BNN: how to assess if a Markov Chain has entered its stationary state.

Even though these two approaches were devised to speed up and facilitate MCMC inference in complicated Bayesian models, the reader should have in mind that for BNN in particular, the uppermost method used has been Variational Inference. Even though Variational Inference is not as general as MCMC approaches because it depends on which family of distributions is used to approximate the posterior distribution, its computation time is greatly smaller and, in some cases, is the only non-prohibitive technique. Since this is the main approach for BNN, several packages for programming languages as Python and R have been developed, being Edward[4] the most influential.

Finally, the reader should know that even though BNN started almost thirty years ago, the interest on BNN has been significantly increased during the last 3-4 years, causing it to be a

[4] http://edwardlib.org

salient topic in the area of Neural Networks. As a consequence, multiple research projects have been instituted with several goals such as enhancing the performance of BNN, reducing the computational time to obtain the posterior probability distribution, identifying a way to properly harness prior knowledge, designing a methodology to compare architectures… Therefore, the reader should understand that BNN is a continuously changing field that is constantly growing but in which there is not a completely established roadmap on how to obtain a BNN.

6. The Bayesian Neural Network proposed in this project

6.1 Model Formulation: Bayesian Neural Network as the bridge between the statistical model and Neural Networks

Due to the fact that BNNs are still at an early stage, in this project our goal is to spread the acknowledgement of its potential as an alternative model for those studies where the main goal is to predict a response variable. Moreover, we believe that the most reasonable way to deal with a BNN is to envision it as a statistical model because, once we have done that, we will be able to apply several results offered by parametric statistics and speak about the uncertainty of predictions. In fact, this interpretation of a BNN as a statistical model is natural and straightforward and it is one of the main contributions of this project, since we avoid dealing with Neural Networks from the computer science and artificial intelligence approach. Of course, other authors like Müller and Rios (Müller & Rios, 1998) proposed a similar approach but, in our case, we generalize it without relying on any particular priors and one-hidden-layer BNNs.

The statistical model that we propose in this project states that the response variable, conditioned to some parameters and the predictors, follows a Normal distribution in which the μ is obtained through a Neural Network and the variability is an unknown parameter σ. Therefore, and recovering the results obtained in subsection 5.2 (*Rationale of feed-forward Artificial Neural Networks*) the model that we are specifying is the following

$$\begin{gathered} Y_i|X_i, W, B, \sigma \sim N(\mu_i = g(X_i, W, B), \sigma) \text{ where} \\ \mu_i = g(X_i, W, B) = (f_L(\dots[f_2(f_1(X_i W_1)W_2 + B_2)\dots]W_L + B_L)W_{L+1} + B_{L+1}, \end{gathered} \tag{2.11}$$

where $g(X_i, W, B)$ is just the function that captures the linear combinations and applications of activation functions to obtain the output of a Neural Network with L hidden layers (equation 2.10). In this proposed BNN all the priors will be uninformative and, particularly, improper. The idea behind this decision is the fact that, as all BNN's researchers argued,

harnessing prior information in a BNN is very complicated, since the model itself is a black box difficult to be interpreted. Moreover, due to the results obtained in (Lampinen & Vehtari, 2001), it seems reasonable to start without forcing any restrictive prior information.

It is relevant to note the fact that, since we are treating the BNN completely from the Bayesian perspective (instead of trying to resemble our BNN to an ANN as previous authors did), there is not any type of regularization explicitly specified. By proposing this tabula rasa to deal with BNN, we expect to avoid falling in the same pitfalls encountered by previous authors. Even though the model defined by equation 2.11 will be our baseline BNN, we will afterwards explore the possibilities that BNN offer through hierarchical models, focusing specially in techniques used to help interpret the BNN, like Automatic Relevance Determination (ARD) proposed by Neal and MacKay. In subsection 6.3.2 (*Regularization and Automatic Relevance Determination through hierarchical models*) we delve into explicit regularization and, moreover, we review how to introduce ARD as an extension of our proposed BNN.

In fact, conceiving a BNN as the statistical model showed above assumes that a BNN is just a nonlinear model and, compared to a linear model, in which $\mu_i = g_2(X_i, \beta) = X_i\beta$, the "only" thing that is different is the function that defines the localization parameter. With this approach, we are stating that our BNN is a parametric model, since we are assuming that the variability and behavior of what we observe can be captured by a probability distribution the parameters of which have a fixed dimension. However, we are aware that even though, by definition, the proposed BNN is a parametric model, some controversy can arise around the idea of a Neural Network being a parametric model. In the following paragraphs, we will discuss several topics around this controversy.

Our main argument to state that the proposed BNN is a parametric model is that, once the architecture of the Neural Network has been defined, the number of parameters is fixed, so the requirement established by the definition of parametric model is fulfilled.

Another usually employed definition for a parametric model is that the number of parameters, i.e. the dimension of the parameter space, does not change according to the data that we observe. One could, then, argue that in the proposed BNN we will need to specify an architecture according to our data, so we are modifying the dimension of the parameter space and, as a consequence, the proposed BNN should not be considered a parametric model. However, this rationale would also imply that a linear model is not a parametric model because in a linear model we add and remove interactions and polynomial terms according to the data that we have collected, since we remove those that are not able to reduce an information criterion, or, in the frequentist approach, that are not significant.

Another reason why someone could argue that the proposed BNN is not a parametric model is because the parameter space can be infinite, since we could constantly add new layers and new nodes in the hidden layers. However, following this rationale, this would imply that also the linear model is not a parametric model because it can also have an infinite number of parameters since one can add polynomials of arbitrary degree.

The only valid reason that would explain that the proposed BNN is not a parametric model is the one stated by Neal (Neal, 1995). He claims that a BNN is not a parametric model in the sense that, compared to a linear model, there is not an easy interpretation of each of the parameters. However, this reason is rather qualitative, since it is not in the definition of a parametric model that each parameter must be interpretable and easy to understand.

Through all these reasoning, we want the reader to be skeptic about the definition of a parametric model and, furthermore, we propose that changing the architecture of a BNN is not that different from adding new polynomial degrees and interactions to a linear model. In fact, both actions are just a procedure of model selection. On the one hand, in a linear model we change the model matrix (the input layer) because the model itself is not able to generalize many functions since it does not automatically combine the predictors and assess nonlinearities. On the other hand, in the BNN we can keep constant the input layer because the model will automatically combine the variables to approximate the process that is generating the data that we observe, but we need to select a suitable architecture to ensure satisfactory results. Therefore, both in a linear model and in a BNN we need to guide our model, but we do it through different leverages in each case.

Now that it is clear that BNN is a parametric model and that it can serve the same purpose than a linear model, which consists of capturing and being able to replicate the behavior of what we observe through a probability distribution, many questions arise. In particular, if the hypothesis that the BNN will be able to yield more accurate predictions than the linear model holds, the uppermost setting-changer debate that it introduces is that there is a grounded statistical model that is more capable to capture the behavior of what we observe than the linear model and, as a consequence, maybe the linear model should not be applied systematically to all statistical studies.

The response to the previous question is obvious, since the choice of the statistical model will be according to the goal of the study because, as Box said in his already quoted sentence (“All models are wrong, but some are useful”), the aim of a model is to be useful, to serve a purpose, to answer a question, rather than capturing reality as it is. If our goal is to replicate reality and capture it as much as possible without trying to explain it, like it happens in the

case of a pricing model, then statistical models able to generalize more should be considered and, from what we have explained until now, the BNN seems to be a better option than a linear model. However, if our goal is to understand reality in a simplified way, then the linear model would be a more suitable option than the BNN.

Until now, we have only accomplished the first goal of this project, which is to imagine a BNN from the statistical point of view and theoretically explain its potential and, in the following subsection, a methodology to fit a BNN based on MCMC methods is explained along with a whole set of theoretical considerations around the suitability of this proposed methodology.

6.2 Fitting the Bayesian Neural Network proposed in this project

Even though the proposed BNN is grounded from the theoretical point of view, the main challenge that will have to be dealt with will be its implementation, since it will be useful only if we are able to obtain the real posterior distribution. If we are not able to obtain the real posterior distribution then our inference will be invalidated and, as it happened with MacKay, even though from a theoretical point of view he devised a reasonable BNN, its implementation caused his BNN to yield unsatisfactory performance in some examples.

Therefore, and learning from Neal's contribution, in order to implement the proposed BNN we will use MCMC methods and, in particular, since he demonstrated that established MCMC methods like the Metropolis algorithm or Gibbs Sampling were too slow exploring the parameter space of a Neural Network, we will use the method that he proposed: the Hybrid Monte Carlo (HMC). As explained before, one factor that distinguishes our thesis is that we will use a new version of this algorithm, called NUTS, in which there is a constant adaptation of some HMC's parameters that ensure its robustness (Hoffman & Gelman, 2014).

Even though we use Neal's proposal to obtain the weights of our BNN, the main difference of our work is that, instead of fitting a BNN with the structure that Neal introduced (i.e. one-hidden-layer BNN with almost infinite number of nodes in the hidden layer), we propose our own structure, which is based in several hidden layers and a relatively small number of nodes in each layer. The main reason behind this particular structure is the fact that, with it, the time finding a suitable architecture is small and, moreover, we reduce the simulation time of NUTS to sample from the posterior distribution because there are less parameters.

On top of that, our proposed procedure to fit the BNN includes also a new schema on how to apply the MCMC, which discusses relevant concepts suggested by Vehtari (Vehtari, Sarkka, & Lampinen, 2000) such as the starter points for each Markov Chain and the number of Markov

Chains that should be used. Moreover, we also contribute by introducing a new way to assess the entrance to the stationary state of our Markov Chains, even though after proposing and using it, we discovered that some authors already implemented it in other Bayesian models different than BNNs.

As a conclusion from all these contributions, in this project we provide a renewed point of view to justify the use of MCMC methods in order to obtain the posterior distribution in a BNN. In fact, the main idea behind our perspective is that we propose that MCMC are actually capable to capture several local optima, while methods like Variational Inference or Gaussian approximations of the posteriors only focus on one optimum, so MCMC will yield BNN with higher performance. Apart from that, there three more reasons that explain why we have used MCMC.

The first one is that MCMC is the most widely established method in Bayesian Statistics and, as a consequence, we wanted to use it in the Neural Network setting in order to make it easier to spread BNN among the Bayesians' community. The second reason, and maybe a more important one, relates to both the fact that MCMC is easily distributable and the fact that nowadays parallel processing through cloud computing is becoming a spearheading technology in which computation time is extraordinarily reduced and several calculations that were prohibitive are becoming possible. Therefore, we decided to use MCMC methods because its potential is rapidly increasing, since cloud computing industry is developing at an implacable pace. Finally, the third reason that explains why we decided to use MCMC is the fact that several researchers are focusing their efforts in devising new algorithms able to enhance its computation time and robustness, so by using MCMC for our BNN we ensure that any discovering in the MCMC field, which is very active, can be implemented into the BNN.

6.2.1 Consideration about the shape of the posterior distribution

Before applying HMC to sample from the posterior distribution, one must first understand the complexity of this distribution in the case of BNN because only through this pondering the results obtained will be meaningful.

As explained in subsection 5.1 (*Overview of the Bayesian framework and its flexibility*), the two pillars of the posterior distribution are the prior distribution and the likelihood. In the proposed BNN the chosen prior distributions are uninformative due to the fact that a BNN is a complex method where the parameters do not have a straightforward interpretation, so it is a grueling process trying to harness any prior knowledge. Moreover, both in (Neal, 1995) and (Vivarelli & Williams, 1997) it is explained that if the sample is large in a BNN fit with

MCMC, then the prior knowledge reduces its effect and, therefore, using uninformative priors almost do not affect inference. Finally, the third argument that sustains these uninformative priors is found in (Lampinen & Vehtari, 2001), where the authors discuss that with less informative prior the generalization obtained by the BNN was superior.

As a consequence of working with uninformative priors and sufficiently big samples, our posterior distribution will be based, basically, on the likelihood and, therefore, if we analyze the shape and complexity of the likelihood, we will obtain several useful insights about the posterior distribution.

The uppermost consideration that one must have in mind about the likelihood in the proposed BNN is that, like in all BNN and as it was discussed by Neal (Neal, 1996) and Müller and Rios (Müller & Rios, 1998), it will be multimodal, meaning that there are several θ (we will use θ to refer to all BNN's parameters) around of which the likelihood has modes of probability or, in other words, that in the parameter space Θ there are multiple regions with high posterior probability separated by spaces of almost zero posterior probability. In the case of a Gaussian response value, the likelihood is intrinsically related to the squared errors of the prediction and, since we know that if we fit an ANN with the sum squared errors of the prediction as objective function we can obtain several local optima, it is obvious that we will face a multimodal likelihood, being each mode of the likelihood associated to the local optima of the ANN.

The reasoning that explains why the likelihood has several optima is, basically, the fact that its underlying function, $g(X, W, B)$, is a highly irregular function as a direct consequence of the nonlinear dependency that exists between the parameters. In order to assess this high nonlinear dependency between the parameters one only needs to extract the derivatives of $g(X, W, B)$ with respect to each parameter. The multimodality of this function has been documented by many authors and, in this project, we will focus on a brief discussion about it to characterize our likelihood.

The first type of multimodality relevant for our case is the fact that there are symmetric solutions with, obviously, a same value for the likelihood. In fact, as the number of nodes in a hidden layer increases the number of symmetrical local optimareases. This symmetry can be seen in Figure 2.4 where two simplistic Neural Network, with two symmetric solutions are shown.

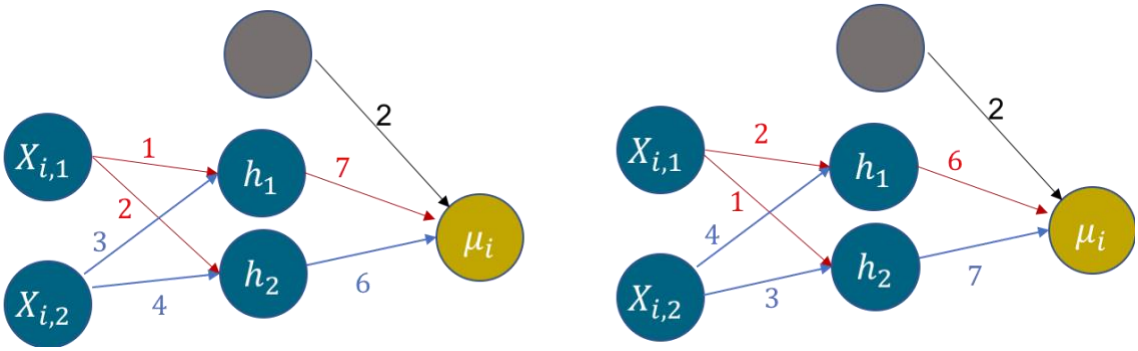


Figure 2.4: Graphical representation of two Neural Networks at two symmetrical local optima.

These two values of θ will have an associated mode in the likelihood with the exact same height, since they yield the same prediction for all individuals. Apart from these symmetric modes, the second type of multimodality is the one shown in Figure 2.5 in which each Neural Network is stuck at different local optima.

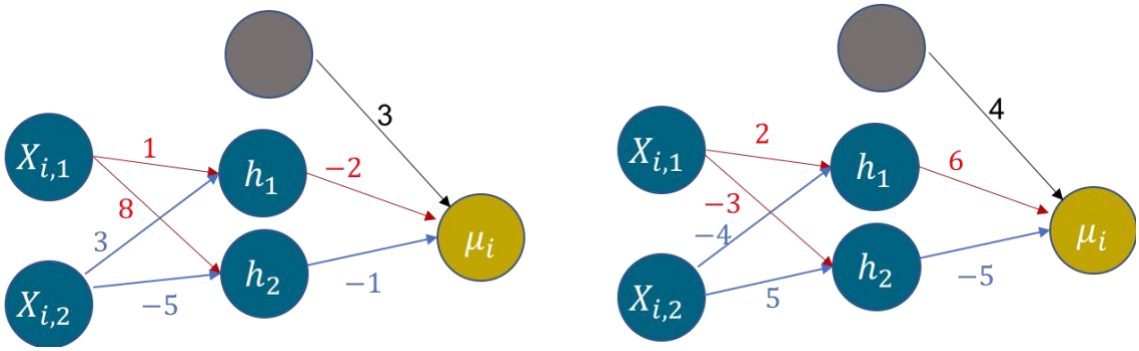


Figure 2.5: Graphical representation of two Neural Networks at two different local optima.

The main problem here is that, even though the two Neural Network are stuck in a local optimum, which means that a further iteration of the optimization algorithm does not sufficiently change the error function, they do not have to yield an exact same prediction, meaning that the value of the optimized error function at each local optimum does not have to present the same value. As a consequence, in our likelihood, for those values of θ there will be a mode, but they do not have to present the same height, i.e. the same accumulated probability.

As explained before, our MCMC-based BNN will capture several local optima, minimizing the risk of getting stuck in an inferior local optimum which would cause our BNN to yield a predictive performance similar to the one obtained from an ANN. Apart from that, it is especially important that our BNN captures more than one local optima because if not we would be biasing the posterior distribution, because only a part of it would be shown and, as a consequence, our interval-based prediction would not be realistic nor credible.

In order to prove how a likelihood can be multimodal, which is the case in a BNN, we will show two simplistic examples. In those two examples the data used is generated from the same model with the following equation

$$Y_i|X_i \sim N\big(\mu_i = 3x_{i,1} + 4x_{i,2}, \sigma = 4\big), \tag{2.12}$$

where x_1 and x_2 are simulated from independent normal random variables. These two examples will be used again during this subsection.

Example 1: Overparametrized linear model (part 1/2)

In order to capture the behavior of the data generated by the previous model, in this first example we will use the following Bayesian model, with uninformative priors:

$$Y_i|X_i, \theta \sim N\big(\mu_i = [\beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2}]\gamma, \sigma\big). \tag{2.13}$$

Of course, we know that we have specified and overparametrized linear model, because we could set $\gamma = 1$ and work with a classical linear model. However, since we have specified this model with γ, our likelihood will be according to it and we expect it to have a maximum of probability in each combination of β_1 and γ in which $\beta_1 * \gamma = 3$ and also, $\beta_2 * \gamma = 4$ because, then, we obtain the exact model that is generating what we observe.

Since Θ is a space of five dimensions, it will not be possible to graphically represent it but, we can fix σ, β_0 and β_2 at an arbitrary point and plot the likelihood according to several values of β_1 and γ. If we do that, we would obtain the surface shown in Figure 2.6:

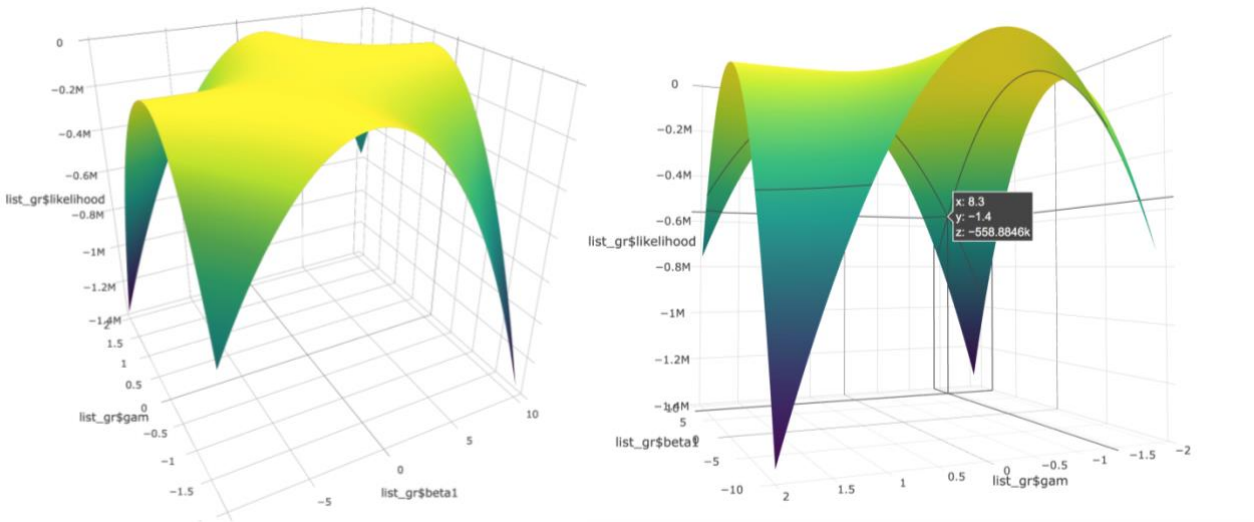


Figure 2.6: Likelihood of the model in equation 2.13 fixing the parameters β_0, β_2 and σ. Surface from above (left) and from below (right).

On the left we have a representation of the likelihood from above in which we can see that there is an almost smooth top, meaning that several combinations of β_1 and γ yield a maximum value for the likelihood. If we observe the likelihood from below, with the representation on the right, we are able to see that for each value of γ (list_gr$gam in the graph) there is a parabola with an associated maximum which, of course, will be that value of β_1 that multiplied by the selected value of γ we obtain 3.

Example 2: Basic BNN (part 1/2)

In the first example we have been able to envision the idea that a likelihood can have several maxima, however, the plotted likelihood does not show that each maximum is associated to a separate mode, since all maxima were side-by-side. In this second example we will use a one-hidden-layer BNN with one node in the hidden layer, with *tanh* as activation function and without independent term in order to fit the same data than before and gain some insights about the behavior of the likelihood in BNNs. In this case, our statistical model would be:

$$Y_i|X_i, \theta \sim N\big(\mu_i = \big[tanh\big(\beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2}\big)\big]\gamma, \sigma\big). \tag{2.14}$$

Like we did before, if we fix β_0, β_2 and σ at an arbitrary value and plot the likelihood for different values of β_1 and γ we obtain the following surface:

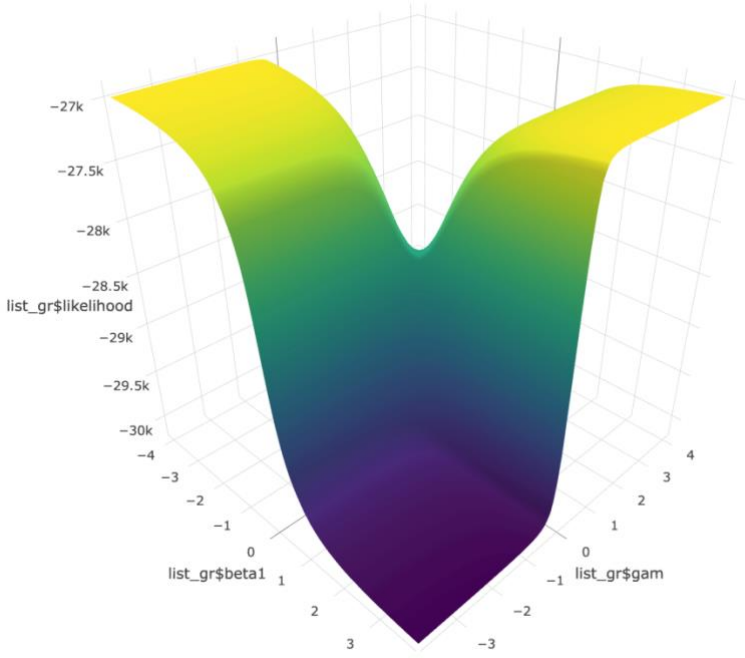


Figure 2.7: Likelihood of the model from equation 2.14 with fixed β_0, β_2 and σ.

Even though with fixed β_0 and β_2, which have a higher relationship with β_1 than in the first example, we are able to see different modes in the likelihood and, of course, one should expect that in the five-dimensional space Θ, since β_0 and β_2 will not be fixed, the number of modes of the likelihood will be bigger.

Therefore, the main conclusion that the reader should extract from this is that a likelihood can be multimodal and, in fact, in BNNs the likelihood is always multimodal and with a huge number of modes. As a consequence, the posterior distribution that must be approximated will have a complicated shape and, since a BNN has lots of parameters, one should expect a considerably high amount of MCMC simulation time to sample from the posterior.

6.2.2 Sampling from a complicated posterior distribution with MCMC methods

Even though simulation time can be prohibitive and must be taken into account, there is an even more important premise that must be held in order to use MCMC methods in BNNs. In particular, this premise is whether if MCMC methods are able or not to simulate from a complicated posterior distribution like the one in our proposed BNN and, in this subsection of the project, we propose a method to reach the true posterior through MCMC simulations.

As it has been explained during the subsection 5.1.2 (*Implementation of the Bayesian approach*), in order to simulate from the posterior distribution, a Markov Chain needs to enter its stationary state. Therefore, we need to assess the convergence of a Markov Chain to its stationary state and, for doing that, there are several techniques. The most basic one is based on plotting each simulated value of the chain for a parameter and when the simulations become steady around some particular values one can assume that it has converged to the posterior probability distribution. However, since it is somewhat difficult to assess convergence by this method, several chains are usually started and when the values simulated by all the chains meet, then it is concluded that we are sampling from the posterior distribution. In order to envision this vastly used procedure to assess convergence in MCMC methods, we are going to fit a linear model with uninformative prior distributions to the data generated by equation 2.12, so our model will be:

$$Y_i|X_i, \theta \sim N\big(\beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2}, \sigma\big). \tag{2.15}$$

In Figure 2.8 there is a representation of all the simulations made by each chain, which is the required plot to assess convergence by the second method explained, the one in which several chains are started.

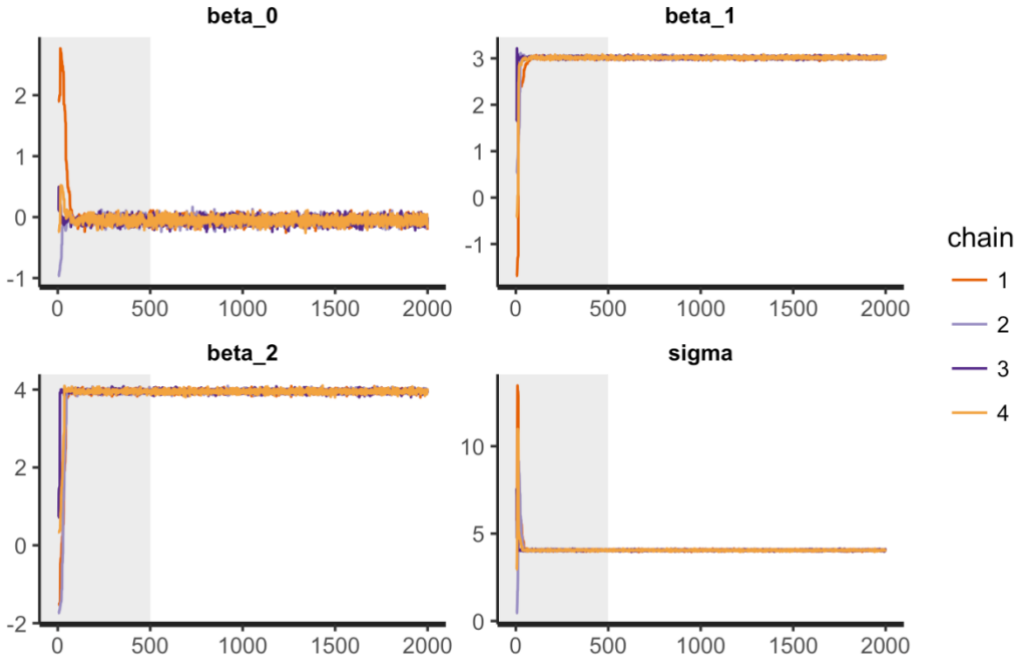


Figure 2.8: Simulations of the parameters from each Markov Chain launched to obtain the posterior distribution of the Bayesian model from equation 2.15.

Since all chains have mixed and they are simulating around some values in a steady way one can conclude that they have reached its stationary state and, therefore, that we are simulating from the posterior distribution. Moreover, one can see that the posterior distribution for each parameter is centered in the value that it takes in the equation that has generated the data (equation 2.12), which is the expected result since it is what the likelihood is telling us.

In this particular case there will not be any problem with the MCMC simulations, since the linear model, by definition, assumes a unimodal likelihood. As it has been said before, our goal is to assess whether MCMC methods, in particular HMC which is the one that we are using, is able to reach the stationary state for multimodal and complicated posteriors. In order to accomplish that, we are going to fit the models presented in the two examples presented above where the likelihood was multimodal.

Example 1: Overparametrized linear model (part 2/2)

The equation of the model that we are aiming to fit is described by equation 2.13 and if we apply the HMC method to simulate from the posterior we obtain the following results for each chain:

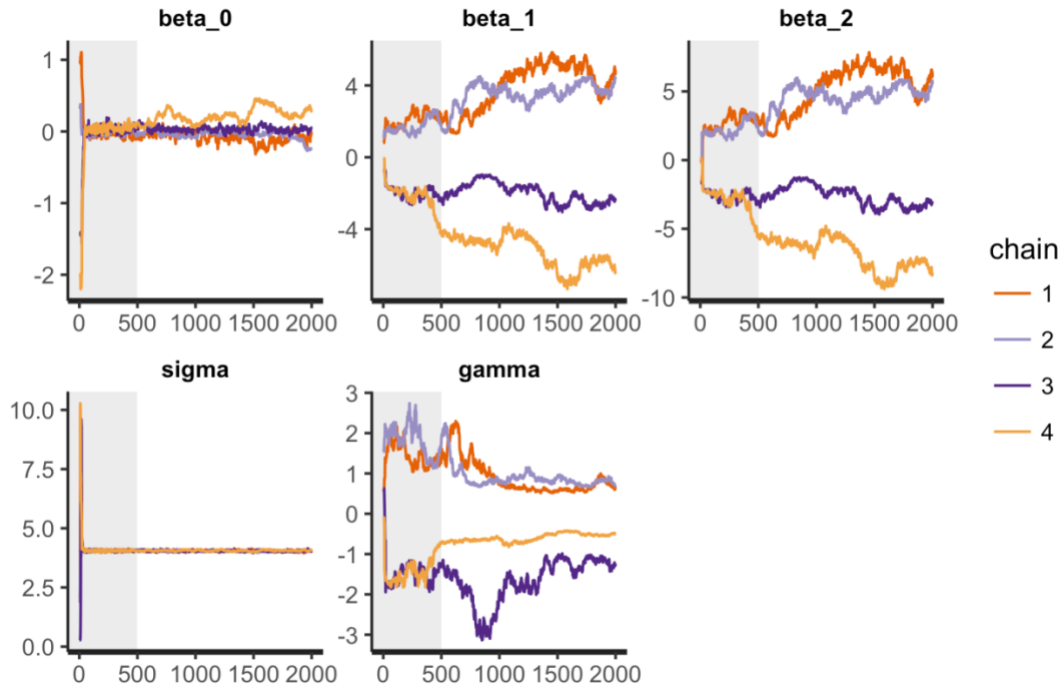


Figure 2.9: Simulations of the parameters from each Markov Chain launched to obtain the posterior distribution of the model from equation 2.13.

After observing these results, it seems that the chains have not reached any stationary state for several parameters, because neither they have meet each other nor have entered a steady state and, as a consequence, that would mean that HMC has not been able to sample from the posterior distribution. However, one can see that for the model's error (σ) there has been convergence of the chains and, moreover, at the real value of the model that has generated the data. In fact, this gives us a clue that our model may have converged, and, in fact, we will demonstrate that it has.

In particular, what is happening here is that we are not able to see convergence because we are looking for it at each dimension of Θ, by separated, when in our likelihood the relationship between the parameters is very high. Our proposal in this project is that, in order to assess convergence of MCMC methods in complicated models one may have to apply some multidimensional function of the parameters and, therefore, look for convergence in this new space generated, different than Θ. Our final goal is to a more informative space than looking at each dimension of Θ by separated. In order to assess convergence, one should calculate the value of the multidimensional function for each simulation of the parameters and, afterwards, assess convergence by tracking the chain generated by these values obtained. This is one of the most important contributions of this project and, from now on, we will refer to this proposal as Multidimensional Convergence for MCMC methods.

For this particular example, a sensible multidimensional function could be computing the product between each β and γ, because we know that it is the natural transformation of our model to the one that has actually generated the data. The plot of the chain associated to the multidimensional function $\beta_1 * \gamma$ and $\beta_2 * \gamma$ is in Figure 2.10 and, in order to obtain it, we have used the simulations of β_1, β_2, γ shown in Figure 2.9.

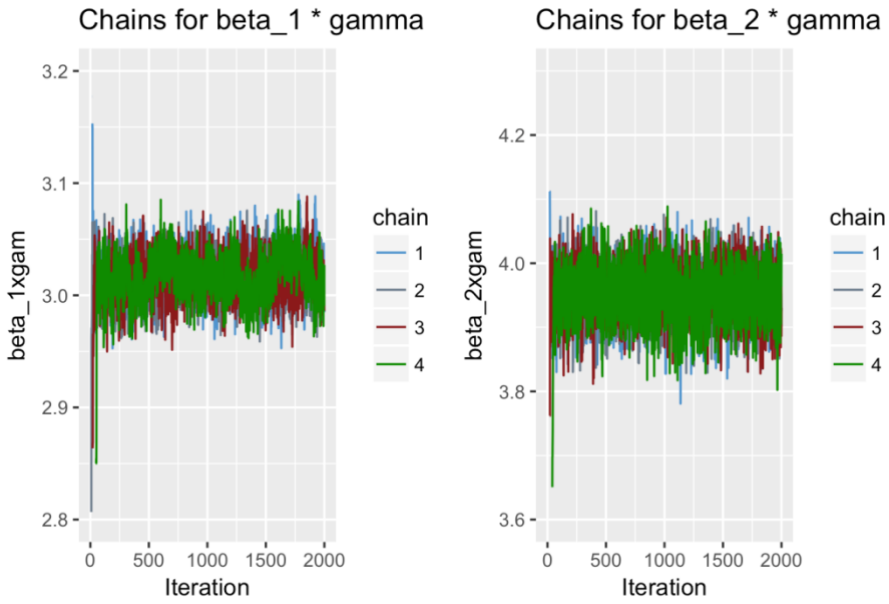


Figure 2.10: Simulations from each Markov Chain of the product between β_1 and γ (left) and the product between β_2 and γ (right) for the model from 2.13.

As we can see in this graphical representation of the chains, there has been convergence to the stationary state, since all the chains are sampling the same values of $\beta_1 * \gamma$ and $\beta_2 * \gamma$ in a steady way. Moreover, $\beta_1 * \gamma$ is sampling around 3, which is the parameter associated to x_1 in equation 2.12, while $\beta_2 * \gamma$ is sampling around 4, the other correct parameter for x_2. Therefore, thanks to the proposed method, we can conclude that even though our likelihood and posterior distribution, since the priors are uninformative, were complicated due to the introduced overparametrization, the MCMC method has been able to correctly sample from the posterior distribution and approximate the correct underlying model that has generated the data.

If we let the MCMC further sample, our predictions will not change since the model, globally, has converged even though it seems otherwise if we analyze the chains of the parameters one by one. In fact, the parameters have converged but not to a particular value due to the fact that their posterior distribution, in this case, is complicated because the parameters are highly related among them.

Example 2: Basic BNN (part 2/2)

As explained before, the likelihood associated to the model in equation 2.14 was multimodal, with the characteristic of having the modes separated by zones of low probability. In this case, the chains for each parameter by separated are the following ones (due to the parametrization in Stan, $beta[1] = \beta_0$ of the equation 2.14):

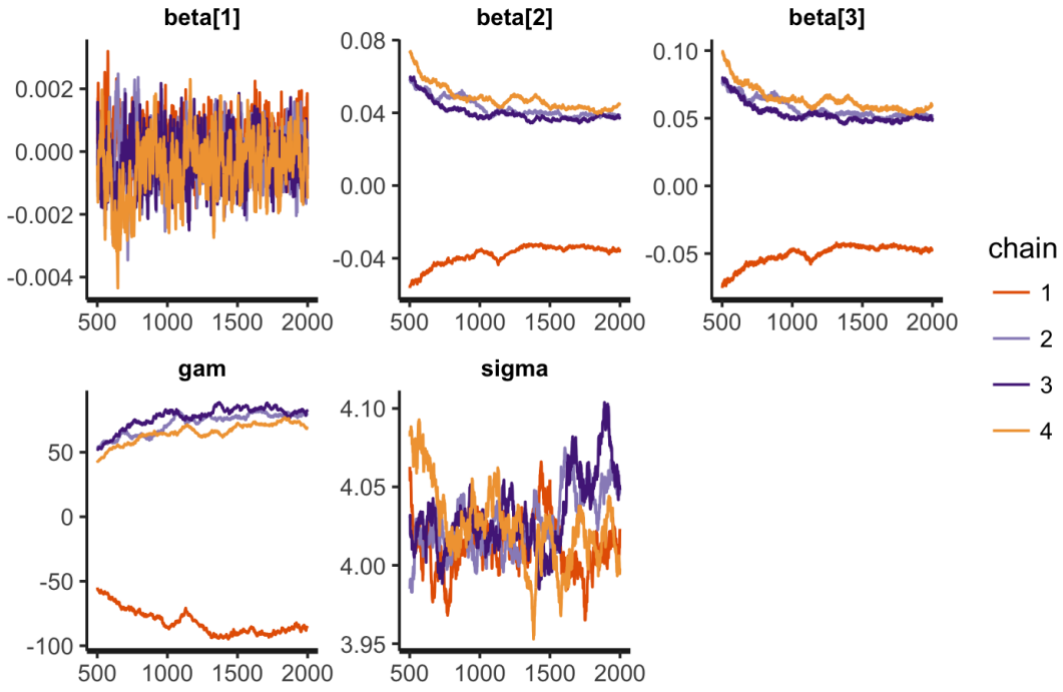


Figure 2.11: Simulations of the parameters from each Markov Chain launched to obtain the posterior distribution of the model from 2.14.

In this case it looks like there is some kind of behavior to reach a stationary state in the sense that all chains tend to sample around some values in a steady way but, of course, we can observe that for β_1 (beta[2] in the plot), β_2 (beta[3] in the plot) and γ, the chains will not meet, so one would conclude that the MCMC simulated has not been able to extract samples from the posterior distribution. According to the Multidimensional Convergence of MCMC, in order to assess if the MCMC simulations have globally converged we should use a multidimensional function of the parameters but, in comparison to the previous example, the product between $\beta_1 * \gamma$ will not be meaningful, so it is discarded.

As a reminder, our goal is to find a function that is able to show us whether our model has globally converged or not, because due to the high dependency among the parameters it is not possible to see that through the chain of each parameter. The basis of a statistical model is, clearly, the likelihood, because it directly relies on the probability distribution that we have assumed to capture the behavior of what we are measuring. Apart from the uppermost

relevance of the likelihood in the statistical model, we also know that it is a multidimensional function that uses all the parameters of the model to R, which places it as a perfect candidate for the Multidimensional Convergence of MCMC because it combines all the information of the parameters into a single-dimensioned space, in which our model summarized. On top of this suitability of the likelihood as the multidimensional function to assess the global convergence of the MCMC method, checking its evolution is reasonable because if it enters a steady state that would mean that further sampling of MCMC method is not yielding any improvement on how the model is interpreting the data observed or, in other words, that even though parameters are changing if we let MCMC further simulate (because we do not expect their simulations to become steady due to the high dependency among them), they are compensating with each other, meaning that the prediction yielded by the model is not changing and neither is the uncertainty about it.

Since the relevant part is to state whether if the likelihood becomes stable or not, instead of using the likelihood itself we will use its logarithm, since it reduces the computation time to retrieve it. Having this in mind, this is the result for this example if we use the likelihood as the function to assess convergence:

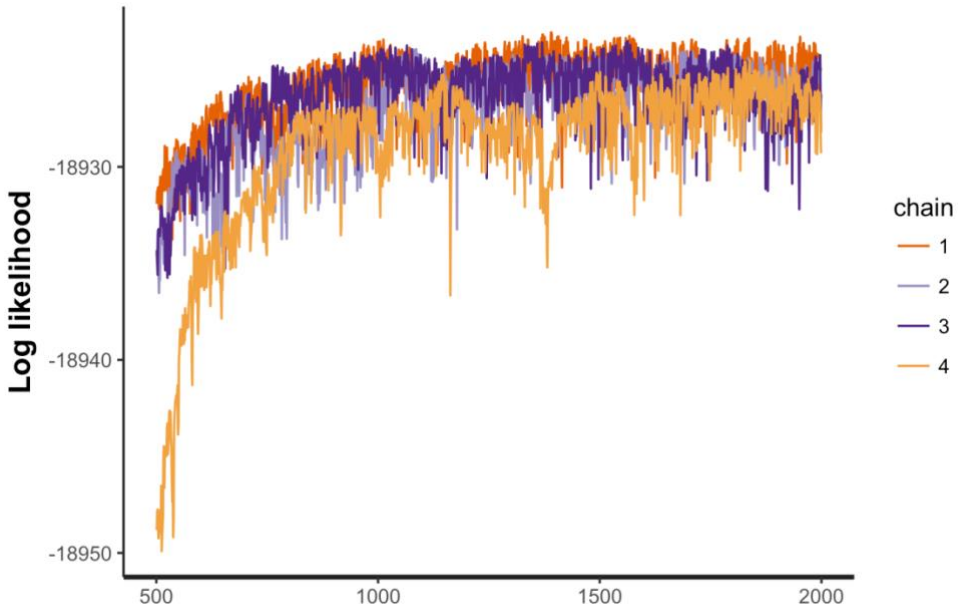


Figure 2.12: Log-likelihood chains associated to the MCMC simulations for the model from equation 2.14.

In Figure 2.12 one can see that all the diagnostics for convergence are fulfilled for the log-likelihood chain: All chains have become steady and, moreover, the values simulated by all chains converged to the same part of the space. Therefore, since we can assume that there has been convergence of the chains to its stationary state, we conclude that we are taking samples from the jointly posterior distribution. It is relevant to use the word “jointly” since in

this case, due to the dependency between the parameters, what is relevant is to extract samples from the jointly posterior distribution because the marginal distribution of each parameter will be kind of erratic (as the chains in Figure 2.11 suggest) because, each parameter by separated is not meaningful in a Neural Network. The use of the likelihood as a way to assess convergence in MCMC methods has been proposed by other authors (Robert & Casella, 2010), so we recommend delving into their contributions for argumentations from a different point of view like the one offered in this thesis.

Even though in the previous example all the chains for the log-likelihood converged to the same part of the space we know that this will not be a general situation since we know that in a BNN there are multiple modes and, moreover, the height of each mode does not have to be the same. In particular, those chains that sample around a superior local optimum (i.e. a higher mode, with better predictions) will present a higher value of the log-likelihood, while worse local optimum will have smaller values of the log-likelihood and, as a consequence, when assessing global convergence using the chain associated to the log-likelihood we can only rely on the fact that each chains becomes steady, because we cannot expect them to converge to the same values. However, in the second example, due to its obvious simplicity, each log-likelihood chain has converged to the same values because even though chain 1 has converged to different local optimum than chains 2, 3 and 4 (taking absolute values of β_1, β_2 and γ yields similar values for all the chains) the two optima are symmetric, so the log-likelihood is the same.

As a conclusion, if one employs the log-likelihood as a function to assess multidimensional convergence of MCMC methods the diagnostic used must be observing if the log-likelihood simulated values become steady and not to look for convergence among chains, because the different chains will not mix as each chain is associated to a particular local optimum. In fact, if the log-likelihood becomes steady, then the model's predictions will not improve because we have reached the jointly probability distribution and, since our goal when using a BNN is to accurately predict, with intervals, a response variable, it will not be required any further simulation of the MCMC method.

Up to this point, we have explained a proposed methodology to assess convergence of MCMC chains to a particular local mode in Bayesian models with multimodal posterior distribution. However, we must have in mind a theoretical result of MCMC chains in order to advance to the goal of obtaining a complete methodology to sample from the whole posterior distribution, instead of just a local mode. This theoretical result is that a MCMC chain has as stationary state the whole posterior distribution and, therefore, since we know that our posterior distribution has several modes, we should expect our MCMC chain to cover all the

modes in order to truly sample from the posterior. Instead, our chains just sample a reduced part of the posterior distribution, the one associated to a particular local optimum and, as a consequence, we cannot say that we are sampling from the true posterior if we use one chain.

The reason why the MCMC chains get stuck in particular modes is that the modes are separated by areas of low probability which causes the chain to consider that there is not any posterior probability out of the mode that it has found. In fact, this topic has been discussed both in (Neal, 1996) and (Müller & Rios, 1998) and the conclusions of both authors are that we need to enhance the MCMC method. Müller and Rios proposal is based on imposing constraints to the MCMC simulation and, moreover, reduce the number of modes by changing the structure of the Neural Network. However, this would mean that instead of sampling from a BNN we would be sampling from a pseudo-BNN and, furthermore, that this sampling would not be the true posterior since the established constraints would bias this posterior. On the other hand, Neal devised and analyzed several modifications of the HMC method, based on the work of Besag (Besag et al., 1995), in which some parameter is added to the simulation in order to facilitate the chains to be able of jumping from one local mode to another. Even though this would mean that we would be sampling from the posterior, the main drawback is that, in order to get enough samples from the posterior we would need a tremendous amount of time, because the chain should jump between all the modes of the posterior.

In this project we offer a new proposal, based on the results found in (Vehtari, Sarkka, & Lampinen, 2000) which demonstrated that in several applied cases, the performance of a BNN increased as the number of chains used increased. In particular, we state that, since each chain is able to sample just a part of the posterior distribution (i.e. a local mode), we could start an infinite number of chains in different parts of Θ and, as a consequence, cover all the local modes of the posterior, which would mean that if we took all the simulations from all the chains, we would have captured the whole posterior distribution. Using the parable of the blind men and an elephant, what we are stating is that if a blind person alone (a MCMC chain) that has never come across an elephant (the posterior distribution of the BNN) was required to conceptualize the elephant (capture all the posterior) by touching it, he/she would not be able to do it but, if instead of only one person, several blind person do that (using several MCMC chains) then, together, they are able to envision the elephant. In this case, each blind person (each MCMC chain) is experiencing a true part of the elephant (local optimum of the posterior) but only when the knowledge of all components is combined, the elephant takes form (the posterior distribution is correctly captured).

Even though this approach is reasonable, its full implementation is prohibitive since, obviously, it is impossible to start an infinite number of chains. Therefore, in this project we

propose starting an arbitrary large number of chains having in mind that, the more chains, the better will be the approximation to the posterior distribution. Moreover, we state that we do not need to start as many chains as optima exist because, in fact, several local optima are symmetric and by capturing just one symmetric solution, our posterior distribution is able to capture the behavior of all the symmetric solutions, since we are just focusing on the prediction yielded and not the behavior of each parameter, and all symmetric solutions give the predictions the same variability.

Even though an infinite number of chains is not required to approximate the posterior distribution, the reader could argue that starting an arbitrary large number of chains is, also, prohibitive, since each chain of the BNN has to simulate from a huge set of parameters. However, in this project we want to emphasize that nowadays, with the implacable rise of parallel computing and the easiness to build own clusters with powerful infrastructure through cloud computing services, the fact of requiring several parallel chains is not a bottleneck anymore because the chains do not depend on each other and, as a consequence, they can be started at each core of the cluster. Therefore, starting several chains is not more time consuming than just starting some chains, because in both cases the computation time will be the simulation time associated to the chain that required the maximum amount of time, so in the same interval of time we are able to approximate the posterior distribution if we start several chains.

Even though this proposed strategy, based on starting several chains and tracking their log-likelihood in order to ensure simulation of the posterior distribution of a BNN, is grounded and reasonable, there are three aspects that must be considered before implementing it. The first one is that it can be that all local optima captured by the chains are inferior local optima, meaning that we will not be able to yield the best predictions with our BNN. However, the thing is that we will never be sure if superior local optima exists and, moreover, even though they existed we would not be able to reach them, because a capable MCMC method to get passed inferiors local optima has not been developed yet. Therefore, and making use of the second-best theory (Lipsey & Lancaster, 1956), even though our MCMC simulation may not be able to reach the superior local optima, if the prediction yielded by the inferior local optima is significantly better than the one offered by the linear model, we can be satisfied about our results, because we have enhanced our baseline performance.

Another consideration that must be taken into account in BNN is that inference at parameter level is neither possible nor useful. In particular, the distribution of each parameter will be multimodal, due to the several local optima that exists, and, in addition to that, it can be that the simulations of each parameter do not converge to any specific values, because as a

consequence of the high dependency between the parameters, the marginal distribution of each parameter is not static around some particular values. In other words, each parameter by separated does not tell anything relevant and only when they are used jointly (joint posterior distribution) the model makes sense, so analyzing by separated each parameter as in the linear model is not useful in BNN.

The final aspect that we want to discuss about the proposed methodology concerns the selection of initial values for the MCMC chains. As we have explained, our goal is to let each chain sample around a local mode of the posterior distribution so, the closer that a chain is to a local mode, the less time it will need to reach and climb the mode. Therefore, we propose fitting an ANN and use its parameters' estimation as initial values for our chains, since we know that those values are associated to a local optimum and, therefore, they may be related to a mode of the posterior distribution. It is crucial to note that we say that a local optimum of an ANN is "related" to a mode of the posterior and, with this, we are stating that they are not exactly the same. In particular, the objective function of ANN is not the likelihood, because it usually includes regularization terms, so the initial values that we provide for the MCMC chain will not be at the center of a posterior's mode, but they are located in a way that reaching the posterior mode is easier for the chain.

As a conclusion, applying this technique for the initial values we avoid problems like providing incorrect initials for the chains and, moreover, we are able to speed up the convergence of the chains. In (Vehtari, Sarkka, & Lampinen, 2000), several methods to provide suitable initial values are discussed and, on top of that, a method is proposed based on early stopping ANNs in which the initial values for the BNN are the value that the ANNs parameters take in the iteration in which the error in the validation set does not reduce. Therefore, our proposal is similar to (Vehtari, Sarkka, & Lampinen, 2000) contribution but without relying on a validation set because we take the local optimal set of parameters of the ANN in the training set.

6.2.3 Using Design of Experiments to choose a suitable architecture

During all this subsection we have explained how to sample from the posterior distribution in a BNN with a given architecture but, of course, we still need to decide the architecture of our BNN. According to (MacKay, 1991a) and (MacKay, 1991b), thanks to the Bayesian approach we are able to choose between architectures because we can obtain the posterior probability associated to each architecture by using the formula described in equation 2.3, where each model is a particular architecture.

Even though this would be the most sensible approach for architecture selection in a BNN, there are some limitations in its implementation that causes this approach to yield misleading results, as it has been explained in the subsection 5.4 (*Bayesian Neural Networks history and literature*). Moreover, trying to estimate the posterior probabilities associated to each model (i.e. architecture in the case of BNN) through MCMC is still a developing area and, even though there are some implementations of it for simple Bayesian models, all of them can be very time consuming (Miazhynskaia & Dorffner, 2006) and they are not guaranteed of yielding an accurate estimate due to its dependency to the prior distribution.

In order to facilitate the implementation of BNN in real cases, we will decide its architecture through an empirical method, which means that we will test several architectures and decide according the results obtained, as it is currently done with ANNs. The most widely used methodology is k-fold cross-validation, in which one must separate his/her training data into k parts and, for each architecture that is being tested, fit k ANNs. In particular, each of the k ANNs use $k-1$ partitions of the training data to estimate the parameters and uses the remaining partition as validation set in which a metric about the predictive performance, based on an error function is collected. Afterwards, for each architecture that is being tested we compute the average of its k error metrics and decide that the best architecture is the one that presents the lowest average error. Therefore, in order to apply cross-validation to BNNs, we would need to specify a set of architectures, with a cardinality of A and, afterwards fit k BNNs to each architecture, which means that a total of $k * A$ BNNs should be fit in order to find a suitable architecture. However, since the MCMC simulation time for a BNN is high, fitting all these BNNs is prohibitive and, therefore, we will use ANN to find the optimal architecture for a BNN.

Even though k-fold cross-validation has clear advantages, like the easiness of its implementation and ability to yield satisfactory results, it is also true that it has some disadvantages and, from a statistical point of view, the most important one is that there is not an efficient use of the information which causes cross-validation to offer unstable solutions which are not fully reliable. The main problem associated to k-fold cross-validation is that the average error metric associated to each architecture is just a punctual prediction of the error for that architecture and, obviously, it has some variability. In particular, the variability of this average comes from two sources. The first one is that there are differences between the data stored in each of the k folds, so the k error metrics that are used to calculate the average error are not equal. The second reason, which is even more important, is that in order to obtain the k error metrics for each architecture, we need to fit k ANN and, since ANN are only able to find local optima, this error metric has variability depending on the local optimum that the ANN has found.

Therefore, since there is variability in the average error associated to each architecture, we cannot take a decision about which is the best architecture just by selecting the one that has a smaller average error because we know that, due to this variability, it could be that if we repeat the cross-validation process another architecture may be better than the previously chosen as the best one.

Before following this explanation, there is a terminology confusion between Bayesian Statistics and Machine Learning that must be solved. In particular, both fields employ the concept "hyperparameter" but they give a different meaning to it. For Machine Learning, a hyperparameter is a parameter of the error function or iterative optimization algorithm that is fixed through k-fold cross-validation or any other method and that does not change during the optimization (i.e. during the estimate of the weights and biases). Therefore, for the case of ANN, this would be the regularization constant or the mutable elements that defined the architecture (i.e. number of hidden layers, number of nodes per layer and activation function per layer), since the optimization algorithm is only applied once these elements have been fixed. On the other hand, for Bayesian Statistics, an hyperparameter is a parameter that characterizes a prior distribution and, like any other parameter, has a posterior probability distribution. For the first type of hyperparameters we will use the concept ML-hyperparameter, while for the second one we will use Bayesian-hyperparameter.

In our proposed BNN, we will have to deal with those two types of hyperparameters, since we need to fix the architecture according to some mutable elements (those elements are ML-hyperparameters) and, at the same time, as it happens in the subsection 6.3.2 (*Regularization and Automatic Relevance Determination through hierarchical models*), we can create hierarchical models in which the priors are ruled by common hyperparameters (Bayesian-hyperparameter). In fact, this existence of the two types of hyperparameters is latent in basic statistical models such as a linear model since in it there are some decisions (ML-hyperparameters) that must be fixed before finding the MLE: The distribution that will be used for the response variable and the model matrix (i.e. polynomial degree and interaction level considered). The main difference is that statistical science has been able to provide grounded solutions (like information criteria or hypotheses test) to select those ML-hyperparameters for those simplistic models and, therefore, there is no need to employ methods like k-fold cross-validation to fix them.

Going back to our discussion about cross-validation, another relevant shortcoming of cross-validation is that, since the architecture of the ANN depends on several ML-hyperparameters (number of hidden layers, number of nodes per layer and activation function per layer), it is very difficult to assess what is causing the error function to be low in a particular architecture

and, therefore, it is complicated to explore a new region of ML-hyperparameters' value in which the error function could be even lower.

Due to these problems that cross-validation has when selecting the ML-hyperparameters of an ANN, we propose a different method in which we can be more confident about the selected optimum and, moreover, we can conduct successive exploration stages in a more grounded way. This proposed method is based on Design of Experiments (DoE) because we assume that the predictive performance of our ANN has many causes of variability and, through experiments, we will assess the effect of some explanatory variables (i.e. the architecture ML-hyperparameters) on it so we can optimize the predictive performance.

In order to explain the use of DoE for architecture selection, we will use a simile with a classical example used to explain DoE: creating the best paper helicopter (Zahraee et al., 2013). The first step required in DoE is to translate the goal of the study into some response variable able to be measured. In the case of the helicopters, the best will be the one that flies for a longer period of time, so our response variable will be "Time flying". On the other hand, the response variable in the case of architecture selection must be something related to the predictive performance which will be, obviously, some kind of error metric. If we apply the most general type of error, the response variable able to capture our DoE goal will be "Root Mean Squared Error" (RMSE) of the ANN.

Once we have defined the response variable (Y), the second step consists in defining a set of factors or explanatory variables (X) candidates of having some effect on Y. In the case of the helicopters, one should consider some variables like length and width of the helicopter's rotor, height of the helicopter's body… but, in the case of the architecture selection the decision is straightforward since the variables that we know that have some kind of effect are the ML-hyperparameters: number of hidden layers, number of nodes per hidden layer, activation function per layer and, finally, some ML-hyperparameters associated to the regularization constant such as the learning rate (regularization is intrinsic in a BNN, so the learning rate only exists for ANN).

Even though those X are the main variables (reasons) that explain the variability of the response variable, we know that there is a large set of small untraceable reasons that cause the response variable to be different between helicopters or ANNs. In other words, if we build two helicopters (ANNs) with a particular rotor length and width (architecture), we know that they will not present the same value for the response variable. In the case of helicopters this variability is due to small reasons such as how the fold has been made, the time that has passed until the paper has been used, the ability of the person manufacturing it… while in the

ANN it depends also on small reasons: the exact values for each variable that present each observation of the training data. Obviously, each observation's values are traceable because we have the dataset but what is untraceable is their effect in the iterative optimization algorithm used to estimate the ANN's parameters and, therefore, their final effect on the predictive performance associated to the estimated parameters. In a summarized way, if we use 100 observations of the underlying population to fit an ANN with a particular architecture A and, afterwards we fit another ANN with the same architecture A but with some other 100 observations, the predictive performance (response variable) will not be the same, because the estimated parameters will not be same.

The third step is to define a range of values for those variables that describe the traceable reasons of variability, i.e. a range for all the Xs. For the number of hidden layers, in this project, we propose experimenting with two or three hidden layers. The rationale behind this decision is that we want our last hidden layer to only have two nodes, because we will analyze those two nodes as latent variables in order to understand our Neural Network. Therefore, one-hidden-layer ANN are discarded, because then they would only have two nodes in the hidden layer and they could badly approximate the relationship between the predictors and the response variable (we mean the response variable of the ANN, like a transaction price). Apart from that, we recommend fixing the structure to a maximum of three hidden layers because, as many authors suggest, they overcomplicate the problem without yielding any clear performance improvement (Heaton, 2008). For the activation function of each layer we have decided, for simplicity, to use a unique activation function in all the hidden layers and, moreover, we have decided that those activation function will be either *logistic* or *tanh*, since they are the typical ones. For the number of nodes per hidden layer we allow some flexibility, even though we recommend working with a small number of nodes, especially in the first hidden layer, because if not MCMC simulation time can become prohibitive. On top of that, we restrict the last hidden layer to have only two nodes, since this will allow us to analyze the latent variables of the Neural Network. Finally, for the learning rate, since it does not exist in BNN we let the user fix them at any level he/she thinks it might be useful.

The fourth step consists in designing useful combinations of the Xs to conduct experiments in order to quantify the effect of each traceable reason on the response variable, knowing that the response variable is influenced by the untraceable reasons of variability. In the case of the helicopter, each experiment refers to taking a new paper and conducting the manufacturing process according to the value of the rotor's width and length (Xs) with which we want to experiment. In the case of the ANN, the paper is the underlying observations used to fit the ANN and the manufacturing process is the optimization procedure, which is made according to the given ML-hyperparameters (Xs) with which we are experimenting.

In DoE, each tested combination of the Xs is called an Experimental Condition (EC) and, since in the case of assessing a suitable ANN architecture the cost associated to each experiment is small as it is only the optimization time to fit the ANN, we will be able to take replicates of each EC. Apart from this consideration, we still need to decide the specific design and, since we know the function that links the predictive performance of an ANN and its ML-hyperparameters must be somehow complicated, we decided to apply a type of design able to capture nonlinearities from the beginning. Since we assume, from the beginning, that there will be nonlinearities in that function, we will use a Box-Behnken Design, instead of conducting a two-step Central Composite Design. Reviewing all those types of designs and further explaining the purpose of DoE would excessively extend the scope of this thesis and, therefore, we recommend the reader to refer to (Box G. E., 2005) in order to become acquainted with these concepts. However, in order to take a glance at the rationale of a Box-Behnken design as a way to select ML-hyperparameters in an ANN, a simplistic example in which the corresponding design matrix is calculated is shown in the annex (section *Example Box-Behnken design matrix* of the annex).

Up until this point we have described the different EC (i.e. architectures) that we will test, which are defined by the Box-Behnken design and, also, we have said that for EC we will aim to have replicates, since we know that our experiments are cheap. The next and final step would be conducting the experiments per se, which requires an abstraction in the case of the ANN. In particular, we want an architecture that is able to generalize the behavior of what we are observing for all possible data and not just the training data and, moreover, we know that for each experiment all the untraceable reasons of variability must affect because, if not, the noise would be reduced and our model, based on stochastic independence of the observations, would not be correct because we would not have a simple random sample.

In the case of a helicopter, in order to conduct an independent experiment, one should, as explained before, take a new paper and carry out the manufacturing process according to the variables of the EC because, acting this way, we let all the untraceable reasons of variability affect the response variable. Since we need to take a new paper for helicopters, we need to take different underlying observations for each ANN experiment because, if not, we would reduce the effect of all untraceable reasons of variability. In order to do that, we propose taking a bootstrap resample of the training data and, with that resample, fit the ANN associated to the EC. By acting this way, we are also generalizing our results for a larger sample than the training set because we are assuming that the empirical multidimensional distribution of the training data is an estimator of the true probability distribution that has generated the individuals that we observe. Therefore, by taking a resample of the training data for each experiment, we are estimating, with the information of the training set, a

possible sample from the population so we are trying to generalize the results of the predictive performance associated to each architecture and, moreover, we endow all experiments with the sufficient noise (untraceable reasons' effect) to be compared with each other. If instead of taking a resample of the observations we took the whole training data at each ANN that we fit, then it would be like creating repetitions instead of replicates, because some sources of variability would be fixed.

Therefore, the designed process would be that, for each experiment we take a resample of the training set and fit, with that resample, an ANN according to the ML-hyperparameters of the EC that we are testing. After having fitted the ANN, since we want to track the out-of-sample predictive performance, we predict with the fitted ANN a value for the training observations that were not included in the resample and, afterwards, we compute the RMSE of that prediction.

In order to summarize all the process of using DoE to find a suitable architecture for ANNs a representation of the process is shown in Figure 2.13, where the subindex u refers to the underlying observations of the ANNs, so Y_u would be the transaction price in our case while Y is the RMSE of a prediction. Furthermore, in this representation b refers to a particular resample of the training data and b^c refers to those observations of the training data not included in b.

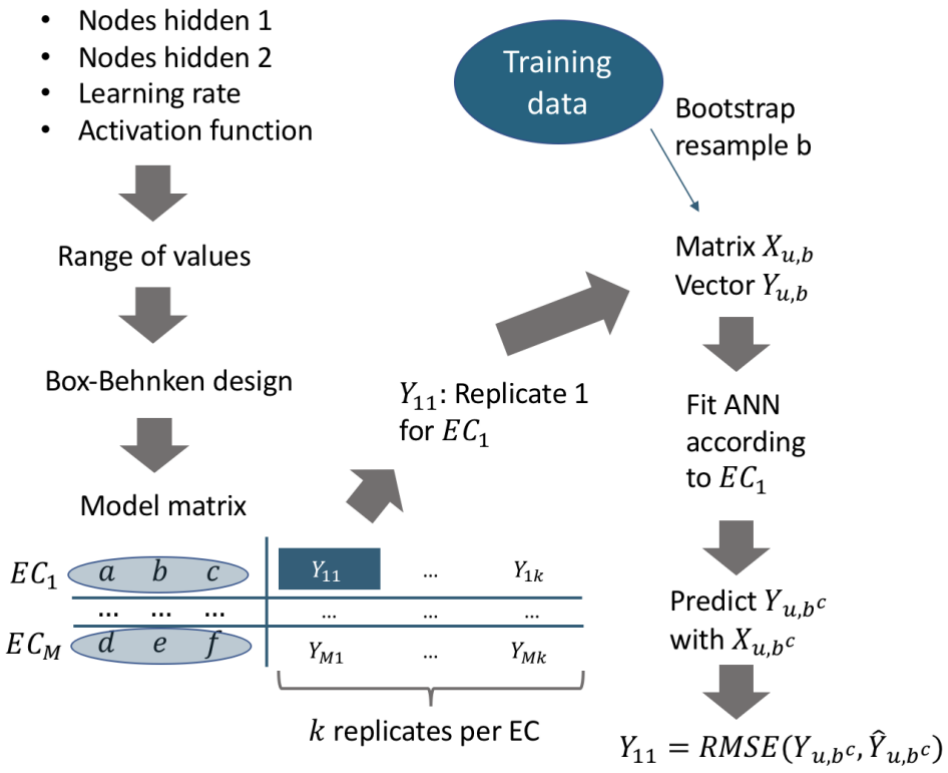


Figure 2.13: Schema summarizing the proposed methodology based on DoE to select an architecture.

Once we have conducted all the experiments, all that is left to do is obtain the linear model that best explains the results obtained in the experiments and, finally, take conclusions according to it. Since we are able to fit a linear model up to two-way interactions and quadratic terms, we would be able to detect optima around the ML-hyperparameters' values with which we have experimented, using the equation estimated by the linear model. Moreover, we will be able to capture the effect of each ML-hyperparameter on the predictive performance of the ANN and, therefore, we will be able to determine a new region, i.e. new architectures, in which we expect to find even better predictive performance.

In order to find the best linear model several techniques can be used, and, in this project, we have decided to work with the Bayesian Information Criteria (BIC). In the case that the best model was the null one, the only thing required is to increase the number of replicates per EC, because that will reduce the noise of the model and, therefore, allow for small-valued coefficients to enter the model.

As a final conclusion, using DoE to fix the ANN's architecture allow us to conduct a more cost-effective procedure in which the amount of information retrieved by each experiment is higher than in cross-validation. The basic idea is that DoE uses interpolation in order to avoid conducting as many experiments as in cross-validation and, with less experiments (ANNs) is able to yield more confident results.

6.2.4 Designed pipeline of our methodology to fit the proposed Bayesian Neural Network

According to everything that has been explained in this subsection, the designed pipeline to fit the proposed BNN to a particular problem would be the following:

1. Data preprocessing and partition of the dataset into training, validation and test: Even though a validation set is not required to find a suitable architecture for the BNN, we still need the validation set to compare the predictive performance between the linear model and the BNN because, depending on the data, it can be that the linear model outperforms the BNN. In all our examples the BNN has been superior than the linear model but, for those cases in which the BNN is just slightly better than the linear model, the user may be more confident to work with the linear model and, through the validation set he/she can assess whether if it is useful to select the BNN or not. Finally, the test set will be used to assess an honest estimate of the out-of-sample prediction error, since the validation set was only used to compare and select between the BNN and the LM.

2. Standardize both the predictors and the response variable: This is a recommended practice for ANNs and, for BNNs, we also recommend it because it simplifies the MCMC simulation since the covariance matrix becomes simpler. As a reminder, the standardization must be done with the information contained in the training set used to estimate the parameters.
3. Define a range for the ML-hyperparameters of the ANN: According to what has been explained in the previous part of this subsection, this would mean defining a range for the learning rate, the number of nodes in the first and in the second hidden layer, being 0 the lower bound for this last ML-hyperparameter.
4. Conduct the proposed DoE methodology to ascertain a suitable ANNs architecture. In particular, we recommend performing more than one experimentation stage.
5. Fit C ANNs with the chosen architecture, being C the number of MCMC chains that you will start to sample from the jointly posterior distribution. The higher the C the more modes of the posterior distribution will be captured and the better will be the uncertainty given to the predictions.
6. Store the weights of the C ANNs and use them as initial values ot start C MCMC chains: We also need a starting value for σ in our MCMC chain and, in order to obtain it, we use the standard deviation of the training squared errors yield by the ANN.
7. Conduct multiple MCMC simulations and discard those in which the log-likelihood associated to the chains has not become steady yet: Only the ones in which the log-likelihood is stable are the ones from the posterior distribution.
8. With the remaining simulations, compute the predictive posterior distribution of the response variable for each individual in order to yield both a punctual and interval-based prediction. In a summarized way, we have M valid simulations of θ so, for each $\theta^{(m)}$ we simulate $\mu^{(m)}$ for each individual and we extract a random draw from a Normal distribution with $\mu^{(m)}$ and $\sigma^{(m)}$. Therefore, for each individual we have a vector of simulations of Y and, by taking the mean or the median we would give a punctual prediction while by taking the 0.025 and 0.975 percentiles we would obtain the 95% probability interval of the prediction.

In this thesis, we have proposed a complete methodology to fit a BNN which includes lots of decisions at many stages. In order to take those decisions (like the initials values, the number of chains to start, how to assess convergence, the use of prior information…) in a grounded way, we have conducted a large set of experiments in several datasets. In particular, we started trying our proposal in simulated datasets, then we applied it to the Fisher's iris dataset and, finally, to another real dataset currently hosted by Kaggle[5]. However, reproducing all the

[5] https://www.kaggle.com/mirichoi0218/insurance/data

experiments and including them in this project would be too extensive, misleading and confusing and, as a consequence, we have decided to reproduce only the most important ones through this dataset hosted by Kaggle[4]. These experiments can be found in the annex (section *Summary of the most relevant results of our methodology through an example dataset* of the annex).

6.2.5 Validation procedure for the proposed Bayesian Neural Network

Just like in a linear model, once we have obtained our BNN we need to conduct an ad hoc analysis to determine whether if the model premises are being held and, therefore, if our model is correct and can be used to perform inference.

The first required check that must be conducted is the one concerning the convergence of the MCMC chains. In particular, we need to confirm that the simulations of θ that we are using come from chains that have reached a steady state in their log-likelihood. If not, we would not be sampling from the posterior distribution and, therefore, our predictions and uncertainty around the predictions would not be valid. In fact, checking MCMC convergence is a general validation procedure in Bayesian Statistics and, in this project, the only thing that is different is how we assess this convergence.

The second type of checks are the ones related to the particular probability distribution that we have assumed for our response variable. In particular, one should analyze if the model residuals behave as a normal distribution and, moreover, if there is some hint that indicates whether if the assumed homoscedasticity is not true.

Finally, in our examples we compared the intervals offered by the linear model with the intervals offered by our proposed BNN because we wanted to check whether if the results were sensible or not. In particular, we computed the 95% probability interval of the response variable for each individual for both the linear model and the BNN and, afterwards, we represented them in a bivariate scatterplot to check if the BNN uncertainty is similar to the offered by the linear model. Even though this is not a required validation step, because we believe that our BNN is able to correctly fit several datasets as it has done in all our examples, we encourage the user to conduct this uncertainty validation in his/her own project, because it will increase the confidence about the proposed BNN.

6.3 Going Beyond: Extensions of the proposed Bayesian Neural Network

6.3.1 Non-compliance with the basic hypothesis of the model

Just like in linear models, it can be that after conducting the validation procedure, the normality and homoscedasticity hypotheses do not hold and, therefore, our model's inference would be invalidated. However, since we are in the Bayesian framework, it is easy to adapt our model and overcome these obstacles.

In particular, if the model's noise is not Gaussian, then we can assume a different probability distribution for the response variable which would need to be validated, also, through the residuals. The key point is that MCMC will be able to simulate according to this new distribution and, therefore, obtain the probability distribution without requiring any special calculations from the user, despite the fact of introducing the new log-likelihood to assess convergence of the chains. In a similar way, if the homoscedasticity premise does not hold, the user could easily specify a particular function relating the model variability parameter σ with the predictors and let the MCMC simulation do the rest.

6.3.2 Regularization and Automatic Relevance Determination through hierarchical models

As it has been explained during the revision of the literature on BNN, MacKay envisioned a particular BNN in which there was a parameter associated to the regularization of the weights similar to the case of ANN. In particular, he demonstrated that if we build a BNN in which all the weights and biases follow a normal distribution centered at zero and with a Bayesian-hyperparameter α as their deviation, then this parameter would be the same regularization constant than in ANN and, moreover, we would obtain a posterior distribution over it that would tell us the best value for this Bayesian-hyperparameter.

In the first version of our proposed BNN we did not specify any normal prior distribution for the weights and biases and, therefore, even though regularization exists because it is automatically captured in the posterior distribution (meaning that it will avoid large values for the weights), it is not harnessed through an explicit Bayesian-hyperparameter α, but rather diluted into all the parameters. The main reason why we avoided this regularization formulation is that, since we are empowering our BNN to handle several local optima, we believe that forcing our weights and biases to follow Gaussian distributions is limiting the capabilities of the posterior distribution and, therefore, causing the model to yield worse performance.

After MacKay's first proposal to explicitly capture regularization, other authors, mainly Neal, Buntine and Weigend and MacKay himself, proposed other approaches to explicitly capture regularization in a more specific way. In particular, the next proposal was based on the fact that, instead of just having one regularization constant, a different regularization could exist for the weights of each layer, allowing more flexibility to the Neural Network. Obviously, assessing a different regularization constant per layer is computationally demanding in ANN because the amount of cross-validation experiments to obtain a suitable value for each of them would be too high. Instead, in the BNN envisioned by those authors, if we allow the weights and biases of each layer to follow a Gaussian distribution centered at zero and with deviation α_l (each layer has a different α_l) then the vector of Bayesian-hyperparameters α would be the regularization constants associated to each layer.

Even though these extensions based on finding explicit Bayesian-hyperparameters to resemble regularization in ANN seems interesting, we did not implement them in our BNN, since as we explained, our goal is to propose BNN as a parametric statistical model and, therefore, there is no need to establish explicit regularization. In particular, we argue that since the weight estimates will be obtained through the posterior distribution instead of an iterative optimization algorithm, there is no need to input any regularization, since the posterior distribution automatically penalizes overfitted estimations of the Neural Networks weights. The main argument that sustains this rationale that in a parametric statistical model there is no need for explicit regularization (i.e. parameter penalization) because it is automatically included is how the statistical linear model works compared to Ridge Regression. In particular, if we fit a linear model with *significant* predictors on a response variable using the Bayesian approach, i.e. obtaining the posterior distribution, the parameters' estimate will be the one that generalize the most and we will be endowed with non-overfitted parameters estimate. However, if we apply an iterative optimization algorithm on some error function using the training observations, the set of parameters of the linear model is usually an overfitted solution and that is why regularization needs to be included (Ridge Regression). With this we do not argue than in BNN there will not be any overfitting, because it is obvious that this will come from the architecture structure, but at least, it is not as relevant as in ANN.

Apart from these applications that aim to explicitly capture regularization, both Neal and MacKay used hierarchical Bayesian models in their BNN in order to capture the importance that each input has when predicting the response variable. In a simplified way, they devised a hierarchical model in which, the weights that connect the input layer with the first hidden layer follow a Gaussian distribution centered at zero and a particular deviation σ_j, where j refers to the node input j. Therefore, this model states that there is a Bayesian-hyperparameter σ_j associated to each input node that regulates the width of the weights that

come from this input j and, as a consequence, both Neal and MacKay state that the greater the σ_j, the greater the effect of that particular input when predicting the response variable, because σ_j is allowing its weight to take huge values. This technique was called Automatic Relevance Determination (ARD) and, in contrast to the previous applications with explicit regularization, we adapted it to be introduced in our BNN. In particular, we established an improper uninformative hyperprior for all σ_j Bayesian-hyperparameters because we wanted to maintain the idea of avoiding too much prior information.

Since σ_j is related to the units of the predictor, it will be necessary to work with standardized predictors, as we have been doing for BNN without ARD. Once we have standardized a numeric predictor, its unit of measurement is "standard deviation from the average value" so all the σ_j from numeric predictors will be comparable. However, there are two reasons why we will not be able to compare the relevance of numeric predictors with the relevance of categorical predictors. The first one is that, while a numeric predictor is harnessed through a single input node and, therefore, it has a straightforward Bayesian-hyperparameter σ_j that captures its relevance, the relevance of a categorical predictor is diluted into $q - 1$ Bayesian-hyperparameters, being q the number of modalities of the variable. Moreover, the second reason is that the inputs for the categorical variables will be the dummy variables created according to the baseline category and, as a consequence they do not have any unit of measurement, because they capture a qualitative characteristic rather than a quantitative one. Therefore, standardization of those variables would not be meaningful and, as a consequence, their relevance will not be able to be compared with the one obtained for the numeric predictors.

The main advantage of assessing the importance of each input through this methodology called Automatic Relevance Determination (ARD) is the fact that it includes all the possible interactions with the other inputs and, moreover, that it automatically takes into account nonlinearities with the response variable. In this project, we have adapted it to our proposed BNN and we observed that even though, of course, we could extract the relevance of each input, but we have not been able to determine if it reduces or it increases the predictive performance with respect to the non-hierarchical BNN (i.e. the first one that we proposed without ARD). In particular, in some cases the predictive performance diminished, like in the example attached at the annex (*Automatic Relevance Determination* from section *Summary of the most relevant results of our methodology through an example dataset*), which aims to summarize all the results that justify our proposed BNN and its extensions. However, in the example provided in the third chapter of this thesis, the predictive performance is enhanced when we introduced ARD to our BNN.

6.3.3 Interpretation Layer

During the subsection 6.2.3 (*Using Design of Experiments to choose a suitable architecture*) we have explained that we limited our last hidden layer to only two nodes, because we wanted to harness all the BNN prediction through only two latent variables. The main idea is that those two latent variables are the "best latent variables" for the response variable, in the sense that they capture all the information from the predictors and with just a linear combination on them we can obtain an accurate prediction for the response variable. We propose extracting their relationship with the predictors to be able to envision what is driving the prediction.

However, depending on the particular local optimum, different latent variables are obtained and, as a consequence, we will not be able to use all simulations from the MCMC because each chain is associated to a local optimum. Therefore, we have decided to work only with the chain that presents a higher log-likelihood because, even though we know that it will not explain all the prediction procedure of the BNN, we will be able to, at least, take a look at the role that the predictors may have.

There are two graphical representations of these latent variables that can be useful to understand our BNN. The first one is based on representing all the individuals. The main idea is that those two latent variables (lv_1 and lv_2) have some variability between the individuals (depending on the value of their inputs) and within the individuals, because lv_1 and lv_2 are a result of using some parameters θ that follow a probability distribution. Therefore, in our plane we can represent a central value for each individual and, thanks to the variability of θ in the posterior distribution, we can also plot circles of probability around that central value by plotting all the simulated values of each individual for the variables lv_1 and lv_2. As a consequence, we could assess which individuals are similar and which are different between them, allowing us to build clusters of individuals and extracting profiles related to the response variable.

Even though this graphical representation seems useful, the main problem associated with it is that, since we will plot all simulations for a high number of individuals, the plot will become too crowded and we will not be able to visualize relevant aspects. Therefore, we propose a second plot in which we will represent some kind of summarization of the relationship between each predictor and the latent variables.

For numeric variables we propose using the Pearson's correlation coefficient, which means that, since we have M simulations (because we have M simulations of the posterior distribution of θ) of lv_1, we would compute M times the Pearson's correlation coefficient

between the numeric predictor and the variable lv_1. And, of course, we would do the same with the variable lv_2. After all these correlation coefficients are obtained, we could plot them and see which are the predictors more correlated to each of the latent variables with the goal of endowing the latent variable with some meaning. However, the Pearson's correlation coefficient only captures linear relationship between the two kinds of variables so the true relationship between the predictors and latent variables is not well represented. Therefore, we propose to use some other statistic like Kendall's τ to just state if some predictor is clearly associated to a latent variable or not but, of course, obtaining some meaning for a latent variable with this is much more complicated than with Pearson's coefficient.

For categorical variables we propose computing, for each category, the M average values of lv_1 and lv_2 that the individuals of that category take. If we plot all these values in a scatterplot we will be able to see in which part of the space is associated each category and, moreover, since each category will be endowed with some circles of probability, we will be able to decide which categories are different to each other according to these latent variables.

As a conclusion, the idea is to conduct some useful representation of all the predictors in order to give some semantic to the latent variables that are driving our prediction. If we accomplish that, we will able to understand the relevance of each variable in our BNN and, furthermore, extract which are the concepts (latent variable) that better explain the response variable.

Chapter III: Bayesian Neural Networks as a pricing model for Airbnb in Barcelona

This chapter is the natural combination of everything that has been explained in the previous two chapters, since we use an example to show the utility of implementing a pricing model for P2P OM, as explained in the first chapter, and also how a BNN outperforms a linear model in some cases, as explained in the second chapter.

7. Motivation to use information about Airbnb

In order to see how the proposed BNN should be used as a price recommender for a P2P OM we have decided to implement it on Airbnb because it is one of the most famous worldwide P2P OM and, therefore, we think that it will be easier to interpret it if the reader has some knowledge about it.

Moreover, since Airbnb is having a considerable impact on the housing market several initiatives have been instituted in order to collect data and conduct studies about it and, therefore, we have decided to use Airbnb because of the easiness to retrieve relevant data about it. In particular, we have used the information from the website insideairbnb.com[6], which regularly launches a web-scraping algorithm to retrieve information about all the published apartments and rooms in Airbnb.

Even though Airbnb is having a significant impact all over the globe, its effect is specially focused on some cities like Barcelona, in which it is completely shaping the touristic model and, therefore, having a huge impact on all related markets and the daily life of its citizens. Therefore, and having in mind the technological infrastructure constraints of this project, we have decided to focus our analysis for Airbnb in Barcelona and, moreover, to use all the listings from the last scraping wave of insideairbnb.com, which was on 7th February 2018[7].

As a conclusion, we have decided to use Airbnb in Barcelona because of technological infrastructure constraints, because it is a well-known platform in the city and, moreover, because several valuable transactions are conducted through it, meaning that an

[6] http://insideairbnb.com/get-the-data.html

[7] Explicitly, the last scraping wave was from 9th June 2018 but, due to the deadline of this project we instituted the project before that date and, therefore, the analyzed one has been the wave from 7th February 2018.

improvement of this platform such as the pricing the model that we suggest would have a relevant impact on society.

8. Fitting the proposed Bayesian Neural Network and validation procedure

As a reminder, the BNN should be fitted using as observations all the conducted transactions within an established period of time because, with that, we could use the real market price (i.e. price at which demand and supply meet) and variables from the product features (i.e. characteristics of the apartment) and the transaction setting (i.e. seasonality). Because of that, we propose P2P OM, like Airbnb, and not traditional markets to implement this pricing model because only P2P OM have the transaction-level data needed to build the pricing model.

However, in this thesis, we do not have access to transaction data from Airbnb; we are only endowed with all the published apartments on a particular date (7th February 2018). Therefore, we will need to handle and adapt that information in order to approximate the most the transaction data that we would need to build the pricing model. All this adaptation is explained during the data preprocessing subsection in which, besides this, we transform some predictors in order to be more understandable and, moreover, we deal with missing values and possible outliers.

8.1 Data preprocessing

8.1.1 Adaptation of the dataset to approximate conducted transactions

As explained before, our dataset only contains all supply of Airbnb listings in Barcelona on 7th February 2018, which is a total of 18,531 apartments and rooms. However, there is a part of these suppliers that are placed out of the market, meaning that they fixed a price for their apartment that does not meet the demand's preferences and, therefore, they do not receive clients. Our goal is to erase those listings placed out of the market because, if not, the predicted price by our BNN will be too high because of these observations and, therefore, it will not be able to estimate the true market price, i.e. the price at which transactions are being conducted and, as a final consequence, the user will not be recommended a correct price for his/her apartment.

In order to erase those out-of-market suppliers, we decided to use a two-step procedure that uses two different variables collected in the dataset.

The first step consists in using the variables that tell us how many days in the following 30, 60, 90 and 365 days the apartment is available. Our rationale is that if an apartment has a lot of available dates during the following days means that it is not finding any clients because it does not meet their preferences, so it has an incorrect value for money. In particular, we decided that those apartments with more than 85 available days in the following 90 days will be considered as suppliers that do not interfere in the market price at which transactions are being conducted (i.e. the one that we aim to predict) because they are not able to collect clients. The decision to use the availability in the following 90 days is because using the availability in the 365 days would be misleading, since many apartments will have lots of available dates not because they do not meet the demand's preferences, but because the demand has not looked for it yet since the holidays are usually not programmed a year in advance. On the other hand, using the variables of 30 and 60 days could also be misleading, since it would be affected, a lot, by seasonality, meaning that apartments that start selling for the next session would present a large number of available dates, causing us to wrongly consider them as if they were placed out of the market.

With this first step we removed 2,845 listings but we need a second step because the variables about the availability include two different concepts. On the one hand, they reflect the demand preferences on each apartment as explained before but, on the other hand, it can be also that the supplier reduces the availability of its apartment maybe because he/she is not available to welcome new guests or because he will be using the apartment. If the second case applies, the number of available days in the next 90 days will be small and, according to what has been explained in the first step, we would wrongly consider that it is happening because it fits the demand preferences.

Therefore, in this second step, we decided to use a recorded variable which tracks the last time that the user modified something about the listing. In particular, we computed the days without updating the listing and we decided to remove those in which the user has been more than 180 days without modifying anything. With this, we aim to remove those suppliers that have a small availability of days because, even though the listing is published on Airbnb, the user is not interested in receiving any guests. If we did not implement this second step, then a lot of apartments that are out of the market because the supplier does not want to make them available would be wrongly considered as if they were part of the effective market. As a result of this second step, we removed from our dataset 1,466 more observations, so the final number of listings used will be 14,220.

This two-step process to adapt the dataset can be summarized with the following plot:

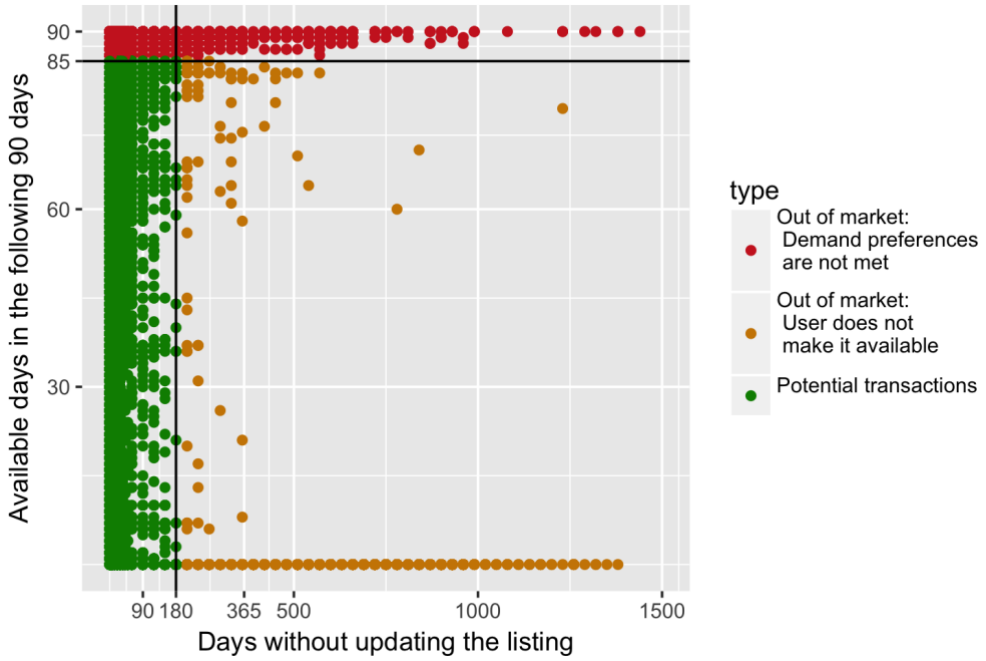


Figure 3.1: Representation of all the listings contained in the dataset according to the decision rules that we used to adapt it in order to discard listings that are placed out of the market.

From Figure 3.1 it can be seen that thanks to the first step we removed all those apartments that do not meet the demand preferences because their price do not correspond to the characteristics of their apartment. Moreover, with the second step, we mainly removed apartments with zero days available that are not available because the user does not want to rent them, rather than because they are crowded with guests.

8.1.2 Filtering, transforming and dealing with predictor's missing values

As explained, our goal is to create a pricing model able to yield the user an interval of the market price for an apartment like the one that he/she is publishing, taking into account all the market competition. Therefore, the user will be required to input the characteristics of his/her apartment and his/her Airbnb profile, and those characteristics, will be the predictors Xs of the BNN, so the first task that must be conducted is to filter the predictors of our dataset and remove all those that the user will not be able to give or that are not meaningful for him/her. Apart from this, we will also remove all those predictors that could be interesting but would need specific techniques like text mining in order to be used and, moreover, those predictors that are included within other predictors. In the annex (*Summary about the reasons why we discarded each predictor of the dataset* from section *Implementing BNN as a pricing model for Airbnb in Barcelona* of the annex) there is a summary that explains the reason why we discarded each of the predictors of this first stage.

Just as an example of this pruning step, we will delve into some particulars predictors, to sketch out the rationale behind it. For instance, the variable *host_name* which collects the

username of the host is discarded as a predictor for our pricing model, mainly because the user would not find it reasonable that his/her name would modify the market price of the apartment. As an example for the text mining ones, the most relevant one is *house_rules*. Of course, this text variable could be relevant, since it captures the restrictions about the use of the apartment and, therefore, affects the utility that the customer receives from using it. However, since there are different ways of writing the house rules, if this variable is not properly treated, it can be misleading for our pricing model, so it is discarded. Finally, for those variables that are kind of included within other predictors an example is the *zipcode*. In particular, this variable presents many categories and refers to the localization of the apartment. However, its information is too specific in order to be directly included in the model and, instead, capturing the localization with a more aggregated variable like the district seems more useful than the zipcode, so the zipcode is discarded. Of course, it could be interesting to include it through a random effect capturing the spatial correlation but, due to the scope of this project, this is not contemplated for our BNN.

As a result, we removed a total of 48 variables from our dataset and for the remaining ones, we will analyze them using the listings that passed the two-step adaptation process. In particular, we will apply suitable transformations in order to facilitate its understanding by the user and, therefore, facilitate the use of the pricing model. Apart from enhancing the meaning of the predictors with this transformation, we will also deal with the missing values of each one of them. In the annex (*Transformation and missing values treatment for the selected predictors* from section *Implementing BNN as a pricing model for Airbnb in Barcelona* of the annex) there is a summary that contains, for each of those predictors, a brief description, the transformation applied and the decision about the missing values.

Just as a way to summarize this process we will present two examples about transformations. For instance, the variable *weekly_price* includes the published price for one-week stays but, since our response variable will be the price per night, this will not be in the response variable, but it can be used as predictor. However, asking the user which weekly_price wants to impose just to predict the price per night would be both misleading (since he/she is using the pricing model to know the price) and complicated to decide. Instead, we converted this variable into *weekly_discount*, which captures the percentage of discount of the weekly_price with respect to the price per night and we did it that by creating four categories: *Negative, None, Up until 25%* and *More than 25%*. With these categories as possible answers, a straightforward question, can be asked to the user: Which approximated percentage of discount are you willing to apply for one-week stays? Another example is the variable *amenities*, which is a text variable that includes all the amenities that the user posted about the listing. The sensible

transformation that we applied was creating a categorical binary variable for each amenity, stating whether if the apartment has it or not.

For the missing values treatment the approach has been based in two principles: Capturing in a particular modality the missing values and avoid erroneous listings that can interfere in our pricing model. For the first principle, it is obvious that it can only be applied to categorical predictors. The main rationale behind that is that, for each variable we encountered a sense for the missing values, meaning that it would be useful to keep them in a category. For instance, and following with the same example, a missing value in the weekly price would mean that the host is not applying any discount, so it is sensible to capture it in a separated category called *None*. For the second principle, we refer to the fact that some listings present missing values in relevant variables such as the number of beds or the date since the host has been host which could be caused, mainly, because the dataset has been obtained through web-scraping and there were problems with the format. Therefore, in order to avoid those listings with missing values to wrongly interfere in our pricing model with inaccurate information and, since they were just, adding all the variables, 120 observations, we decided to remove them from our dataset, like if we considered that they are placed out of the market because they do not reveal the required information to the customers.

After all this process, the dataset has final dimensions of 14,100 observations and 115 variables. The reason why there are more variables is because we parsed some text variables and created categorical variables according to them, like the ones relating to the amenities.

The final step of this preprocessing procedure will be dealing with the response variable of our pricing model, i.e. the price. Here are presented the histogram of the price per night and the logarithm of it.

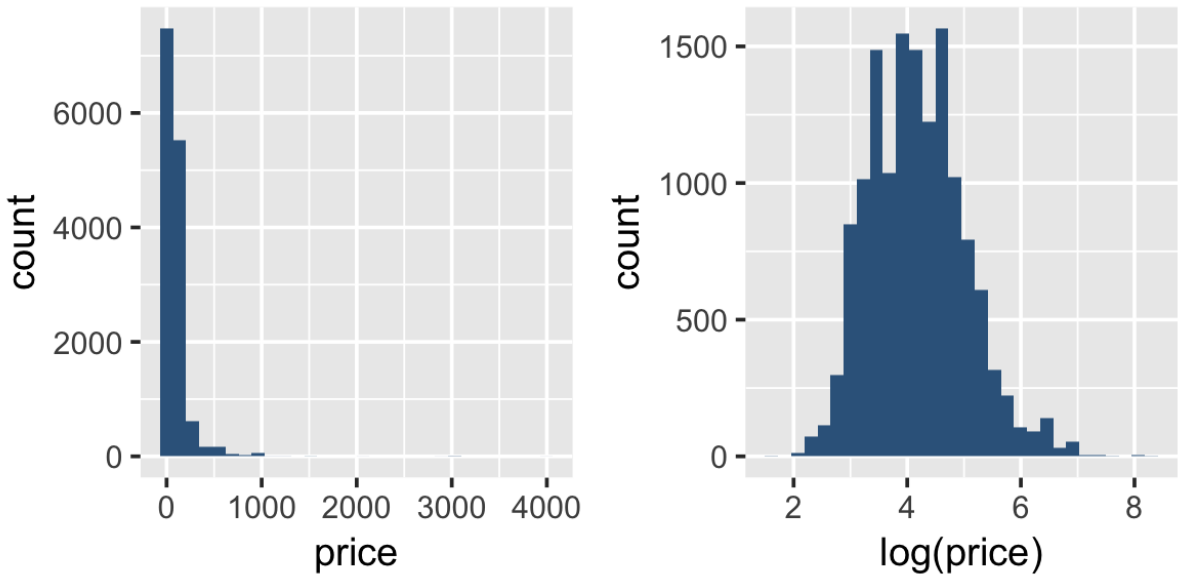


Figure 3.2: Histogram of the listings' published price (left) and its logarithm (right).

Observing these histograms, one can see that there is the possibility of outlier existence which could spoil our pricing model. In order to assess what is happening with all those observations with an extraordinary high price, we searched them on the Airbnb website. In order to do that, one needs to access the following URL where the id refers to the variable *id* of our dataset: https://www.airbnb.com/rooms/{id}

In particular, we started searching the listing with the highest price and we continued up until a point where most of the listings were correct, which was at the price of 1,100 dollars per night. In particular, we encountered that for the 17 listings with a price higher than 1,100, 8 could not be found on the website, meaning that the platform removed them because either they were scams or because they did not fulfill the requirements established. Besides that, we encountered that 5 of the listings passed from a price higher than 1,100 to a price between 50 and 100 dollars per night, meaning that published price was erroneous when the scraping wave was conducted. Finally, only 4 of those 17 listings were correct but the main problem is that they are listings of extremely high luxury because they are either yachts or complete mansions. This rental of luxurious apartments is a different market in which the equilibrium is far from a competitive one, since only a small number of suppliers exists. Therefore, we considered that they are not from the general market that we are willing to analyze and, as a consequence, in order to avoid interference when estimating the market price, we discarded them. For prices lower than 1,100 (we analyzed until 950) the luxury level clearly diminished and, moreover, we were able to encounter almost all the listings so that is why we decided to cut off at 1100 dollars. Just as a final note about the listing price, we will use the price with the cleaning fee included, as it has been explained in the annex when dealing with the variable *cleaning_fee*. Moreover, we decided to work with logarithm of the response variable in order to avoid predicting negative values.

Therefore, the final conclusion is that we discarded a total of 17 listings, which endowed us with a final dataset of 14,083 observations and 115 variables to build the BNN (114 predictors and the response variable). After this, we conducted a bivariate descriptive analysis between each predictor and the response variable, in order to understand the role of each predictor and detect possible problems with the preprocessing procedure. All the plots from this bivariate descriptive can be found in the annex (*Bivariate descriptive analysis of each predictor with the logarithm of the listing price* from section *Implementing BNN as a pricing model for Airbnb in Barcelona* of the annex).

The main conclusion extracted from it is that it seems that there are a lot of predictors, especially those related with the amenities that does not seem to have any impact on the price of the listing but, of course, they will be included in our BNN, just to analyze possible

interactions with other variables. Another relevant conclusion is that both the type of room that is being published (i.e. entire apartment, shared room or private room) and the district seem to be relevant for the price of the listing, which seems reasonable since they define different substitutive markets.

After this, we have divided our dataset into train, validation and test data with proportions of 0.6, 0.2 and 0.2, which yielded three sets of 8,450, 2,817 and 2,816 observations respectively. As it will be explained in the next subsection, the goal of the validation set will be to compare the linear model with the BNNs, while the goal of the test set will be to deliver an unbiased estimate of the predictive performance of the best model chosen according to the validation set. Moreover, we have standardized the numeric predictors of our train dataset and converted the categorical variables into dummies using contrast treatment, i.e. with a baseline category.

8.2 Finding a suitable architecture through Design of Experiments

The total number of predictors is 114 which, translated into a model matrix means that there will be 139 input nodes. As a consequence, and following the proposed methodology, we will start with a small number of nodes in the first hidden layer, since increasing one more node causes a relevant impact on the number of parameters to be estimated. In particular, we will test 2, 5 and 8 nodes in the first hidden layer. For the second hidden layer, in order to avoid complicating it too much, our values for the Box-Behnken design will be 0, 5 and 10. For the learning rate we have decided to test small values of it, in particular: 0.001, 0.051 and 0.1. Finally, just as it was explained in subsection 6.2.3, we will either use Neural Networks with only *tanh* or *logistic* activation function.

In order to ensure that we are able to capture the effect of each ML-hyperparameter on the out-of-sample predictive performance of the ANN, we decided to use 5 replicates per each EC. Since we have used two central points in the Box-Behnken design, we conducted a total of 140 experiments[8], which required a total time of 41 minutes. With those results, we fitted the best linear model applying logarithm to the error metric (RMSE in this case) of the out-of-sample prediction. According to BIC, the best linear model, which has an R^2 of 78.69%, has the following predictors:

- Second degree polynomial of the number of nodes in the first hidden layer

[8] 140 comes from: [(3 numeric factors * 4 EC per each factor + 2 central points) * 2 categories of the categorical factor] * 5 replicates of each EC.

- Second degree polynomial of the number of nodes in the second hidden layer
- Second degree polynomial of the learning rate
- Categorical variable referring to the activation function
- Interaction between the third and fourth items of this list

During this thesis, it has been explained that once the equation of the best linear model is obtained, one should analyze it in order to obtain the minimum value of the out-of-sample error. However, since the best linear model obtained is rather simple, as there are not many two-way interactions, we will assess the effect of each ML-hyperparameter by fixing the other ML-hyperparameters and predicting the out-of-sample log(RMSE) for a range of values of the ML-hyperparameter that we are analyzing.

For the number of nodes in the first hidden layer, we will use the range between the values that we have tested (i.e. between 2 and 8) and we fix the other numeric factors at their central point and the activation function at *tanh*. This is done because the number of nodes in the first hidden layer does not interact with any other ML-hyperparameter according to the best linear model, so changing the fixed value for the other ML-hyperparameters would only move upwards or downward the curve. The effect of the number of nodes in the first hidden layer is summarized in Figure 3.3:

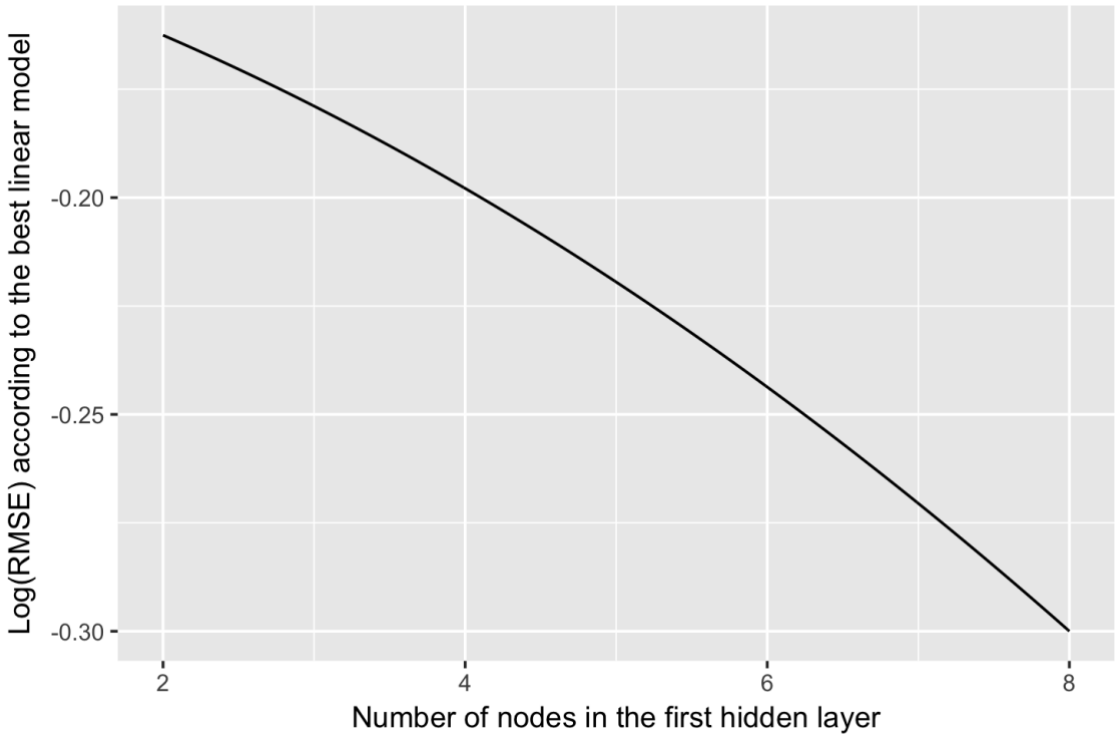


Figure 3.3: Predicted value of the log(RMSE) for a range of nodes in the first hidden layer according to the best linear model fixing the number of nodes in the second hidden layer at 5, the learning rate at 0.051 and with tanh activation function.

Thanks to the linear model we have been able to see that, the more nodes in the first hidden layer, the better will be the out-of-sample prediction. This is a very relevant result, since we know that there are possibilities of enhancing our ANN prediction.
For the number of nodes in the second hidden layer we fixed the other ML-hyperparameters at their central point, we worked with *tanh* activation function and we predicted the out-of-sample RMSE error for a range of values from 0 to 10 nodes in the second hidden layer. The resulting plot that summarizes the effect of the number of nodes in the second hidden layer is presented in Figure 3.4.

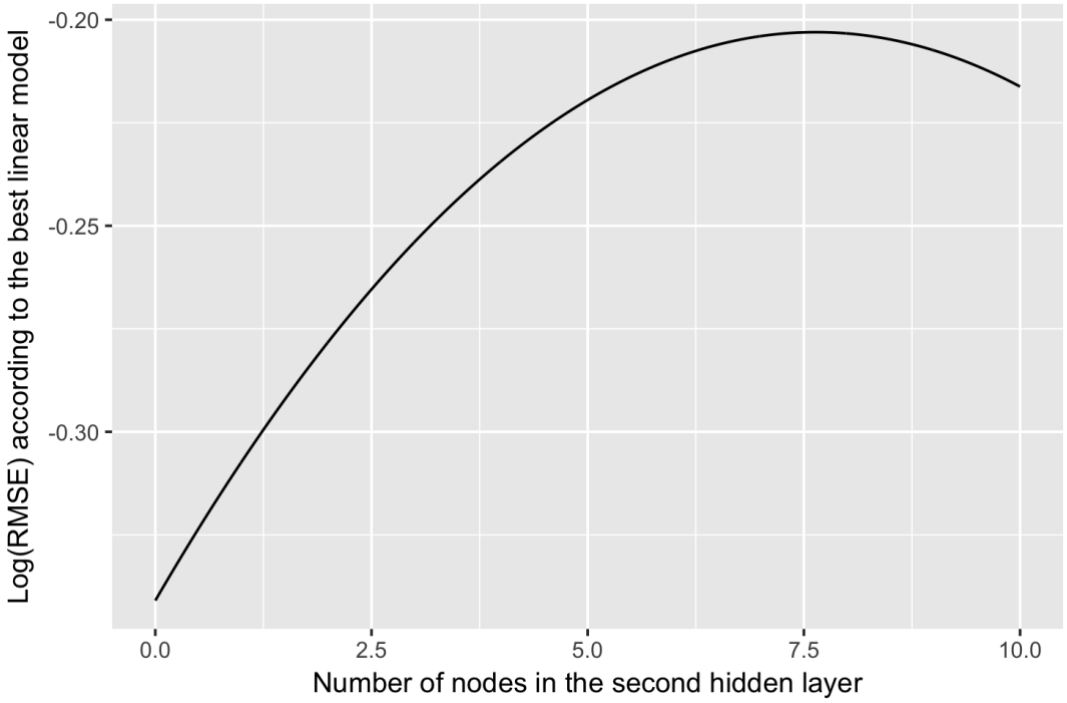


Figure 3.4: Predicted value of the log(RMSE) for a range of nodes in the second hidden layer according to the best linear model fixing the number of nodes in the first hidden layer at 5, the learning rate at 0.051 and with tanh activation function.

According to the best linear model the best option would be to work with only two hidden layers (the first and the third one), since a lower out-of-sample RMSE is yielded when there are 0 nodes in the second hidden layer. Apart from that, we can also see that, if we fit ANNs with more than 10 nodes in the second hidden layer we may obtain a better out-of-sample prediction, since the error seems to diminish after 8 nodes in the second hidden layer.

Finally, for the learning rate and activation function we will need to use two curves, since they are interacting with each other. In particular, in both curves we will fix the number of nodes in the hidden layers at their respective central points, but the difference will be that, in one curve we will use *tanh* activation function to predict the out-of-sample log(RMSE) for the supplied range of the learning rate, while in the second curve we will use the *logistic* activation

function. Those effects of the learning rate depending on the activation function are shown in Figure 3.5.

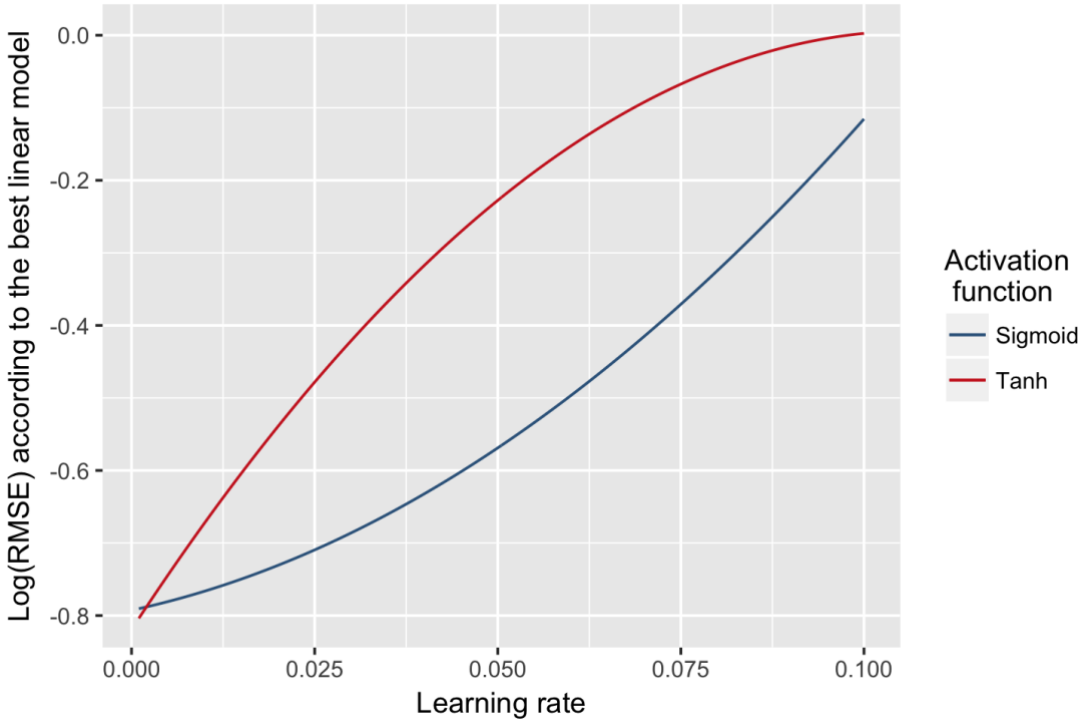


Figure 3.5: Predicted value of the log(RMSE) for a range of learning rate according to the best linear model fixing the number of nodes in the first and second hidden layer at 5.

Even though the effects of the learning rate depend on the activation function, for small values of the learning both activation functions yield a similar out-of-sample RMSE. For our purpose of finding a suitable architecture, the main conclusion is that, in order to find ANN with accurate prediction we need to work with learning rate values close to 0.001 and, if we do that, we do not need to worry about the activation function.

According to the results obtained, the best ANN in the tested range of ML-hyperparameters, would be a two-hidden layer ANN with 8 nodes in the first hidden layer, with a learning rate of 0.001 and with either *logistic* or *tanh* activation function. However, thanks to the linear model, we have been able to see that, if we move from this region of the ML-hyperparameters space, we could be able to find ANNs with superior out-of-sample prediction. Moreover, and this is the key advantage of assessing a suitable architecture for an ANN through DoE, we know which region of the ML-hyperparameter space we should explore.

We have decided to increase the number of nodes in the first hidden layer, since Figure 3.3 suggests a clear reduction of the out-of-sample prediction for larger values of that ML-hyperparameter. In particular, we are going to test three values in order to capture nonlinearities and they will be 10, 15 and 20 nodes. For the second hidden layer, even though it seems that increasing the number of nodes could reduce the prediction error it looks like a

lot of new nodes should be added in order to find a similar log(RMSE) than the one obtained with 0 nodes in that second hidden layer. In order to avoid too much extrapolation, we decided to fix the number of nodes in the second hidden layer at 0, so we will not explore that region. When it comes to the learning rate, even though in Figure 3.5 it seems that reducing the learning rate could further decrease the out-of-sample error, the thing is that the learning rate is almost 0, so changing it will not have any impact on the prediction. Therefore, we have decided to fix the learning rate at 0.001 for this second experimental stage. Finally, for the activation function, even though the two seem similar in Figure 3.5 for learning rates of 0.001 we have decided to maintain it, because the extra cost that it will suppose, in terms of time, is minimal.

In this second experimental stage, since the only factors will be the number of nodes in the first hidden layer and the activation function, we will not use a Box-Behnken design. Instead, our ECs will be all possible combinations between the chosen levels for the number of nodes in the first hidden layer (10,15 and 20) and the activation function (*tanh* and *logistic)*. Since we will use 5 replicates per each EC, the total number of experiments conducted in this second stage will be (3*2*5) = 30, which took a total time of 7 minutes. Therefore, experimenting with the activation function only required to pass from 15 to 30 experiments, which translates into 3.5 minutes.

Like in the first experimental stage, we applied logarithms to the obtained out-of-sample RMSE per replicate and we fitted the best linear model up to two level interactions and second degree polynomials according to BIC, using only the activation function and number of nodes in the first hidden layer as predictors, since in the experiments conducted in this second stage, which are the only ones used to fit this linear model, the other ML-hyperparameters are fixed. The best linear model, that has an R^2 of 71.08%, comprises the following elements:

- Second degree polynomial of the number of nodes in the first hidden layer
- Activation Function

Since the interaction is not included, it means that we can separately analyze the effect of each ML-hyperparameter in this new region. If we fix the activation function at *tanh* and we predict the out-of-sample log(RMSE) for a range of nodes in the first hidden layer from 10 to 20 according to this best linear model obtained in the second experimental stage, we obtain the curve in Figure 3.6.

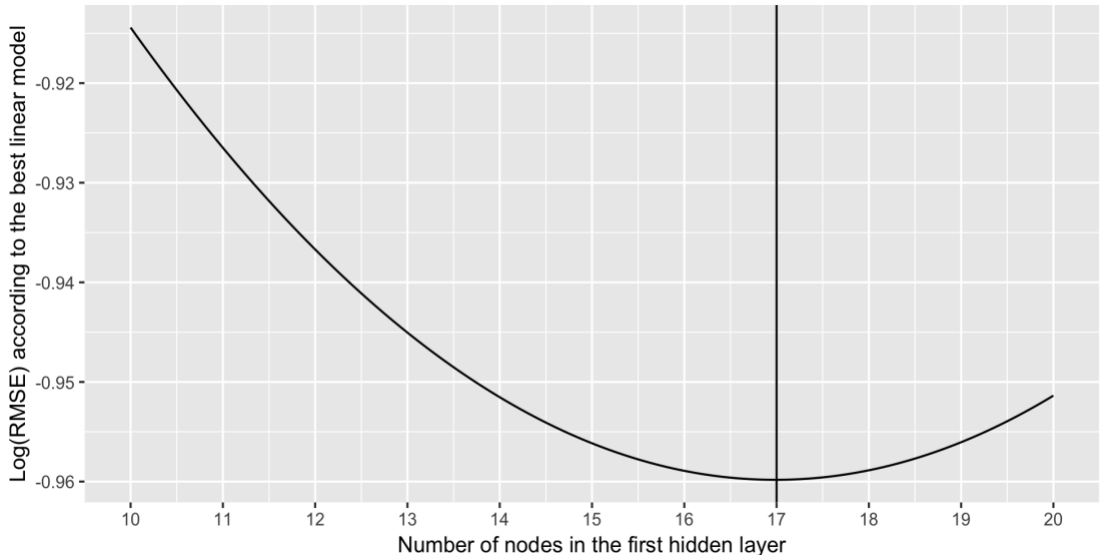


Figure 3.6: Predicted value of the log(RMSE) for a range of nodes in the first hidden layer according to the best linear model obtained in the second experimental stage fixing the activation function at tanh.

According to this, after 17 nodes in the first hidden layer the out-of-sample error seems to increase, so we will use 17 nodes in the first hidden layer. After this, we need to analyze the activation function, since we know that it has been included in the best linear model of this second experimental stage. Since we have built a Bayesian linear model, we present in Figure 3.7 the posterior distribution associated to the parameter of the dummy variable created for the category *tanh*.

In Figure 3.7 it can be seen that the effect of *tanh* is negative and, moreover, that the 0 is not included in the 95% probability interval of the parameter (vertical bars), meaning that this effect is *significant*. Therefore, since the parameter is negative, there is a reduction of the out-of-sample error when *tanh* is used.

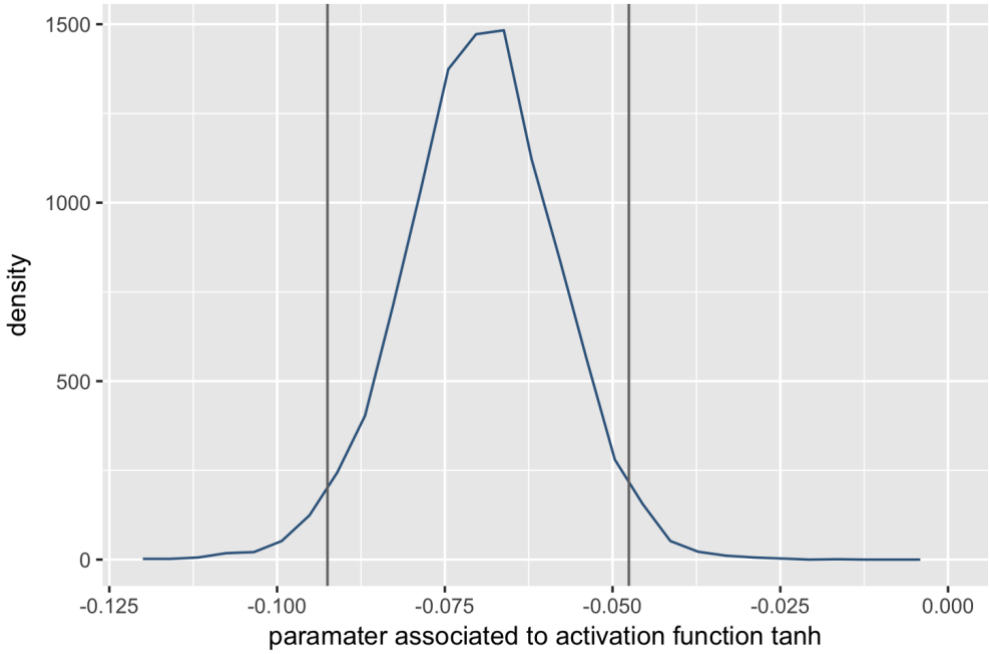


Figure 3.7: Posterior distribution of the parameter that captures the effect of changing from logistic activation function to tanh in the best linear model for the second experimental stage.

Therefore, and as a conclusion of the DoE procedure, we have found that the best architecture for our ANN will be the one with two hidden layers, with 17 nodes in the first hidden layer, 2 nodes in the second hidden layer (the last hidden layer will always have 2 nodes in our proposed ANN), learning rate of 0.001 and activation function *tanh*. As a consequence, we will fit a BNN according to these values of the ML-hyperparameters, without considering the learning rate because the weights in the BNN are obtained through the posterior distribution and not trough an iterative optimization algorithm, so regularization is implicit when estimating the weights and there is no need to provide this ML-hyperparameter.

Before finishing this subsection, It is relevant to note that, if we had used k-fold cross-validation in order to assess a suitable we would only have explored the first region of the ML-hyperparameter space, since we would not have been able to determine that with more nodes in the first hidden layer the out-of-sample prediction of the ANN becomes more accurate. As a way to observe the potential of choosing a suitable architecture with DoE instead of CV we present a boxplot with all the log(RMSE) obtained in all the experiments conducted, separating them between experiments of the first stage and the second stage.

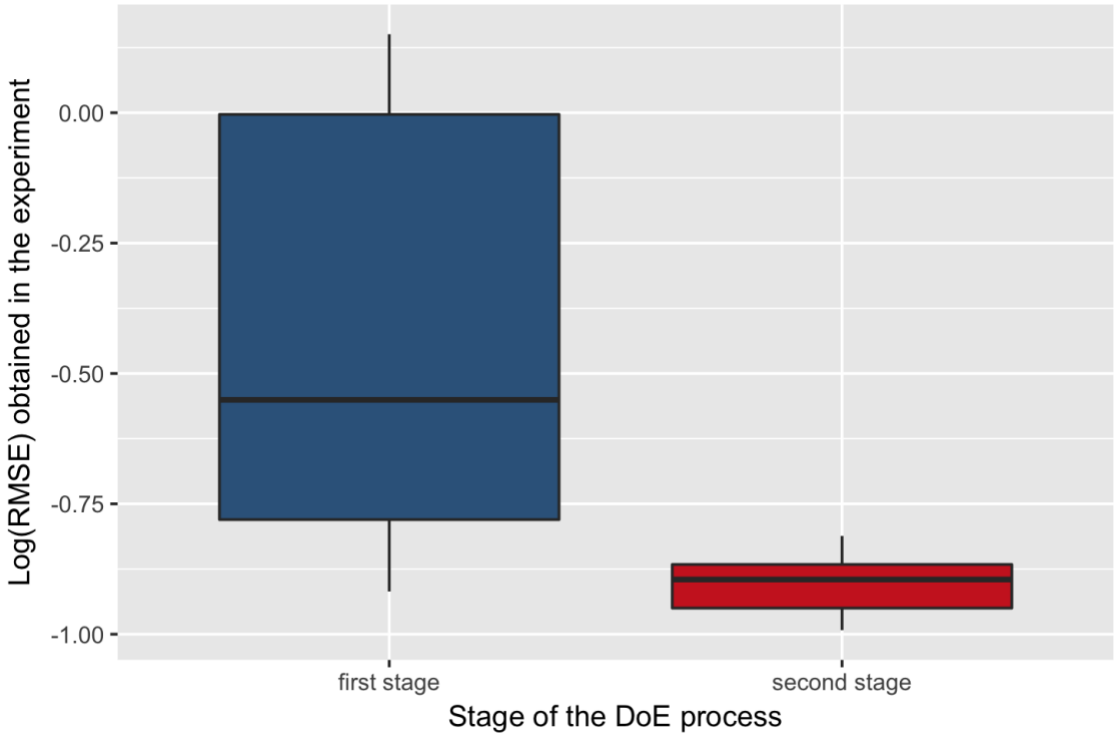


Figure 3.8: Boxplot of the logarithm of the out-of-sample RMSE obtained in the conducted experiments from both the first and second experimental stage.

According to this, it is clear that in the second region the out-of-sample RMSE is smaller and, with CV we would have never had the chance to discover it. Therefore, even though choosing a suitable architecture through DoE requires more effort, it uses all the information generated by cost-effective experiments and, as a consequence, we can be more confident about the

found optimum and, moreover, we are able to find better regions in the ML-hyperparameter space.

8.3 Validation and comparison with the best linear model

After having found a suitable architecture for our BNN, we are going to fit both the baseline proposed BNN (i.e. the one without hierarchical model) and the BNN with ARD, in order to enhance our analysis about Airbnb.

The first validation that must be conducted is the one related with Multidimensional Convergence of the MCMC simulation, so in Figure 3.9 it is attached the evolution of the log-likelihood of each chain for the baseline BNN and in Figure 3.10 for the BNN with ARD. In both of them, we let 4,000 simulations for 4 chains and took one simulation every five, to avoid correlation between the posterior samples.

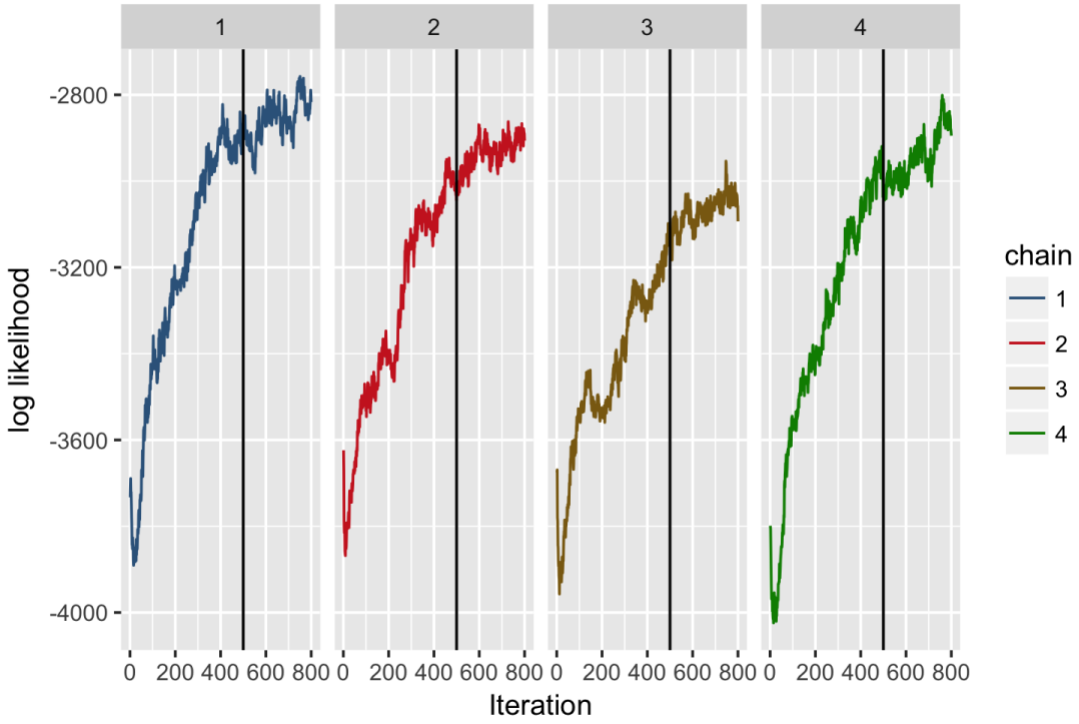


Figure 3.9: Log-likelihood chains associated to the simulations of the BNN

In those plots it can be observed that the chains seem to arrive a steady state after the 500th simulation (that is, the 2,500th original simulation), so we will take samples after that simulation because it is when each chain has arrived a local optimum of the posterior distribution, for both cases. Of course, the fourth chain in the Figure 3.9 and the third in Figure 3.10 do not seem like they have completely converged so, even though we may not be fully sampling from the posterior distribution with each chain, we will be close to it. If we let further simulation, we should obtain better results with our BNN but, since the purpose of this project

is to disclose the possibilities of the BNN we will not further simulate the chains, mainly due to the computational cost that they require.

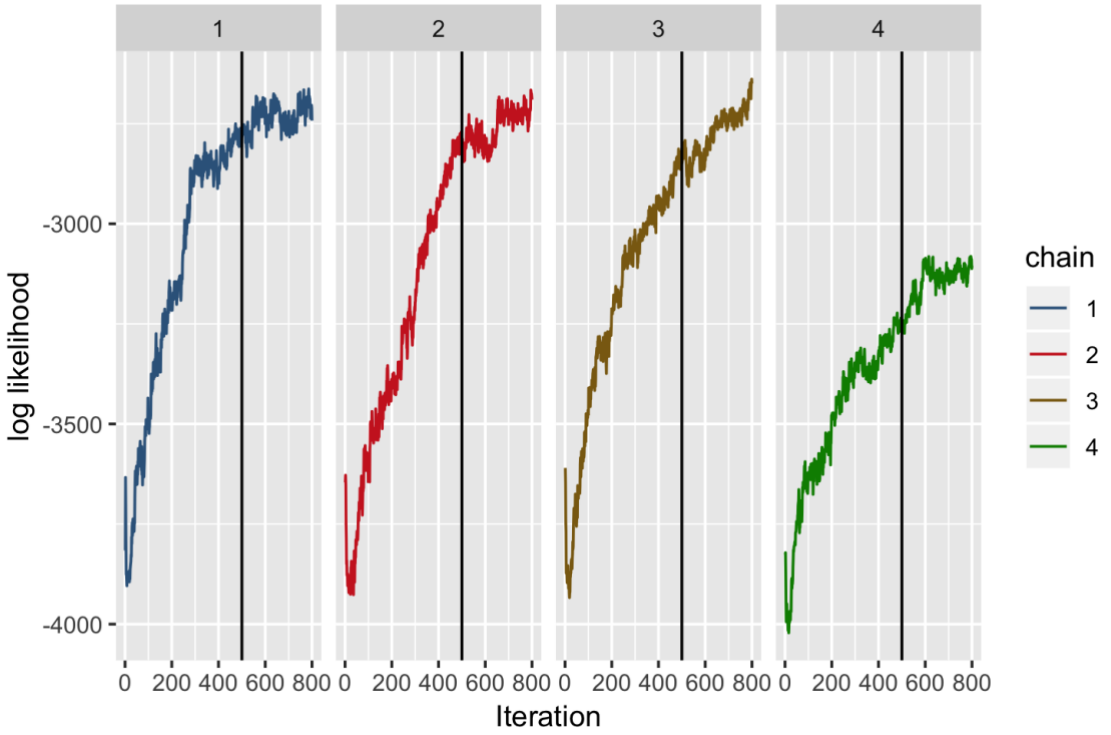


Figure 3.10: Log-likelihood chains associated to the simulations of the BNN with ARD

In Figure 3.11, on the left, there is a QQplot of the BNN residuals in order to decide whether if the normality distribution hypothesis for the response variable is suitable or not. In the same Figure 3.11, on the right, there is a representation of the residuals against the predicted value that will be used to check the homoscedasticity hypothesis. The same plots included in Figure 3.11, but for the BNN with ARD, can be found in Figure 3.12.

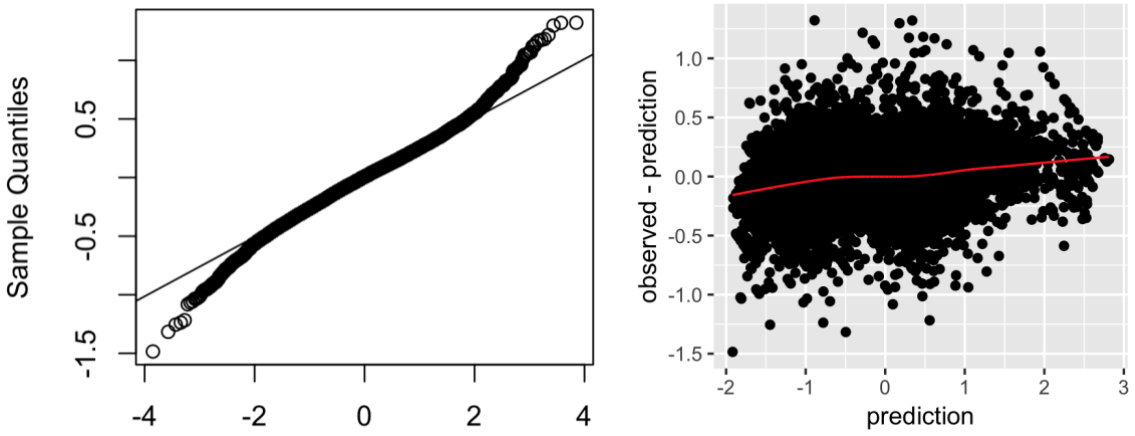


Figure 3.11: QQplot of the BNN residuals (left) and BNN residuals vs BNN prediction on the right. The prediction is the standardized logarithm of the response variable.

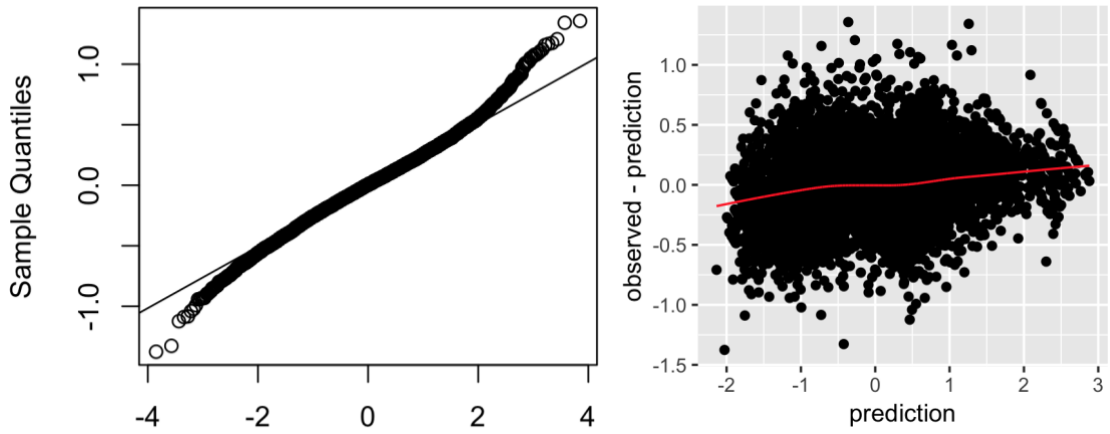


Figure 3.12: QQplot of the BNN with ARD residuals (left) and BNN with ARD residuals vs BNN with ARD prediction on the right. The prediction is the standardized logarithm of the response variable.

The first element observed in these plots is that both BNNs seem to behave equally, so we will extract the same conclusion for each of them. The conclusion is that the BNN is not completely validated, since the residuals do not have a normal distribution. In fact, it seems that there is a problem with the tails of the distribution and, maybe, working with a more robust noise such as a t-student noise would help to eliminate this problem. For the homoscedasticity it is clear that the slope of the local regression is almost negligible, so it is considered that the variability of the normal distribution for the response variable is constant. Even though working with a t-student noise could be a solution, we will continue with our BNNs with Gaussian noise, since it simplifies the analysis and it does not seem a complete error according to the QQplot. The main consequence is that we will not be able to completely rely on the intervals provided, but they will be meaningful.

Now that we have decided to keep those BNNs, the next step will be to compare them with the prediction obtained by the best linear model. In order to compare those predictions, we will use, for the first time, the validation set because we will compute the RMSE of the predictions in it.

In order to obtain the best linear model in a reasonable time, we have conducted a five-step procedure. First of all, we started with the null model and we added categorical variables and linear terms of numeric variables according to BIC which yielded the model “Best linear model step 1”. Then, we added, sequentially, quadratic terms for the numeric variables that were included in “Best linear model step 1” and, for each term that we included we compared the BIC with the previous best model. After this process, we obtained the “Best linear model step 2” which, compared to “Best linear model step 1”, now has some numeric predictors with quadratic effects and some other with linear effects.

The third step aimed to include those numeric predictors that do not have a linear effect on the response variable but a quadratic effect and, therefore, were not included in the previous model because the linear effect was never included in "Best linear model step 1". In order to do that, we added to "Best linear model step 2" all those not included numeric predictors with quadratic effects that were able to reduce the BIC, and this yielded the "Best linear model step 3". The fourth step consisted in repeating the process of the second step but with cubic terms, meaning that we tried for each variable with quadratic effect in "Best linear model step 3", its cubic effect and we kept those that reduced the BIC. This gave us the "Best linear model step 4", which has numeric predictors with linear effects and some with cubic effects. We then aimed to include those numeric predictors that still have never entered the model by introducing them directly with third degree polynomials, but none reduced the BIC, so they were discarded.

After this, we decided to stop including more polynomial degrees for the numeric predictors, so the fifth step consisted in finding the best model with two-way interactions with the terms included in "Best linear model step 4". This yielded the "best linear model step 5", which we considered to be the best linear model up to third degree polynomials and two-way interactions. In Table 3.1 there is a summary of the BIC per each linear model:

Linear Model	BIC
Step 1: Linear model with categorical variables and linear effects	9,099.26
Step 2: Linear model with predictors from step 1 but with useful quadratic effects	9,002.16
Step 3: Linear model from step 2 with quadratic effect of the non-included numeric predictors in step 1 (only the ones that reduced BIC)	8,827.2
Step 4: Including cubic terms to linear model from step 3	8,477.4
Step 5: two-way interactions of lineal model from step 4.	7,795.08

Table 3.1: BIC of the best linear model obtained at each step to find the best linear model up to third degree polynomials and two-way interactions.

From now on, we will refer to "best linear model step 5" just with "best linear model" and, a summary of the terms that are included in that model can be found in the annex (*Included terms in the best linear model* from section *Implementing BNN as a pricing model for Airbnb in Barcelona* of the annex). In Figure 3.13 there is a normal QQplot of the residuals associated to the linear model, just to show that the normality assumption is also not completely hold in the linear model either.

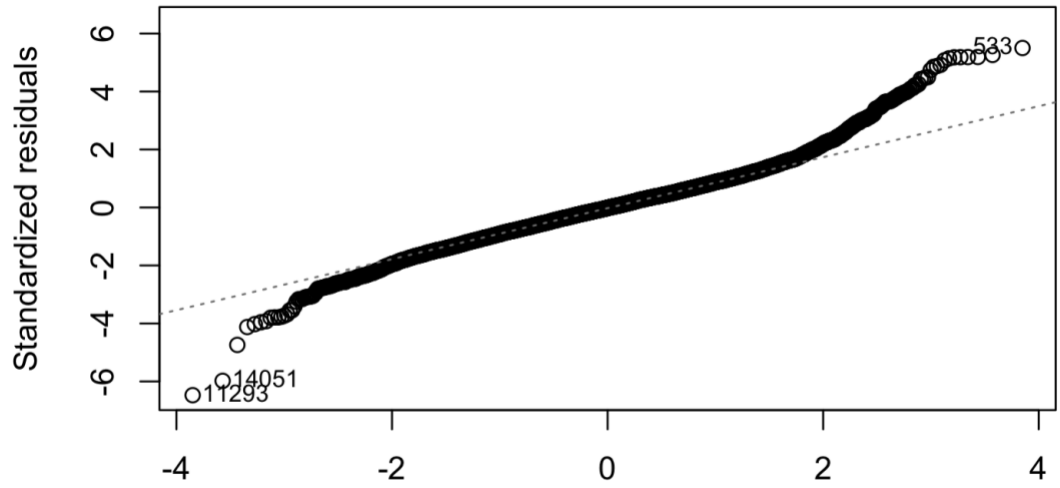


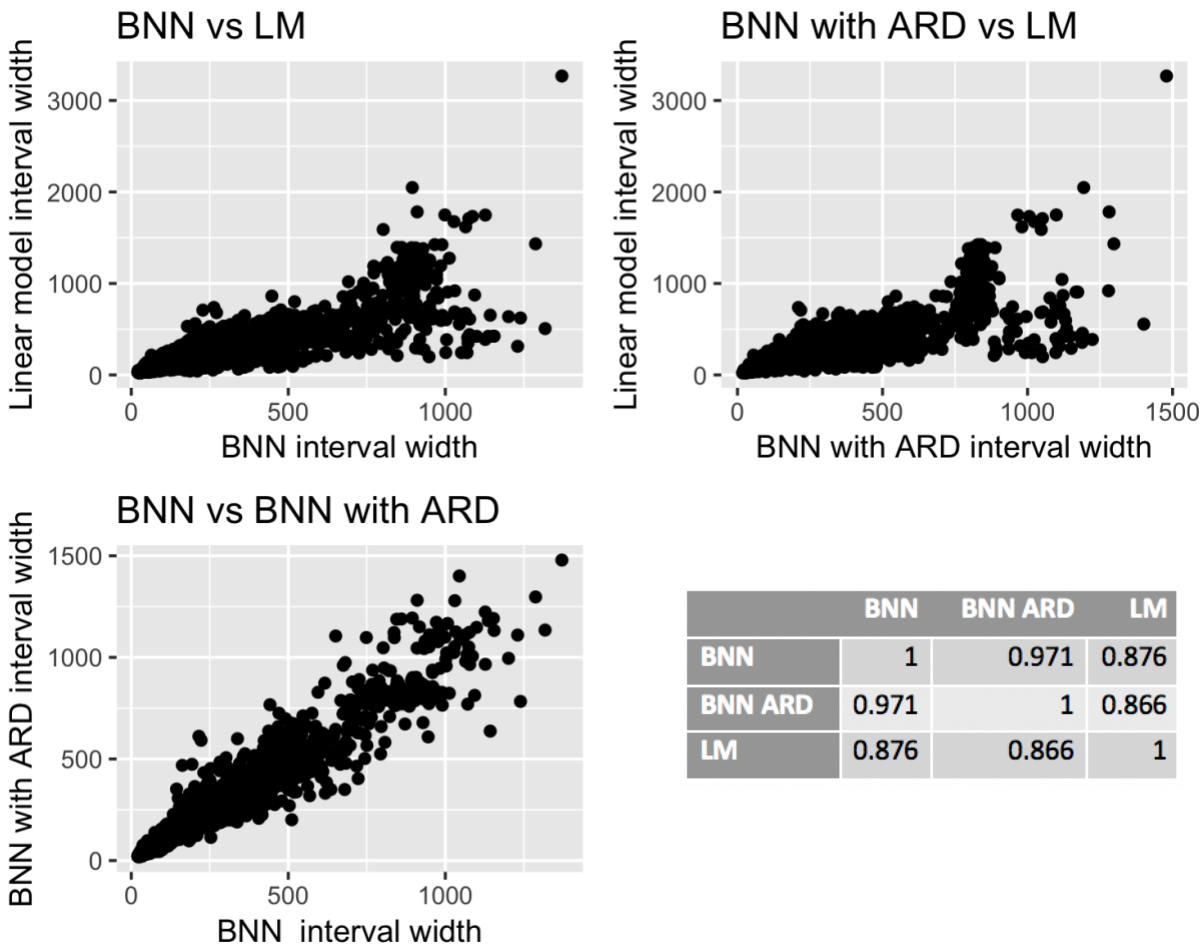
Figure 3.13: Normal QQplot of the standardized residuals of the best linear model.

The best linear model achieved an RMSE on the validation set of 74.88 dollars. The ANN with the best architecture, as it was expected because it easily deals with nonlinearities, obtained an average RMSE of 71.2 dollars. We provided an average for the ANN because several were instituted in order to find the initial values for the BNN chains and, as we know, there are several local optima for ANN so the predictive performance of each one is not the same. Finally, the baseline BNN with 4 chains achieved a remarkable predictive performance, since the RMSE on the validation set was, only, of 66.12 dollars, almost 10 dollars less than the best linear model. However, the predictive performance of the BNN with ARD was even more astonishing, since the RMSE on the validation was of 63.69 dollars. For the BNNs we used as punctual prediction the expected value of the posterior distribution associated to the localization parameter of the individual's posterior predictive distribution. In other words, we used $\hat{y}_i = E(\mu_i|y)$.

Thanks to these more accurate predictions, the BNN will yield more reliable predictions about the market price and, therefore, the users of the pricing model will be able to take better decisions than if the linear model was provided, so this reduction of error has a relevant impact in the final effect of the pricing model. As explained in the first chapter, when the user takes better decisions it means that the price that he/she fixes is closer to the demand preferences and, therefore, more transactions are conducted. In this case, it means that more customers will be interested in his/her apartment and, as a consequence, he/she will be able to rent for more days the listing, which leads to a reduction of idle products.

The final part to validate the BNNs will be to compare them with the best linear model in terms of how they are dealing with uncertainty. In Figure 3.14 we present, for all combinations of the three models that we obtained, a scatterplot of the 95% probability interval width

associated to the listing price in the original scale by each model, and the Pearson's correlation matrix for those widths.



	BNN	BNN ARD	LM
BNN	1	0.971	0.876
BNN ARD	0.971	1	0.866
LM	0.876	0.866	1

Figure 3.14: Scatterplots of the 95% probability interval width of the published price for each listing in the training set for each combination of the models and the associated Pearson's correlation matrix

It can be seen that the uncertainty of the linear model is not exactly similar to the one offered by the BNNs, especially for those listings that present a width higher than 500 dollars in the BNN. However, since the majority of the listings have a smaller width than 500 (they are the 96.1% of the train set in the baseline BNN and 96.05% in the BNN with ARD) it can be concluded that the models seem to endow the predictions with similar uncertainty, which means that our BNN is well-behaved. In order words, those listings about which the linear model is more uncertain are the same listings about which the BNNs are more uncertain.

If we compare the average width of those intervals, having in mind that this metric is heavily influenced by listings with a very wide interval as Figure 3.14 suggests, we obtain 166.32 dollars for the BNN, 163.61 for the BNN with ARD and 182.33 for the linear model. This means that the linear model intervals are wider than the ones offered by the BNNs. Knowing that, the following question is whether if the BNN intervals, which are narrower, are able to

correctly capture out-of-sample observations. To do that, we compare the percentage of listing prices from the validation set captured in the 95% probability interval of both the BNN and the linear model. The results are that the baseline BNN captures the 95.03% of the observations in the validation set and the BNN with ARD the 95.2%, while the linear model captures the 94.18%.

Therefore, the conclusion extracted is that both BNNs are yielding a prediction that is clearly outperforming the linear model. For starters, the punctual prediction is better but, moreover, the BNNs outperforms the linear model in the interval-based prediction, since it is able to yield narrower intervals (16 dollars less) that are equally able to capture the variability of the response variable (because they capture the same amount of observations in the validation set). Just as it was explained during the second chapter, the BNN (with or without ARD) is being able to better capture the behavior of the response variable and, as a consequence, it seems a more suitable model to predict the market price of the Airbnb apartments than the linear model, because it endows the user with a more informative tool to take decisions. Following the same rationale, if we were to choose between the baseline BNN and the BNN with ARD we would select the one with ARD, since the punctual prediction is more accurate, and the intervals are narrower (and able to capture the response variable variability).

Therefore, in the following section we will work only with the BNN with ARD, since it has been the best model obtained according to the validation set. In order to analyze this BNN with ARD we will use the observations in the test set to evaluate out-of-sample performance because, as explained during the second chapter, we need to evaluate and analyze the BNN through the test set because the validation set has been used to select it among other models, so it would be biasing to draw conclusions according that particular set. In other words, in this section we have decided that the BNN with ARD is clearly better than the linear model and, in the following section, we will study whether if this BNN with ARD is good enough for the purpose of helping the users take decisions.

9. Analysis of the results

The first element that we are going analyze about our selected BNN (i.e. the BNN with ARD) is its honest out-of-sample predictive performance with the test set, in order to analyze the quality of the model for its purpose, which is to be able to deliver the users the market price. The RMSE obtained in the test set is of 58.78 dollars, meaning that for each apartment, on average, there is an error on the predicted price of 58.78. However, the RMSE is calculated using the squares of the residuals and since the distribution of the price is very skewed, it delivers a rather high average error. Therefore, using the mean absolute error would be more

meaningful. If we compute that, we obtain an average error of 29.66 dollars in the test set. In order to further analyze this predictive performance, we will divide the observations of the test set according to the value that they take in the price creating four categories, and we will compute the average absolute error in each of those categories. The results are shown in Table 3.2:

Category of the price	Number of observations	Mean absolute error
Less than 50 $	818	10.71
Between 50 and 100 $	729	19.92
Between 100 and 200 $	878	28.53
200 $ or more	391	89.98

Table 3.2: Average mean error, in dollars, obtained by the BNN with ARD in the test set, according to four categories of the published price.

According to this, the predictive performance of the punctual prediction is more than suitable, especially for the apartments with a price smaller than 200$. What would be unacceptable is delivering average errors greater than the central point of the category created because, then, the user could not rely on the pricing model. For instance, imagine yielding 30$ or 40$ for the first category. Nevertheless, it is obvious that the punctual prediction will have some error because as we explained in the first chapter, two listings with the exact same characteristics do not have to present the same price, mainly because behind the price there is a human decision which is somewhat unpredictable.

Therefore, what is more relevant according to the purpose of this project is to obtain reliable and useful intervals for the price of the listing, because that is the element that will capture the behavior of the market (i.e. competitors and customer preferences) and, therefore, what will be meaningful for the user that is willing to publish an apartment on Airbnb. The main decision that must be taken about this interval comes from the fact that the BNN endows us, for any new apartment, with a whole probability distribution (the predictive posterior) of the possible prices that apartments like this new one are fixing or, in other words, we are given the probability distribution of the market price for apartments like the new one. Therefore, if we want to provide the user an interval for the market price, we need to decide a particular percentage of the probability of that distribution to be represented by the interval (i.e. how much of our market we want to represent).

Even though the default option in statistics would be to work with the 95% probability interval, which would be an interval that captures almost all competition, it is obvious that those intervals will be too wide for the user to help him/her take a decision. Instead, we believe that is better to provide the user with the 80% probability interval and, if he/she decides to choose

a price outside that interval, deliver him/her the probability that the market price of his/her apartment is still higher than the chosen one or, in other words, the probability that a competitor is still fixing a price higher than the chosen one and still getting some clients (as a reminder, we are assuming that the apartments used to train the BNN are placed inside the market, i.e., are receiving clients). However, in order to do that, we need to use the test set to validate if the 80% probability interval is really capturing 80% of the observations. For our BNN the percentage of observations of the test set captured by this 80% interval is 83.84%, which is reasonable and, therefore, using the 80% probability interval will be acceptable. The main reasons that explain why the 80% probability interval is not exactly capturing the 80% of the observations is, apart from the obvious variability of the test set, the fact that, as we have seen, normality is not completely validated and, moreover, that for some chains it did not seem that we achieved full convergence to the posterior distribution so, instead of sampling from the posterior they sampled an approximation of it.

Just like we did before with the punctual prediction of the market price, just to summarize the behavior of those 80% probability interval we present in Table 3.3 the width of the intervals, in dollars, according to each categorization of the response variable in the test set:

Category of the price	Number of observations	Average width 80% interval
Less than 50 $	818	39.5
Between 50 and 100 $	729	74.1
Between 100 and 200 $	878	138.43
200 $ or more	391	286.54

Table 3.3: Average width, in dollars, of the 80% probability interval of the posterior predictive distribution for each category of the listings' price.

According to this table, the intervals seems to be useful for the user because the width of the interval seems to be acceptable according to the price of the apartment. Just like with the punctual prediction, what would be unacceptable is that, for prices between 50 and 100$ the average width of the interval was of 120$, because the range would be too wide, and the user would find it useless to take a decision. However, if we want to capture almost all the market behavior we should use the 95% interval and, surely, we would have intervals way too wide which are useful because encapsulate all the market behavior, but they are not as useful as the 80% when it comes to help the user decide a plausible value for his/her apartment.

Of course, all these metrics and analysis have allowed us to decide that our BNN will be a useful pricing model for the users. However, they all can be a little abstract since are averages values. In order to show how is working the pricing model, we took a random sample of 10

observations from the test set and have shown their punctual prediction, their 80% probability interval and, moreover, the real value.

Inferior limit ($)	Superior limit ($)	Predicted Market price ($)	Real price ($)
19.45	49.81	31.47	40
98.8	237.03	154.04	180
45.02	118.64	73.61	65
25.88	65.03	40.49	40
34.44	163.02	64.68	50
38.7	110.74	64.93	200
89.84	189.56	131.7	100
19.66	47.13	30.2	20
125.01	626.44	254.64	200
22.38	59.96	37.23	50

Table 3.4: 10 random examples with the predicted intervals and market price and, moreover, with the exact price that they were published.

Imagine that there is a user with an apartment exactly equal than the third of this list. Thanks to the BNN now he/she is provided with two relevant pieces of information: The predicted market price for this/her apartment and the interval of this market price for his/her apartment. If this pricing model did not exist, he/she would have decided the price of the apartment just by checking the price of some *similar* apartments in the area and we remark the word *similar* because it would be impossible to find an exact equal apartment and also, because being *similar* would be a subjective impression of the user. Therefore, he/she would have taken a biased decision, since only a few apartments, in a subjective way, would be used to take the decision while in the BNN all the possible competitors, being similar, exact or very different from his/her apartment are objectively taken into account to provide the market price because, at the end, all of them are competing in the same market.

With this Table 3.4, apart from the usefulness of the pricing model, we want to remark that the taken decision will be quick and straightforward, since the user is endowed with all useful information in just a second. Moreover, if the user with an apartment exactly like the third one is confident about his/her apartment and decides to publish a price of, for instance, 140 dollars per night then the pricing model would show him/her the probability of the market price of being higher than 140, which would mean the probability of finding a client willing to pay a price higher than 140 dollars, since the market price reflects the price at which the transactions are being conducted. In our particular BNN this probability would be 0.043. However, in our case, since we were not able to build the BNN on top of a transaction-based

dataset, but rather a dataset with all the supply (that may or may not have found clients), the only thing that we can say is that, with a price of 140 the probability that a potential client finds a competitor with a higher price is only of 4.33% or, in other words, only 4.33% of this apartment direct competition is fixing a price higher than 140 dollars (but we cannot assure that they are still finding any client, because of the nature of our dataset). Even though our dataset is not the most suitable, note that without a pricing model able to capture the market behavior, these probabilities would be impossible to be extracted, so that is why we defend the outstanding usefulness of a pricing model from the Bayesian approach for P2P OM platforms.

Up until this point, we have demonstrated the utility and quality of our pricing model to help the users take suitable decisions, which is what has been explained during the first chapter. However, there is more knowledge waiting to be extracted from this BNN that can be used to explore and discover the Airbnb market in Barcelona. In order to do that, we will need to delve into the interpretation of the BNN. The first element that we will analyze is the level of competition in each district of Barcelona and, afterwards, we will delve into the effect of the predictors in our BNN. In order to interpret the BNN only the training observations will be used, since are the one that built the BNN.

As we have explained, the punctual prediction will not be exact for all the apartments, mainly because behind each real price there is an unpredictable human decision based on expectations and, moreover, because it can be that some characteristics of the apartment have not been collected in our dataset. The BNN encapsulates this variability inside the predictive posterior of each apartment which, as explained before, is the probability distribution of the market price for an apartment. Therefore, if we analyze the variability of that distribution we would be able to see the level of price differentiation that exist in apartments of that type. As market analysis from microeconomics states, the more price differentiation, the less competitive behavior there is in the market, because the suppliers are endowed with the ability of choosing a price for their product. However, we cannot use the variability of the posterior predictive, since it is clear that the variability will depend on the localization of the posterior predictive, meaning that for more expensive apartments, there is obviously a higher variability in the posterior predictive. As a solution, we propose to standardize that by computing the coefficient of variation of the posterior predictive of an apartment, which will tell us, for each dollar predicted, which is the uncertainty associated or, in other words, the flexibility of the supplier to fix a differentiated price.

Therefore, we can compute the average coefficient of variation for all the apartments of each district and use those values as indices of price differentiation that exists at each district.

Moreover, we know that for those districts with a small amount of observations, the predictive posterior will be wider, because the BNN has less information about it and therefore, endows that observation with more uncertainty, so the coefficient of variation will be higher. In fact, this is a positive aspect of our BNN, since we should expect higher price differentiation in those markets (i.e. districts) in which there are less competitors. In Figure 3.15 there is a scatterplot representing each district according to both the number of competitors in the district and the price differentiation index.

The first relevant aspect of Figure 3.15 is that, as it was expected, the more competitors in the market, the less price differentiation there is in the market, so the closer is the market to a unique price. What is relevant to note, also, is that Gracia, with a smaller number of competitors is closer to a unique price, meaning that the competition is more intense in that district than others like Sant Martí.

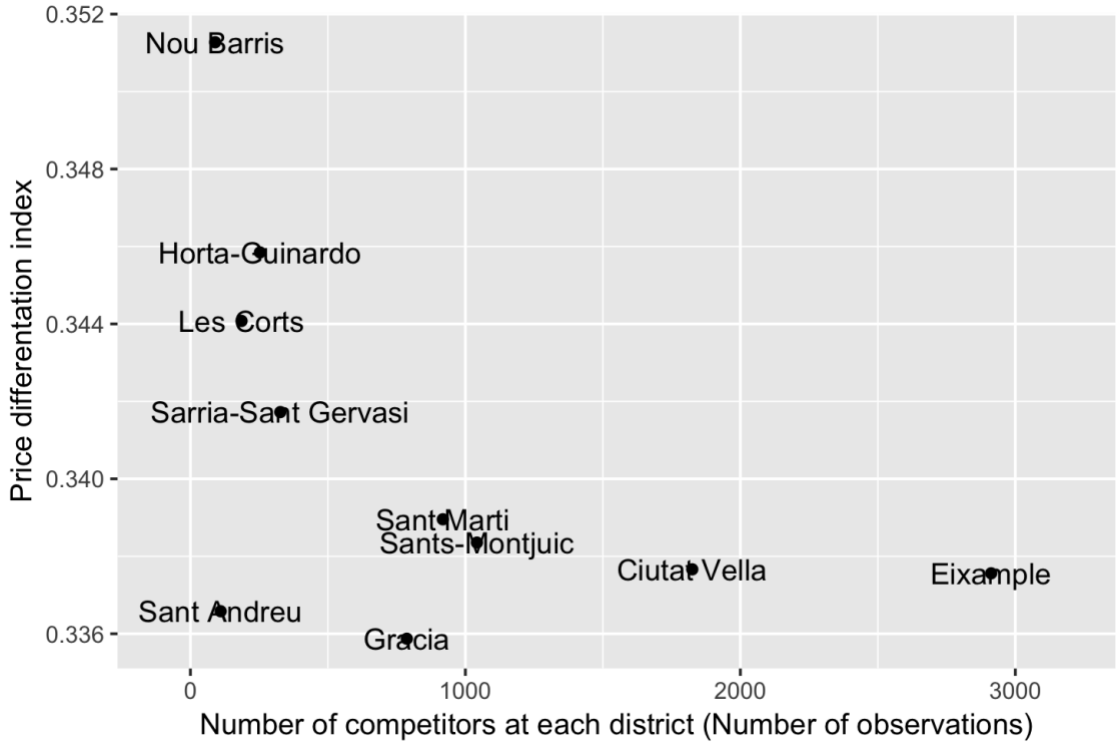


Figure 3.15: Number of competitors and price differentiation index for each district of Barcelona

Now that we have sketched out the competition level of Airbnb listings, we can delve into interpreting the BNN through the ARD Bayesian-hyperparameters and also the interpretation layer, because with that we will understand what defines the price of an Airbnb apartment.

Before analyzing the expected value of the posterior distribution of the ARD Bayesian-hyperparameters we have assessed the convergence of their chains because, if they do not have converged, then we cannot interpret those parameters. However, we have not included

the plots because the number of inputs nodes (139) is too high. Moreover, since the number of nodes is too high, we will only sketch out the most important ones for both the numeric and the categorical variables.

For the numeric variables the most important one is the number of guests that can be included in a reservation (variable *accommodates*), with an expected Bayesian-hyperparameter of 0.8. This is an obvious result, since the price of the listing is the price of renting the apartment for a night, and not the price for each accommodate. The following relevant variables have been the number of minimum nights allowed and the variable that captured how many listings has published in Barcelona the same host, with associated values of 0.73 and 0.6974. This information is meaningful because we can inform the user of the pricing model that, if he/she changes the minimum length of the stay, that will allow him to fix a different price for his/her apartment. However, it is impossible with just this Bayesian-hyperparameter to interpret how those variables are affecting the price of the apartment.

On the other extreme, those variables that are less important when predicting the price of the listing are the number of beds per accommodates (0.1798) and the maximum nights allowed (0.2681). Therefore, maybe it would be interesting to try to fit a BNN without those variables, because the less questions the pricing model asks, the better, because the user can employ it in a quicker way.

For the categorical variables, the two most important modalities correspond to the variable that defines the type of room that is the listing. In particular, the modalities Shared room and Private room (the baseline category was entire apartment) present an associated expected value of the relevance coefficient of 1.34 and 1.33. The following most important variable is the binary *has_cleaning_fee*, with an associated coefficient of 1.12. All these results seem reasonable, since the three different types of room are three different substitutive markets and, for the cleaning fee, because we included it in the final price. Therefore, it is telling us that those listings with cleaning fee do not compensate by reducing the price of the listing, compared to their competitors. Finally, the last most relevant category is the one associated with those apartments in which the host does not have a response time, meaning that he/she has become host recently. The coefficient is of 0.91. These would mean that the fact of being an inexpert host in Airbnb causes the price to vary, which is obvious because those host do not know well the behavior of the market. However, after implementing this pricing model, we should expect a reduction of the effect of this variable in the final price of the listings. After this category, the associated parameters are less than 0.63, so they are pretty far from the ones analyzed and they will not be explained.

However, we will just sketch out the input nodes associated to amenities with the highest and lowest relevance Bayesian-hyperparameters. In particular, we believe that these are useful because they are elements that the supplier can decide to include or not, so providing the user a list of the most important amenities (and the less important ones) in order to affect the price of the apartment is useful, because it can help him/her take decisions about possible investments and changes in the apartment. For the most important ones we find the fact of having a Pool (relevance parameter of 0.62), bed linens (0.55), Kitchen (0.54), Internet (0.53), Luggage drop-off (0.46) and First aid kit (0.46). On the other hand, the less relevant ones are Cooking basics (0.18), Laptop friendly workspace (0.18), Wide hallway clearance (0.20), toilet (0.21) and being greeted by the host (0.23). Some of this can be confusing, like the toilet not being relevant, but the thing is that the suppliers are not forced to state all the amenities of the apartment and, therefore, they do not say that there is toilet because it is obvious since it can be seen in the images attached to the listing.

Now that we have summarized the most important results from the ARD Bayesian-hyperparameters, we can delve into the interpretation layer. Just as a way to connect ARD with the interpretation layer we present in Figure 3.16 a scatterplot with the expected value of each individual (i.e. average value of the simulations) in each latent variable, classifying the listings according to the type of room that they are, since in ARD we have seen that it is the most relevant feature.

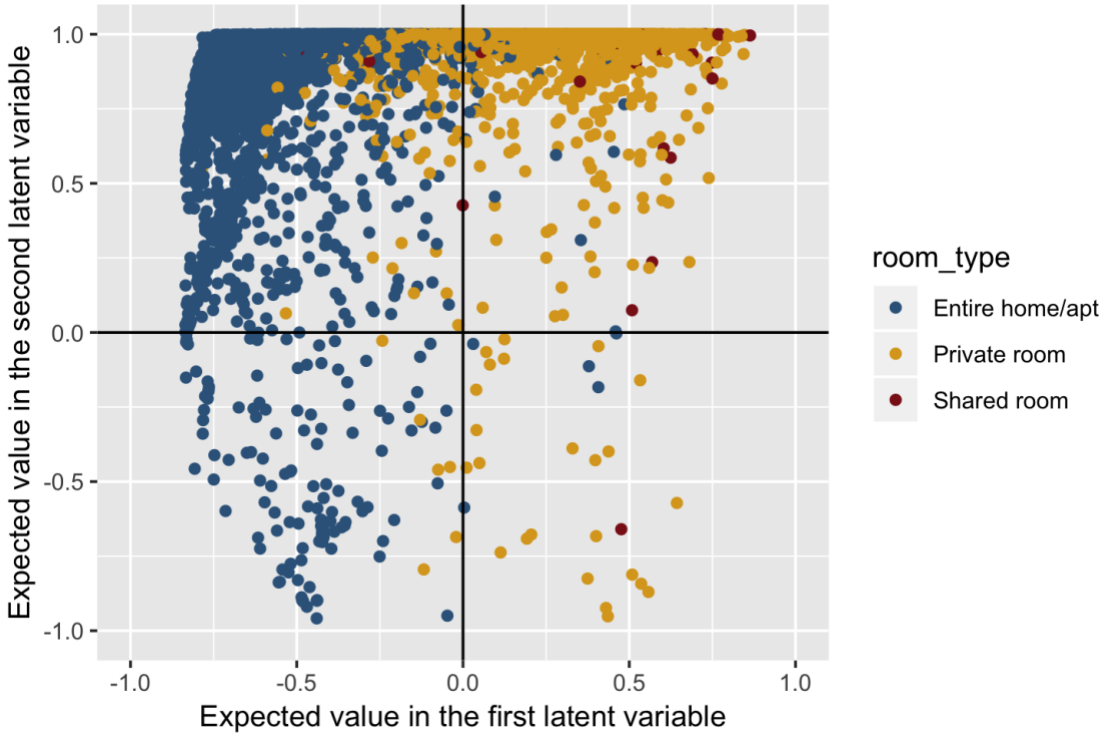


Figure 3.16: Scatterplot of the expected value of each individual at each latent variable according to the type of room that is published.

As it should be expected, this variable *room_type* defines how the values are displayed in this plane, because we know that it is a variable with a high influence on the price of the listing, so it must determine the latent variables. Thanks to this we can see that the first latent variable is somehow related to this positioning of the three markets, being on the rights the listings with less privacy (shared rooms) and on the left the ones with most privacy (entire home for the customer). In order to identify the second latent variable, we have plotted each relevant variable according to ARD in this plane, just as we did with *room_type*. What we encountered is that the variables that define that axis are, mainly, if the apartment has Bed linens and also the weekly and monthly discounts. In Figure 3.17 we present the variable Bed linens and in Figure 3.18 the weekly discount.

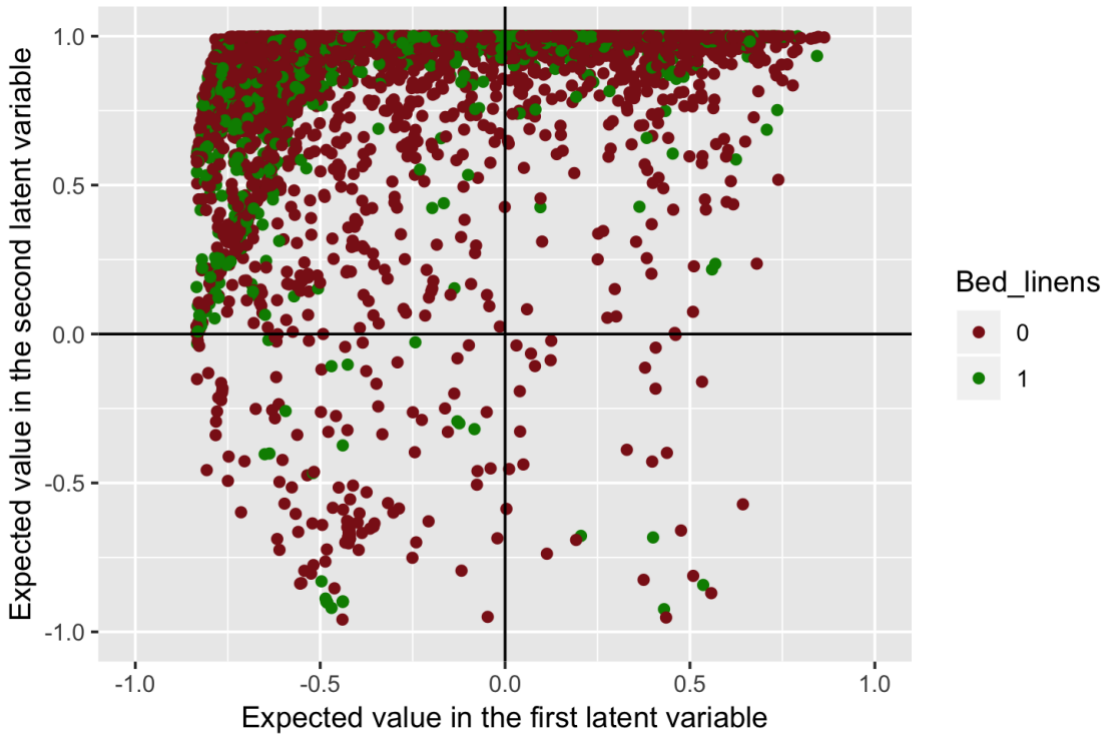


Figure 3.17: Scatterplot of the expected value of each individual at each latent variable according to if the listing specifies if it has bed linens or not.

The main conclusion that is obtained from Figure 3.17 is that in the bottom part of the plane the number of listings with Bed linens is scarce. In Figure 3.18, we can see that in that there are not apartments with weekly discounts in the bottom part of the plane. Therefore, this is telling us that this second latent variable may be related to the attitude of the host towards the listing. In particular, having Bed linens requires the host to clean them after the guests leaves and, not imposing any discount would mean that the host is not analyzing any strategy for long term stays or, in other words, that he/she does not care that much about renting the listing. Therefore, we can say that this second latent variable measures the effort/attention of the host invested in the apartment.

Now that we have given the latent variables some semantic, we can expand this analysis by plotting some summarizing value of the relation between each predictor and the latent variables. For the numeric predictors we plot the Kendall's τ coefficient between the numeric predictors and each simulation of the latent variables. This plot is shown in Figure 3.19.

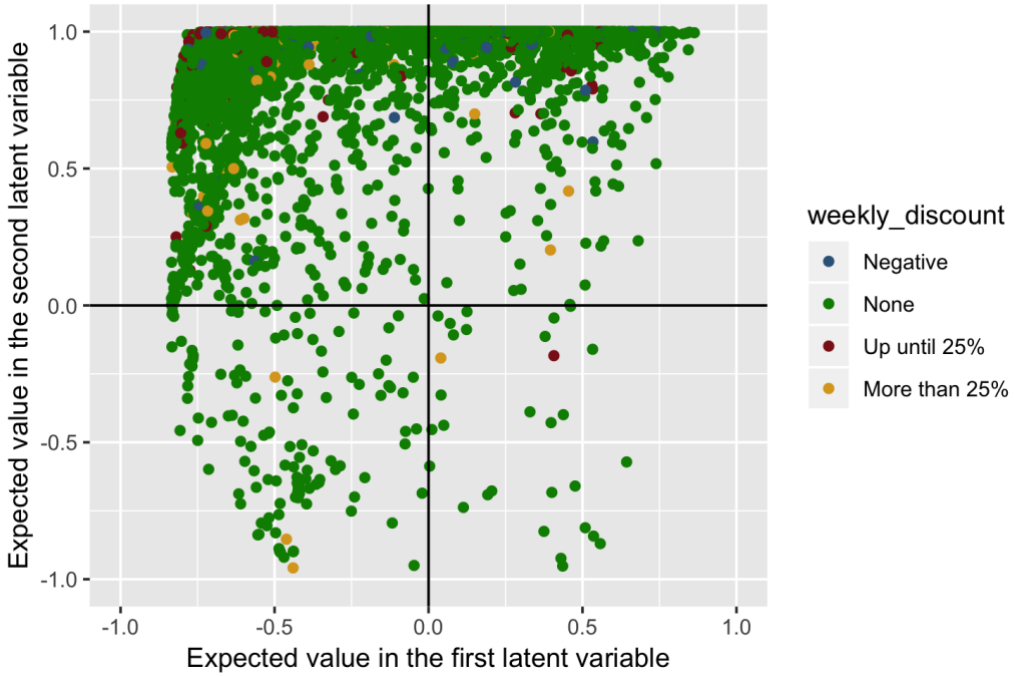


Figure 3.18: Scatterplot of the expected value of each individual at each latent variable according to the weekly discount specified for the listing.

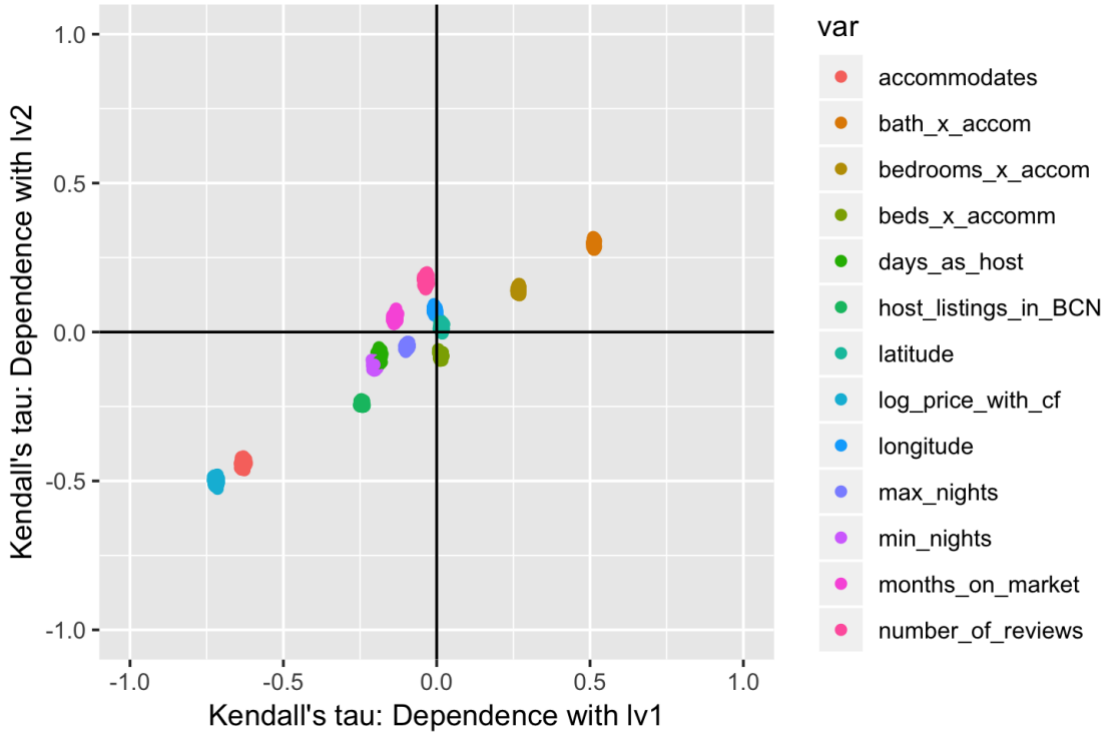


Figure 3.19: Kendall's τ coefficient between each numeric predictor and all the simulations from the two latent variables.

According to this plot, the most important numeric variable, as it was stated by ARD is the number of accommodates since is the one that is further away from the origin of coordinates and, as a consequence, has a higher dependency with the latent variables. It is positioned in the third quadrant, meaning that has negative dependency with both the first and the second latent variables. This tells us that the lower the privacy of the apartment (i.e. left part of the first latent variable), since the Kendall's τ is negative, the higher it will be the number of accommodates, which is reasonable because those listings that are entire homes are the ones that can accommodate more guests of the same reservation. Moreover, this is also telling us that, the apartments with more accommodates are usually those in which the host spends less effort on it, because those two variables (accommodates and second latent variable) have a negative Kendall's τ. Moreover, since the price of the listing (*log_price_with_cf* in the plot) is at the same position in the plane than the number of accommodates, we can say that those two variables are positively related, meaning that the more accommodates, the higher the overall price per night, which is obvious.

Even though meaningful conclusions can be extracted from this plot, it is also clear that there some contradictions with the results obtained from ARD. In particular, according to ARD Bayesian-hyperparameters the less important variable was *bedrooms_x_accomm*, while in this plot this variable is not at the origin of coordinates, meaning that it seems to define the latent variables. However, its position is meaningful since it is positioned with high values of the first latent variables, meaning that in those apartments with less privacy (i.e. shared rooms and private rooms) there are more actual beds for each accommodate, which is obvious because in apartments with higher privacy (i.e. entire homes) in which several known accommodates make a reservation they usually share double beds or couches, because they know each other.

Finally, the last conclusion that can be extracted from this is that the latitude and the longitude are placed at the origin of coordinates, meaning that maybe they are not useful to define the latent variables. According to the semantic that we have given to the latent variables, it seems sensible that geolocation does not affect them. Moreover, it could be interesting to remove this variables from our BNN because they are not affecting the latent variables and, therefore, maybe they are not relevant for the price of the apartment. However, removing those observations would be contradictory with ARD, because they were not the less relevant ones.

The last thing that will be analyzed is the relationship of the categorical predictors with the latent variables, just as a way to determine which are the most relevant ones for the prediction and understand what is driving our prediction or, in other words, what causes the price of a listing. As explained in subsection 6.3.2, in order to identify a variable as relevant, the different

categories of that variable must be scattered and separated in the plane formed by the two latent variables. However, there are a total of 228 modalities and, for each of them we should plot the 300 simulations to see the areas of probabilities. Since plotting all those variables would be uninformative, we will just plot some of the ones that have defined our latent variables (i.e. *room_type* and *weekly_discount),* some others that seem to be relevant and, furthermore a variable that does not have any impact to the response variable.

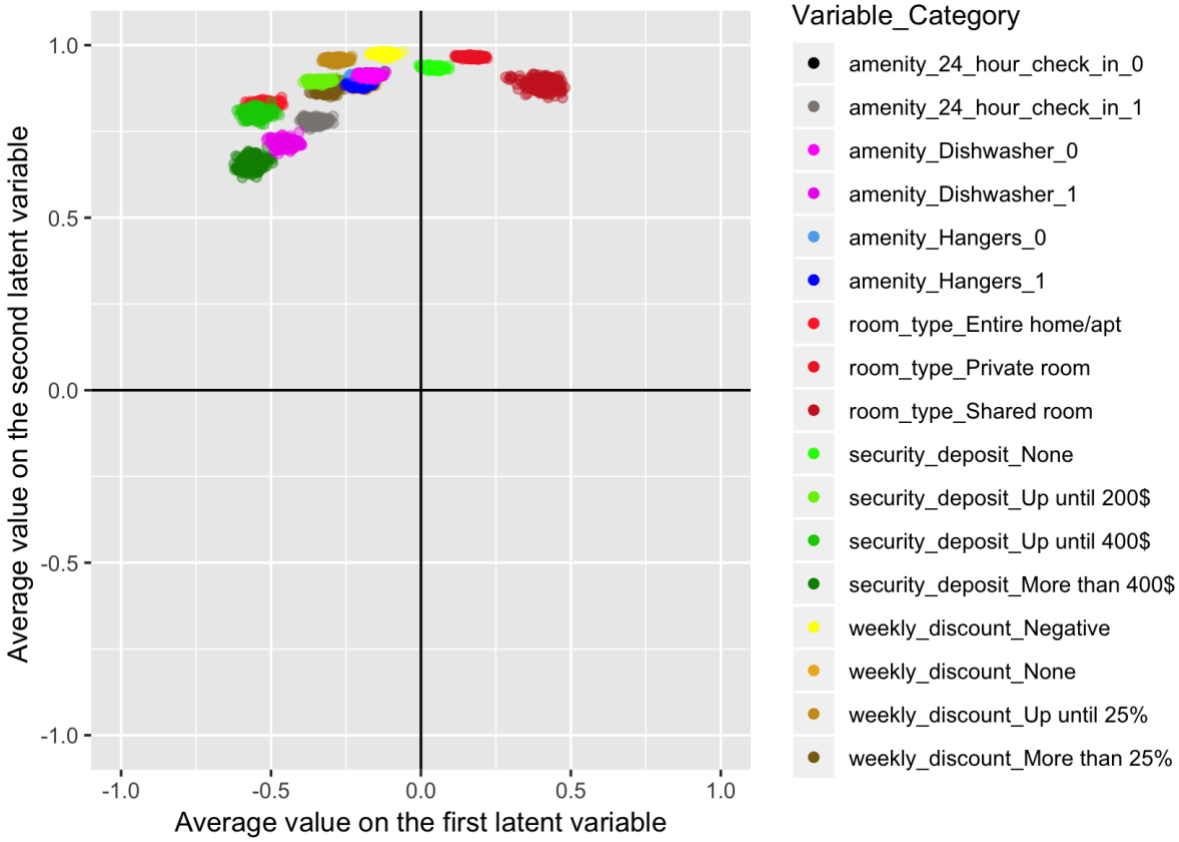


Figure 3.20: Average value of the latent variables in each modality of the categorical variables

As it should be expected, both the categories of the room type and weekly discount are in different regions of the plane, meaning that these variables are meaningful for the prediction since the latent variable take different values at each category. Moreover, this plot also remarks the relevance of imposing a security deposit on the apartment rental. In particular, we see that those categories with higher value of a security deposit are on the left part of the plane and, moreover, they are at a lower part of the plane than the other categories (this is the same direction described by the response variable, *log_price_with_cf,* in Figure 3.19), meaning that the listings that include security deposit are usually those with a higher price. The same thing happens with the Dishwasher and the 24-hour check-in amenities[9]: the

[9] In the plot it is difficult to see it, but the category amenity_24_hours_check_in_0 is just behind the amenity_dishwasher_0. Similarly, the category weekly_discount_None is at the same region than amenity_Hangers_1.

category relating to those listings that include those characteristics is positioned more on the left and on the bottom part than the baseline category, meaning that customers tend to value those amenities and, therefore, they are willing to pay more for the listing. However, for the hangers the two clouds of points intersect with each other, which means that this variable is not relevant to determine the response variable. This could be done for all the categorical variables of the model but, since in this project our goal was to just describe the potential of the BNN, we will not delve into it.

Just to prove that adding a Dishwasher allows the host to post a higher price without losing clients (in other words, the market price becomes higher) we will take all the listings from the training set and compute their predicted market price but fixing that any of them has Dishwasher. Afterwards, we will predict the market price as if all of them had Dishwasher (i.e. changing the dummy variable to ones). We will do the same for the hangers, just to see how meaningful are the conclusions extracted from the interpretation layer.

Variable	Average market price without amenity ($)	Average market price with amenity ($)
Dishwasher	85.99	94.07
Hangers	87.8	86.72

Table 3.5 Average market price for the observations from the training set with and without dishwasher/hangers.

Changing the fact of including or not hangers, the average price only changes one dollar while including or not dishwasher changes the predicted value by 8 dollars, which clearly indicates the validity of the conclusions extracted from the interpretation layer. In fact, ARD also stated that something like this would happen, because while the input nodes that indicates that a listing has hangers has an associated coefficient of 0.34, the one for the Dishwasher has a coefficient of 0.42. However, one should have in mind that not in all apartments, adding a Dishwasher will have an effect, this is just an average effect. The positive aspect is that the user, employing the pricing model will be able to compare his/her apartment with an without Dishwasher.

Therefore, the final commentary of this section is the fact that, thanks to our BNN we have discovered that what is mainly driving the price of the listings is both the privacy that it gives to the customer and, moreover, the effort/attention given by the host on the apartment. Furthermore, we have been able to assess the relevance of each predictor, thanks to ARD and, thanks to the interpretation layer, we have been able to determine how some predictors are affecting the price of a listing.

Conclusions

The uppermost underlying conclusion of this thesis is that, thanks to ICT, there are opportunities for Data Science to reduce information costs in many markets, because there is a huge amount of secondary-generated data waiting to be processed and converted into useful information. As explained during the first chapter, seizing these opportunities is important for our society, since a reduction on the information costs improves the market allocation of resources and, as a consequence, improves the overall welfare of our society. In this project, we focused on a particular case, the P2P OM platforms and, through the results obtained in the example of the third chapter, we concluded that exploiting those opportunities translates into useful tools able to solve real problems.

In particular, in the example of the third chapter we showed that, thanks to the pricing model devised, the user not only knows in just a few seconds the market price for the apartment that he/she is willing to publish but, moreover, he/she has been given the opportunity to easily explore the market. For instance, he/she can see how including a new amenity can affect the price of the apartment, so it can help him/her to consider possible investments on the apartment. All this facilities that the pricing model give the suppliers the chance to provide apartments (in the case of Airbnb) that fit the demand preferences and, therefore, increase the amount of transaction conducted. Therefore, the conclusion that we want to remark is that, thanks to the third chapter, we have been able to envision everything that was explained during the first chapter about how reducing information costs can improve the market efficiency and, moreover, that we can interpret the BNN in order to understand the behavior of the market price and, also, study the level of competition in several sectors defined by the features.

Nevertheless, this positive impact on the market efficiency must be proven and, to do that, we propose applying this pricing model in a real dataset from a P2P OM platform and conduct an experiment by offering only to a sample of suppliers this new tool while another sample is kept without it and, afterwards, compare the amount and value of transactions conducted by each group. However, since this experiment would take a remarkable amount of time, since we do not have access to a real transaction-based dataset from a P2P OM and, moreover, since we do not have capabilities to interfere in the decisions of a P2P OM platform, this was not included in the project, but we encourage the reader to think that quantifying the impact of a pricing model is possible.

Another crucial conclusion extracted from this thesis is that Data Science must be enhanced with new tools, algorithms and models in order to be able to fully seize these new

opportunities that are appearing. In particular, we conclude this because in this project, as it is shown in the third chapter, we have been able to devise a model (BNN) more suitable than the statistical linear model and conventional Artificial Neural Networks, since in that example that these extra capabilities of the BNN helped to better seize the opportunity of P2P OM platforms, yielding a more accurate pricing model for the users.

This last conclusion is connected to another conclusion which relates to the fact of how we obtained this new model, the BNN. In particular, the conclusion that we extract is the outstanding potential of combining the knowledge from Statistical Science and Machine Learning, which is one of the main goals of the thesis. In fact, we successfully connected those two fields two times during this thesis. The first one relates to devising a Neural Network as a parametrical statistical model, which has allowed us to obtain a model more able to capture the behavior of what we are observing, both because it yields a more accurate punctual prediction and because the intervals offered are, at the same time, narrower than the ones obtained through a linear model and able to reflect the correct variability of the response variable, as it can be concluded from the third chapter. The other successful connection between the two fields has been using DoE to learn about the effect of ML-hyperparameters such as the regularization constant or the Neural Network architecture on the predictive performance of an ANN. In fact, in the example of the third chapter it can be seen that thanks to employing DoE to find a suitable architecture for an ANN allowed us to find a region of ML-hyperparameters with higher out-of-sample prediction that, with other techniques like CV we would not have been able to find. However, and this is another relevant conclusion extracted from this thesis, in order to combine Statistical Science and Machine Learning we will usually be required to work with a more abstract and capable approach for statistics, which is the Bayesian framework.

If we delve into the results obtained by the proposed BNN, several relevant conclusions can be extracted. The first one, and related to what has been previously stated, is the need of thinking about BNN completely from the statistical point of view and, in particular, having in mind that there is no need for explicit regularization in a BNN, since it is automatically assessed in the posterior distribution of the weights. This has been seen both in the example on the annex and in the third chapter, since all the applied BNNs in this project do not include explicit regularization and they have been able to outstandingly generalize the prediction. A second relevant conclusion relates to the capabilities' enhancement of our BNN. In particular, since we based our methodology to fit it with MCMC methods and MCMC is a constantly growing field, we expect our BNN to become more suitable to be applied as time passes by.

On top of these conclusions, there are three more essential conclusions about how BNNs are more suitable than ANNs that are extracted from the results obtained in the third chapter and all the experiments conducted, like the one in the annex. The first one relates to the fact that the architecture chosen for our BNN seems to be less important in the final predictive performance than in an ANN and, moreover, that in a BNN the number of ML-hyperparameters that must to be fixed is smaller, since ML-hyperparameters like the regularization constant (or learning rate depending on the algorithm), batch size or number of times that an individual is used to train the weights of the Neural Network do not exist in BNN. Therefore, this first conclusion is that finding a suitable architecture for a BNN is not as grueling as doing the same for an ANN. A second relevant conclusion is that BNNs are more general than ANNs and, as a consequence, they are able to include extensions, like ARD or an interpretation layer, that allow us to "cast light on the black box", meaning that interpreting a BNN is easier than doing the same for an ANN. However, we have not been able, due to the current extension of the thesis, to assess how introducing these extensions affect the predictive performance of our BNN, since we have encountered contradictory results in different experiments. Finally, one last conclusion about the BNN capabilities is that it is able to offer grounded intervals for the response variable (i.e. market price) and the predicted value (i.e. expected market price) according to an underlying model, while bootstrapping ANN usually only allows to deliver approximated intervals for the predicted value (i.e. expected market price). Instead, if one wants to offer an interval for the response variable (i.e. market price) through bootstrap, he has to rely on a probability distribution and, since we do not know the MLE because the derivatives are intractable, we need to use some approximated estimator, which causes the bootstrap intervals to be less reliable than the ones obtained in the BNN.

However, it is important to remark the fact that, even though BNN present all those enhancements with respect to ANN and the linear model, we must pay a relevant price for it: a significantly higher time and computational cost is required to obtain a BNN. In particular, for an ANN in the third chapter we needed around two minutes, while for the BNNs the required time to simulate from the posterior distribution was of 11 hours.

As a final and wrapping conclusion, we want to remark all the concepts and techniques that are new in this thesis (i.e. the contributions of this thesis) in a summarized way, just as a way to help the reader realize that merging Statistical Science and Machine Learning is a grueling task, since as it can be seen, the list is rather big:

1) Dealing with Neural Networks as a parametric statistical model and, in particular, using the Bayesian approach with either uninformative improper priors (baseline BNN) or

hierarchical Bayesian models (proposed BNN with ARD), but without explicit regularization Bayesian-hyperparameters, which is what differs our proposal from previous BNNs. Moreover, we propose a particular structure based on a few number of nodes per layer, an interpretation hidden layer and, finally, a maximum of three hidden layers.

2) A complete methodology on how to fit the devised BNNs, which includes several new concepts and techniques explained in the following elements of this list. In a summarized way, we propose a renewed point of view on how to obtain the weights of a BNN.
3) Employing MCMC methods to obtain the posterior distribution of the weights. Even though many authors, mainly Neal, proposed this, what is new in this project is how we apply them in order to capture the multimodality of our posterior distribution which is, basically, starting several parallel chains that encapsulate a different local posterior probability mode each. With this, we aim to overcome other methods (like Gaussian approximations or VI) that only focus on a particular mode. Moreover, there are two more elements that are new in this thesis about how we are implementing MCMC methods for BNNs: The starter values for the chains and the particular MCMC method used (NUTS).
4) Developing a technique, that we called Multidimensional Convergence of MCMC methods, in order to determine the entrance to the stationary state of the MCMC chains in complicated Bayesian models, which is a required step to validate the use of MCMC methods. In particular, we look for the entrance of the model log-likelihood chain to a steady state and, this technique, has been proposed by other authors. However, what is really new in this project is the rationale about why the log-likelihood chain is useful to check the entrance to the stationary state (demonstrated in two simulated examples) and, moreover, its implementation for BNNs.
5) Finding a suitable architecture for our BNN with a more practical and applied method than the pure Bayesian approach, because the latter has serious implementation problems with MCMC. In particular, our proposal is to find the architecture using ANNs to speed up the process and DoE instead of CV. The main advantages of working with DoE are that less experiments must be conducted but each of them is more informative, that we use all the information provided by the experiments (not just the minimum), that we take into account the variability of the error calculated per each architecture so we can be more confident about the found optimal architecture and, finally, that we can discover new regions of the ML-hyperparameters space in which the predictive performance is higher.

6) Devising a validation procedure, based on techniques for the linear model, in order to decide whether if our BNN is acceptable or not. This includes the checks about the normality and homoscedasticity hypotheses and the treatment of the uncertainty.
7) Adapting the work about ARD of previous authors in our proposed BNN. In particular, what differences our approach about it is that we use an improper hyperprior and, moreover, that we discuss that categorical predictors are not comparable with numerical ones, since the units of measurement are not the same.
8) Imposing an interpretation layer for BNNs, as explained in the first item of this list, in order to understand, taking account the underlying variability, which explanatory variables are driving our prediction and how they are doing it.
9) Devising a bootstrap method in order to obtain approximated ANN confidence intervals of the response variable, and not only the predicted value. This was not required for the BNN, but we developed it just to compare the intervals of our BNN with some "established" technique like bootstrap. Due to its secondary relevance, this was included in the example in the annex, but not in the third chapter of the thesis.

However, we conclude that developing all these techniques has been a thrilling activity and, seeing the results in the third chapter, that it has been worth the effort not just because in statistical terms we have obtained better results, but because these results matter in the development of our society, which was the underlying goal of the thesis.

Next steps

Due to the fact that the obtained results have been astonishing, we want to finish this thesis by mentioning some topics about which it would be interesting to conduct research. As before, we present them as a list just facilitate its understanding:

1) Quantitatively comparison of BNN obtained with VI, Stochastic gradient Langevin Dynamics and the Hybrid Monte Carlo. The latter is the one used in this project.
2) Developing Recurrent Neural Networks from the statistical point of view, since in this project the focus was on feed-forward Neural Networks.
3) Developing some measurable indicator of the convergence for the log-likelihood chains, as a way to be endowed with a more robust method than assessing it through visually tracking the chain.
4) Further analysis of how ARD affects the predictive performance of the BNN. In the same area, it would be also interesting to further analyze the effect on predictive performance of removing some predictors through the Kendall's τ coefficient with the latent variables of the Neural Network.

5) Expanding the DoE methodology to find suitable ML-hyperparameters for other Machine Learning algorithms.
6) Approaching BNNs with a superior infrastructure provided by cloud computing services. In particular, with a highly parallelizable cluster more MCMC chains could be started and, therefore, the samples would be more representative of the jointly posterior distribution and, moreover, the DoE process to find a suitable architecture could be parallelized and, as a consequence, done with BNNs instead of ANNs.

Bibliography

Akerlof, G., & Shiller, R. (2010). Animal spirits: How human psychology drives the economy, and why it matters for global capitalism. Princeton University Press.

Anderson, D., & McNeill, G. (1992). Artificial neural networks technology. Kaman Sciences Corporation, 1-83.

Banks, M. A. (2008). The Second Wave. In On the way to the web. Apress.

Barber, D., & Bishop, C. M. (1998). Ensemble learning in Bayesian neural networks. NATO ASI SERIES F COMPUTER AND SYSTEMS SCIENCES, 215-238.

Baum, E. B., & Wilczek, F. (1988). Supervised learning of probability distributions by neural networks. Neural information processing systems, (pp. 52-61).

Bauman, Z. (2000). Liquid modernity. UK: Polity Press.

Beck, P. (2017). The feasibility of measuring the sharing economy: November 2017 progress update. UK: Office for National Statistics.

Belk, R. (2014). You are what you can access: Sharing and collaborative consumption online. Journal of Business Research, 1595-1600.

Bellinger, G., Castro, D., & Mills, A. (2004). Data, information, knowledge, and wisdom.

Besag, J., Green, P., Hidgdon, D., & Mengersen, K. (1995). Bayesian computation and stochastic systems (with discussion). Statistical Science, 3-66.

Bishop, C. M. (1995). Neural Networks for pattern recognition. Oxford university press.

Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational inference: A review for statisticians. Journal of the American Statistical Association , 859-877.

Bolstad, W. M., & Curran, J. M. (2016). Introduction to Bayesian statistics. John Wiley & Sons.

Botsman, R. (2010). The Case for collaborative consumption. TEDxSydney, Sydney.

Botsman, R., & R., R. (2010). What's Mine is Yours: how collaborative consumption is changing the way we live. HarperCollins.

Box, G. E. (2005). Statistics for experimenters: design, innovation, and discovery. New York: Wiley-Interscience.

Box, G. E., & Draper, N. R. (1987). Empirical Model-Building and Response Surfaces. Wiley.

Buntine, W. L., & Weigend, A. S. (1991). Bayesian back-propagation. Complex systems, 603-643.

Campbell, J. A., & Thomas, H. B. (1981). The videotex marketplace: A theory of evolution. In Telecommunications Policy (pp. 111-120).

Castells, M. (1996). The information age: Economy, society, and culture. Volume I: The rise of the network society.

Catledge, L. D., & Pitkow, J. E. (1995). Characterizing browsing strategies in the World-Wide Web. In Computer Networks and ISDN systems (pp. 1065-1073).

CompuServe. (1984). CompuServe Electronic Mall. Retrieved from https://www.youtube.com/watch?v=k-oBJml1mL0

Cox, R. T. (1946). Probability, frequency and reasonable expectation. American Journal of Physics, 1-13.

Coyle, D. (2014). Beyond GDP. Foreign Affairs.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. In Mathematics of control, signals and systems (pp. 303-314). Springer.

December, J., & Randal, N. (1994). The World Wide Web (Unleashed).

Doucek, P. (2010). Human resources in ICT–ICT effects on GDP. Doucek, P. (2010). Human resources in ICT–ICT effects on GDP. IDIMT-2010: Information Technology–Human Values, Innovation and Economy, (pp. 97-105).

Duane, S., Kennedy, A. D., Pendleton, B. J., & Roweth, D. (1987). Hybrid Monte Carlo. Physics Letters B, 216-222.

Dybowski, R., & Roberts, S. J. (2001). Confidence intervals and prediction intervals for feed-forward neural networks. In Clinical Applications of Artificial Neural Networks (pp. 298-326).

El-Jaroudi, A., & Makhoul, J. (1990). A new error criterion for posterior probability estimation with neural nets. IJCNN International Joint Conference on Neural Networks.

Ellison, N. B. (2007). Social network sites: Definition, history, and scholarship. Journal of computer-mediated Communication, 210-230.

European Comission. (2017). Assessing the size and presence of the collaborative economy in Europe. EU publications.

European Ecommerce, A. (2015). European B2C E-commerce Report 2016.

Farmer, R., & Guo, J. (1994). Real business cycles and the animal spirits hypothesis. Journal of Economic Theory, 42-72.

Fragoso, T. M., Bertoli, W., & Louzada, F. (2018). Bayesian model averaging: A systematic review and conceptual classification. International Statistical Review, 1-28.

Fraiberger, S. P., & Sundararajan, A. (2017). Peer-to-Peer Rental Markets in the Sharing Economy. NYU Stern school of Business research paper.

Franke, J., & Neumann, M. H. (2000). Bootstrapping neural networks. Neural computation, 1929-1949.

Freitas, J. D., Niranjan, M., G., H., A., & Doucet, A. (2000). Sequential Monte Carlo methods to train neural network models. Neural Computation, 955-993.

Hall, S., & Pennington, J. (2016). How much is the sharing economy worth to GDP? World Economic Forum website.

Hamari, J., Sjöklint, M., & Ukkonen, A. (2016). The sharing economy: Why people participate in collaborative consumption. Journal of the Association for Information Science and Technology, 2047-2059.

Heaton, J. (2008). Introduction to neural networks with Java.

Hebb, D. O. (1949). The organization of behavior: A neuropsychological theory.

Hinton, G. E., & van Camp, D. (1993). Keeping neural networks simple by minimising the description length of weights. Conference On Computation Learning Theory.

Hoffman, M. D., & Gelman, A. (2014). The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. Journal of Machine Learning Research, 1593-1623.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. In Neural networks (pp. 251-257). Elsevier.

Jaynes, E. T. (1986). Bayesian methods: general background. In J. H. Justice, Maximum Entropy and Bayesian Methods in Applied Statistics (pp. 1-25). Cambridge University Press.

Jefferys, W. H., & Berger, J. O. (1992). Ockham's Razor and Bayesian Analysis. American Scientist, 64-72.

Jensen, R. (2007). The digital provide: Information (technology), market performance, and welfare in the South Indian fisheries sector. The quarterly journal of economics, 879-924.

Jericho, G. (2016). The dark side of Uber: why the sharing economy needs tougher rules. The guardian.

Keynes, J. (1936). General Theory of employment, interest and money. London: Macmillan.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. IJCAI.

Kruschke, J. (2014). Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan. . Academic Press.

Lampinen, J., & Vehtari, A. (2001). Bayesian approach for neural networks—review and case studies. Neural Networks, 257-274.

Larsen, J., & Hansen, L. K. (1994). Generalization performance of regularized neural network models. In Neural Networks for Signal Processing (pp. 42-51).

Laurell, C., & Sandström, C. (2017). The sharing economy in social media: Analyzing tensions between market and non-market logics. In Technological Forecasting and Social Change (pp. 58-65). Elsevier.

Lawrence, S., & Giles, C. L. (2000). Overfitting and neural networks: conjugate gradient and backpropagation. In Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference.

Lee, S. Y., Gholami, R., & & Tong, T. Y. (2005). Time series analysis in the assessment of ICT impact at the aggregate level–lessons and implications for the new economy. In Information & Management (pp. 1009-1022). Elsevier.

Lewis, D. D., & Jones., K. S. (1996). Natural language processing for information retrieval. Communications of the ACM, 92-101.

Lipsey, R. G., & Lancaster, K. (1956). The general theory of second best. The review of economic studies, 11-32.

Lu, Y., Zhao, L., & Wang, B. (2010). From virtual community members to C2C e-commerce buyers: Trust in virtual communities and its effect on consumers' purchase intention. In Electronic Commerce Research and Applications (pp. 346-360).

MacKay, D. J. (1991a). Bayesian Interpolation. Neural Computation, 415-447.

MacKay, D. J. (1991b). A Practical Bayesian Framework for Backprop Networks.

MacKay, D. J. (1992a). Bayesian model comparison and backprop nets. Advances in neural information processing systems.

MacKay, D. J. (1992b). The Evidence Framework Applied to Classification Networks. Neural Computation.

Malhorta, A., & Van Alstyne, M. (2014). The dark side of the sharing economy… and how to lighten it. Communications of the ACM, 24-27.

Mankiw, N. G. (2011). Principles of microeconomics (5th edition). South-Western Cengage Learning.

Mas-Colell, A. (1998). On the theory of perfect competition. Econometric Society Monographs, 16-32.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 115-133.

McGillivray, M., & White, H. (1993). Measuring development? The UNDP's human development index. Journal of international development, 183-192.

Miazhynskaia, T., & Dorffner, G. (2006). A comparison of Bayesian model selection based on MCMC with an application to GARCH-type models. Statistical Papers, 525-549.

Minifie, J. (2016). Peer-to-peer pressure: policy for the sharing economy. Grattan Institute.

Moraru, M. (2008). E-commerce. Romanian Economic and Business Review, 44-50.

Müller, P., & Rios, D. (1998). Issues in Bayesian analysis of neural network models. Neural Computation, 749-770.

Neal, R. M. (1992). Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Dept. of Computer Science, University of Toronto.

Neal, R. M. (1993). Bayesian learning via stochastic dynamics. Advances in neural information processing systems.

Neal, R. M. (1995). Bayesian learning for neural networks.

Neal, R. M. (1996). Sampling from multimodal distributions using tempered transitions. Statistics and computing, 353-366.

OECD. (2017). OECD Digital Economic Outlook 2017. Paris: OECD Publishing.

Opper, M., & Haussler, D. (1991). Generalization performance of Bayes optimal classification algorithm for learning a perceptron. Physical review letters, 2677-2680.

Oram, A. (2001). Peer-to-Peer: Harnessing the power of disruptive technologies.

Paass, G. (1993). Assessing and improving neural network predictions by the bootstrap algorithm. Advances in Neural Information Processing Systems, (pp. 196-203).

Porat, M. (1998). The information economy: definition and measurement. In Rise of the knowledge worker (pp. 101-131). Butterworth-Heinemann.

Robert, C. P., & Casella, G. (2010). Introducing monte carlo methods with R. New York: Springer.

Roncaglia, A. (2006). The Wealth of Ideas. Cambridge University Press.

Rosen, S. (1974). Hedonic prices and implicit markets: product differentiation in pure competition. Journal of Political Economy, 34-55.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 386-408.

Rumelhart, D. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. nature.

Sarwar, M., & Soomro, T. R. (2013). Impact of Smartphone's on Society. European Journal of Scientific Research, 216-226.

Schor, J. (2016). Debating the sharing economy. Journal of Self-Governance & Management Economics.

Smith, A. (1776). Wealth of nations. London: W. Strahan and T. Cadell.

Stigler, G. J. (1957). Perfect Competition, Historically contemplated. Journal of Political Economy, 1-17.

Stigler, G. J. (1961). The Economics of Information. Journal of Political Economy, 213-225.

Stiglitz, J. E. (1989). Imperfect information in the product market. In Handbook of industrial organization (pp. 769-847).

Taeihagh, A. (2017). Crowdsourcing, Sharing Economies and Development. Journal of Developing Societies, 191-222.

Talarzyk, W. W., Widing, R. E., & Urbany., J. E. (1984). Videotext and consumer behavior. ACR North American Advances.

Thodberg, H. H. (1996). A review of Bayesian neural networks with an application to near infrared spectroscopy. IEEE transactions on Neural Networks, 56-72.

Tishby, N., Levin, E., & Solla, S. A. (1989). Consistent inference of probabilities in layered networks: Predictions and generalization. IJCNN International Joint Conference on Neural Networks. New York.

Vehtari, A., Sarkka, S., & Lampinen, J. (2000). On MCMC sampling in Bayesian MLP neural networks. Neural Networks.

Vivarelli, F., & Williams, C. (1997). Using Bayesian neural networks to classify segmented images. Artificial Neural Networks, Fifth International Conference.

Welling, M., & Teh, Y. W. (2011). Bayesian learning via stochastic gradient Langevin dynamics. 28th international conference on machine learning.

Wilson, J., Tyedmers, P., & Pelot, R. (2007). Contrasting and comparing sustainable development indicator metrics. In Ecological Indicators (pp. 299-314). Elsevier.

Zahraee, S., Rezaei, G., Memari, A., Afshar, J., & Mohd, J. R. (2013). Teaching Design of Experiment and Response Surface Methodology Using Paper Helicopter Experiment.

Annex

Example Box-Behnken design matrix

In DoE, the design matrix is the element that captures all the Experimental Conditions that will be tested. In order to obtain the design matrix of a Box-Behnken design one needs to select a minimum and maximum value for each X and also a central point which is the intermediate point between those two limits.

For instance, we would take for the number of nodes in the first hidden layer: 2,5 and 8. For the number of hidden layers, since we will only experiment with two or three, we can harness this information through the number of nodes in the second hidden layer. In particular, if we set the number of nodes in the second hidden layer at zero, then we state that our ANN will only have two hidden layers, the first and the third. Therefore, the range of values for the nodes in the second hidden layer would be, for instance: 0,5,10. The number of nodes in the third hidden layer will always be 2 so it will not be a changing factor and, of course, the two levels for the activation function would be *logistic* and *tanh*. Finally, we set our learning rate values to 0.01, 0.051 and 0.1. In Box-Behnken designs we still need to decide, apart from the range of the factors, the number of central-point experimental condition (EC) that will be conducted. If we decide to conduct two central-point EC and one replicate for each EC our design matrix would be:

Nodes hidden layer 1	Nodes hidden layer 2	Learning rate	Activation function
2	0	0.051	*Logistic*
2	10	0.051	*Logistic*
8	0	0.051	*Logistic*
8	10	0.051	*Logistic*
2	5	0.01	*Logistic*
2	5	0.1	*Logistic*
8	5	0.01	*Logistic*
8	5	0.1	*Logistic*
5	0	0.01	*Logistic*
5	0	0.1	*Logistic*
5	10	0.01	*Logistic*
5	10	0.1	*Logistic*
5	5	0.051	*Logistic*
5	5	0.051	*Logistic*

2	0	0.051	*tanh*
2	10	0.051	*tanh*
8	0	0.051	*tanh*
8	10	0.051	*tanh*
2	5	0.01	*tanh*
2	5	0.1	*tanh*
8	5	0.01	*tanh*
8	5	0.1	*tanh*
5	0	0.01	*tanh*
5	0	0.1	*tanh*
5	10	0.01	*tanh*
5	10	0.1	*tanh*
5	5	0.051	*tanh*
5	5	0.051	*tanh*

Annex Table 1: Box-Behnken design matrix with 3 numerical factors, one categorical and two central-point experimental conditions (i.e. all numeric factors at their central point).

Each row of the design matrix is an Experimental Condition (EC) and, for each row, an ANN should be fitted. The last two EC (for each activation function) are the central-point EC and it is an EC in which all factors are fixed to the central value from the range that we have provided. A Box-Behnken design can be modified by adding more central point EC, which means that more experiments must be conducted. Finally, if we wanted to add a replicate for each EC, we would need to duplicate this design matrix.

Summary of the most relevant results of our methodology through an example dataset

Introduction to the dataset

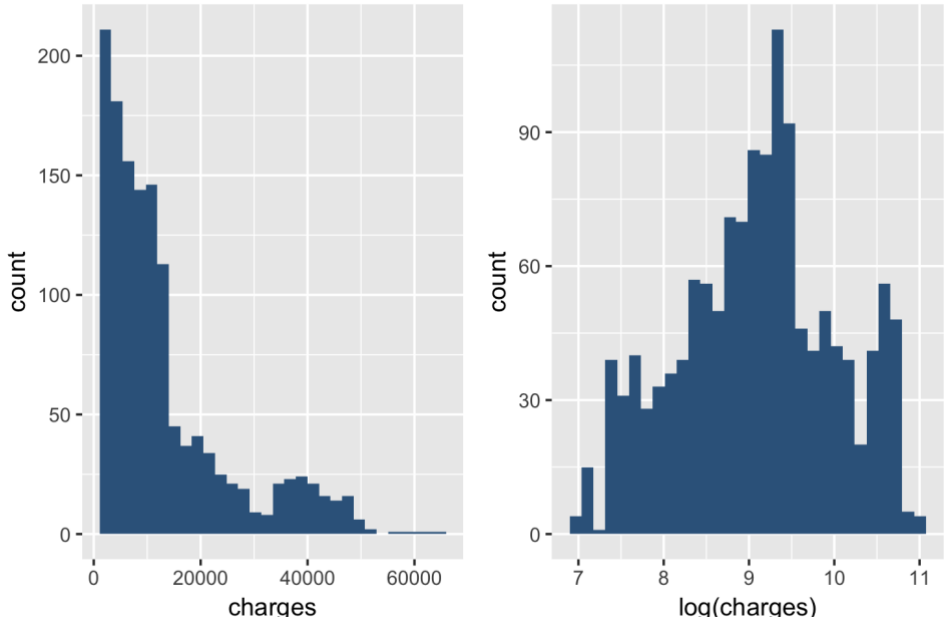
The dataset through which we are going to present the most relevant results that have guided our decisions about the proposed methodology can be obtained in https://www.kaggle.com/mirichoi0218/insurance/data. In particular, we have found it useful because it has a small number of observations and variables and, therefore, the simulation time to sample from the posterior distribution will be reduced. In particular, it contains information about 1,338 individuals and the goal is to predict the medical insurance charges applied to them through the predictors showed in Annex Table 2.

Another advantage of this dataset is that there was not any missing value and that there is a good balance between numeric and categorical predictors to test our BNN, so we have not

considered applying transformations to any predictor. Therefore, the only preprocessing action conducted was the one aiming to find outliers and preparing the response variable. In particular, for the response variable we observed the histogram in the left part of Annex Figure 1, so we decided to apply logarithm to it to facilitate its treatment through our models, which yielded the histogram in the right part of Annex Figure 1.

Variable	Definition	Type	Unit	Range values
age	Age of the client	numeric	years	(18,64)
sex	Gender	categorical		2 classes
BMI	Body Mass Index	numeric	kg/m^2	(15.96,53.13)
children	Number of own children	numeric	children	(0,5)
smoker	If the client smokes	categorical		2 classes
region	Region of residence	categorical		4 classes
charges	Applied charges	numeric	dollars	(1,121;63,770)

Annex Table 2: Simplistic metadata for the dataset example used to represent the results from the experiments.



Annex Figure 1: Histogram of the response variable (left) and its logarithm (right).

As explained in our pipeline, the next step is based on partitioning our data in training, validation and test set. We have decided to use a 60/20/20 repartition, so the number of observations are 803, 268 and 267, respectively. Moreover, we standardize the numeric variables, including the response variable, and we convert the categorical into dummies using contrast treatment.

Architecture selection

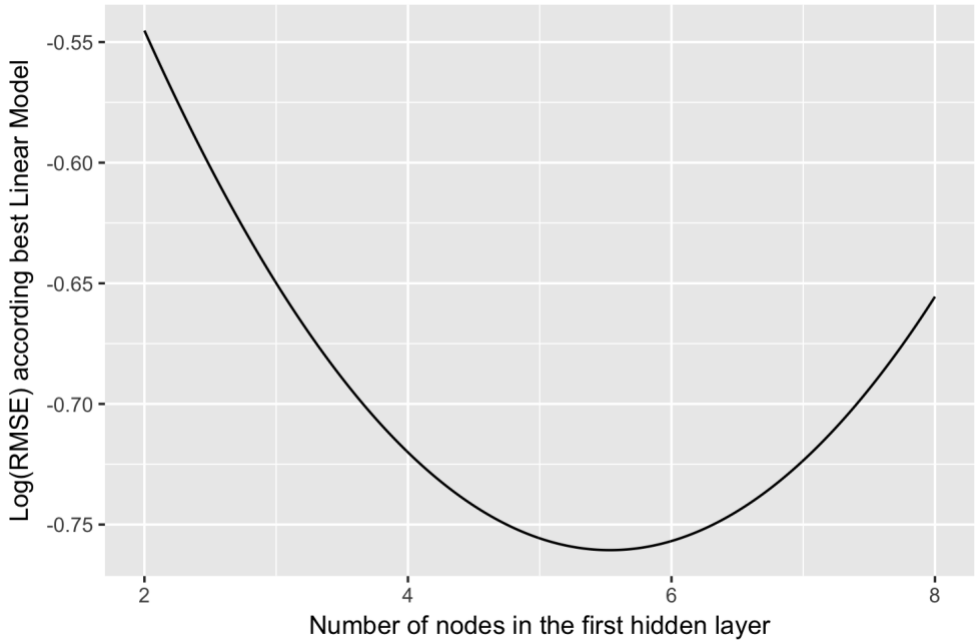
In this particular example, since we are using it to decide whether if choosing the architecture with DoE is comparable to k-fold cross-validation, we will find the best ANN according to k-fold cross-validation and the best according to the proposed DoE methodology and, afterwards, we will compare their performance in the validation set.

First of all, we will find the best architecture according to DoE, so we need to establish a range of values for several ML-hyperparameters. For the first hidden layer we have chosen between 2 and 8, so the central point will be 5, and for the second hidden layer the levels will be 0,5 and 10 nodes. Finally, for the learning rate the levels will be 0.001, 0.051 and 0.1. Moreover, we have decided to use 5 replicates for each experimental condition, since the optimization time for an ANN of 803 observations and 6 variables is minimal. After conducting the experiments, we transform the RMSE obtained for each EC by applying logarithms and, the best linear model according to BIC was the one that includes:

1. Second degree polynomial of the number of nodes in the first hidden layer
2. Second degree polynomial of the number of nodes in the second hidden layer
3. Second degree polynomial of the learning rate
4. Activation function
5. Interaction between 3 and 4.

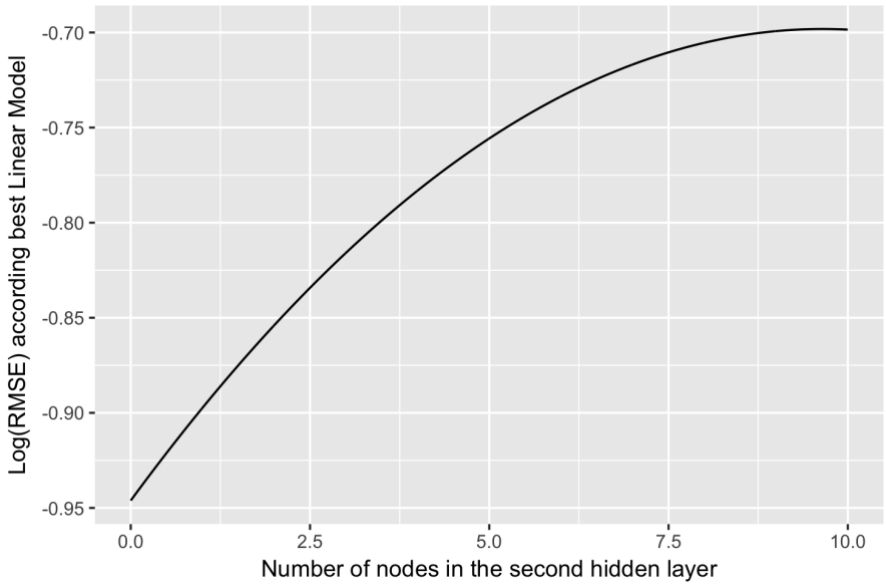
In order to find the best architecture, one could work with the equation offered by the linear model. Instead, we will use graphical representations because we believe that are more explanatory than extracting some derivatives.

Since the number of hidden nodes in each layer has come up to be independent to the rest of explanatory variables, we can decide the optimal value for them by fixing the values of the other predictors. In particular, for the number of nodes in the first hidden layer we have fixed the number of nodes of the second hidden layer to 5, the learning rate to 0.051 and the activation function to *tanh* and the parabola obtained is:



Annex Figure 2: Effect of the number of nodes in the first hidden layer according to the best linear model.

Therefore, our model is telling us that it seems to be an optimum in either 5 or 6 nodes in the first hidden layer and, since we want the most possible simplistic Neural Network, we decide that the best is at 5 nodes. Of course, we have decided this through this plot because the number of nodes in the first hidden layer does not relate to any other predictor in the equation of the best linear model and, as a consequence, by changing the value that we fixed for the other predictors this parabola would only move upwards or downwards. Doing the same for the nodes in the second hidden layer, we obtain the following:



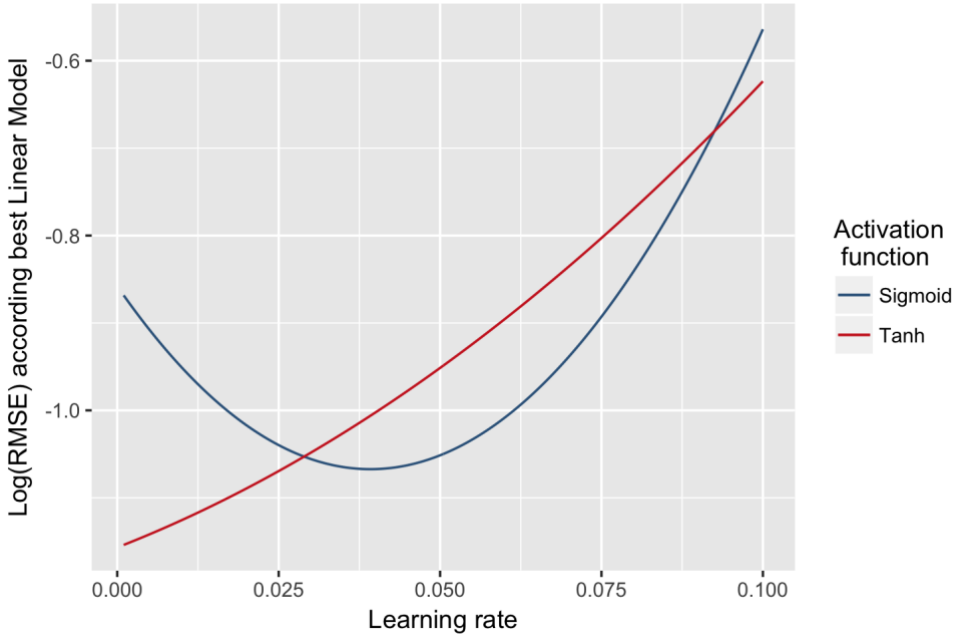
Annex Figure 3: Effect of the number of nodes in the second hidden layer according to the best linear model.

This graphical representation is showing that, according to our Model, the more nodes in the

second hidden layer, the worse predictive performance of the ANN and, as a consequence, that we need to set them at 0, meaning that our best ANN will only have two hidden layers.

Since the learning rate is interacting with the activation function, we will represent two curves for the learning rate, one associated to each activation function. Those curves have been obtained by fixing to 5 the number of nodes in the first hidden layer and 0 for the second hidden layer and they are shown in Annex Figure 4.

According to Annex Figure 4, even though it could be interesting to find the minimum with *sigmoid* (*logistic*) activation function, our linear model is telling us that if the learning rate is very small and *tanh* activation function is used, then the obtained ANN will yield a better predictive performance.



Annex Figure 4: Effect of the learning rate for each activation function according to the best linear model.

After having decided the best architecture according to this first phase of DoE, one could explore a new region of the ML-hyperparameters in order to find a better architecture. However, in this case getting away from the tested range of ML-hyperparameters does not seem to diminish the predictive performance and, therefore, we will not conduct a second phase with a second Box-Behnken design.

The next step will be obtaining the best ANN according to k-fold cross-validation and, to do that, we have used 5 folds, to obtain a similar fitting the ANN associated to each EC. However, of course, with cross-validation the number of EC (Experimental Conditions) are higher and, as a consequence, the time to find a suitable architecture will be higher. In particular, we are going to test the values for the number of nodes in the first hidden layer of 2,4,7 and 9, for

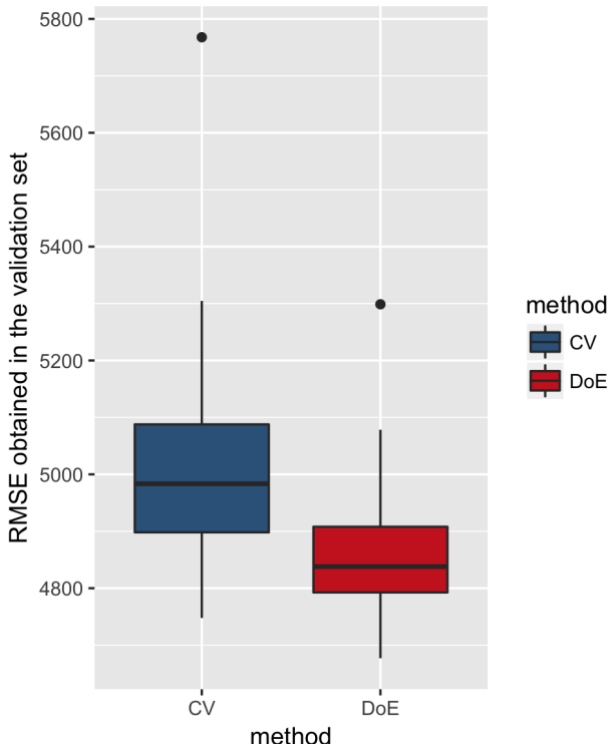
the second hidden layer, 0,4,8 and, for the learning rate 0.001,0.01,0.05 and 0.1. According to this 5-fold cross-validation, the best four architectures are the following ones:

Nodes hidden layer 1	Nodes hidden layer 2	Learning rate	Activation function	RMSE average
7	0	0.01	*tanh*	0.456
4	0	0.01	*tanh*	0.457
9	0	0.01	*tanh*	0.466
9	8	0.01	*tanh*	0.468

Annex Table 3: Best four architectures according to 5-fold cross-validation.

Even though they are similar to the one selected by DoE, the parameter associated to the regularization (i.e. learning rate) is ten times higher than the best one selected by DoE.

Now that we have obtained the best ANN architecture according to DoE and cross-validation, we will use the validation set to compare which one of the two architectures is better. However, we know that there is variability of the error obtained by each architecture (two ANN with same architecture yield different prediction) and, therefore, we will take a sample of 50 ANNs of each architecture and compute the RMSE on the validation set and, afterwards, we will use statistical science to establish if there are significant differences. A boxplot with the associated RMSE for each architecture (i.e. for the 50 RMSE obtained for each architecture) is the following:

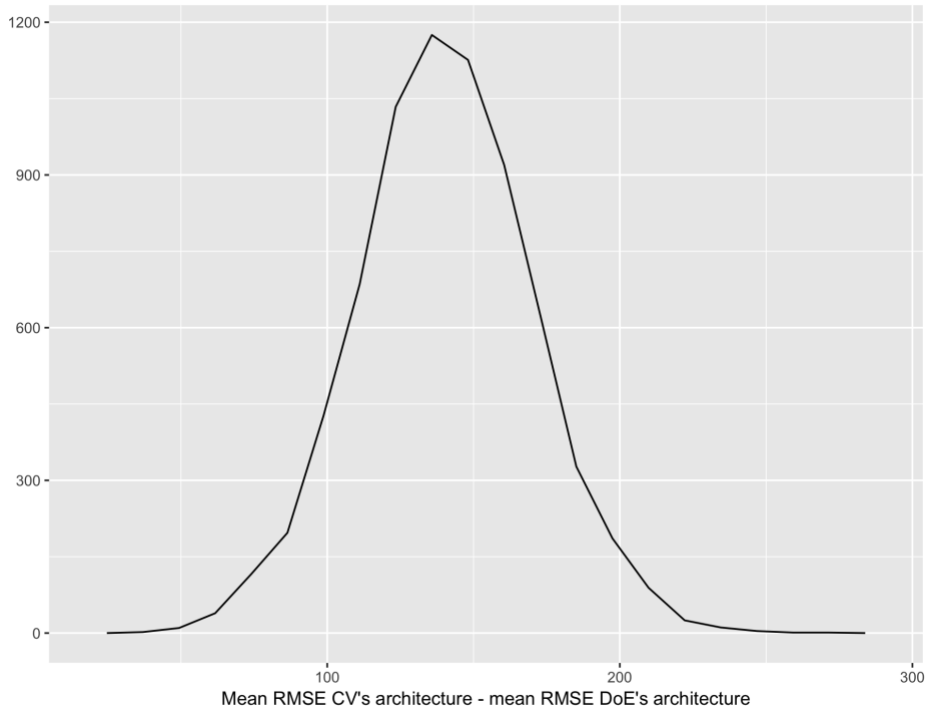


Annex Figure 5: Boxplot of the RMSE on the validation set for the ANNs fitted with the best architecture according to DoE and the best architecture according to 5-fold cross-validation.

Only by observing this, one could conclude that the architecture chosen by DoE seems to be performing better than the one obtained by cross-validation but, obviously, some rigorous statistical test must be conducted. In particular, we are going to fit a Bayesian Model in which we suppose that this data follows two Normal distributions, one per each architecture and, first, we are going to compare if the dispersion associated to each Normal is similar.

The 95% probability interval of the ratio between dispersion in the architecture chosen by cross-validation and the dispersion in the architecture found by DoE is the following is [1.121;1.996], meaning that the two normal distributions have different dispersion. In fact, if we conduct a Fisher's test to compare variances we obtain a p-value of 0.006, which is the same that our Bayesian model is telling us.

The posterior distribution of the differences between the two location parameters is the following:



Annex Figure 6: Posterior Distribution of the difference between the location parameter for the normal distribution fitted with the RMSE on the validation set obtained with the best architecture according to 5-fold cross-validation and the same location parameter for the case of DoE.

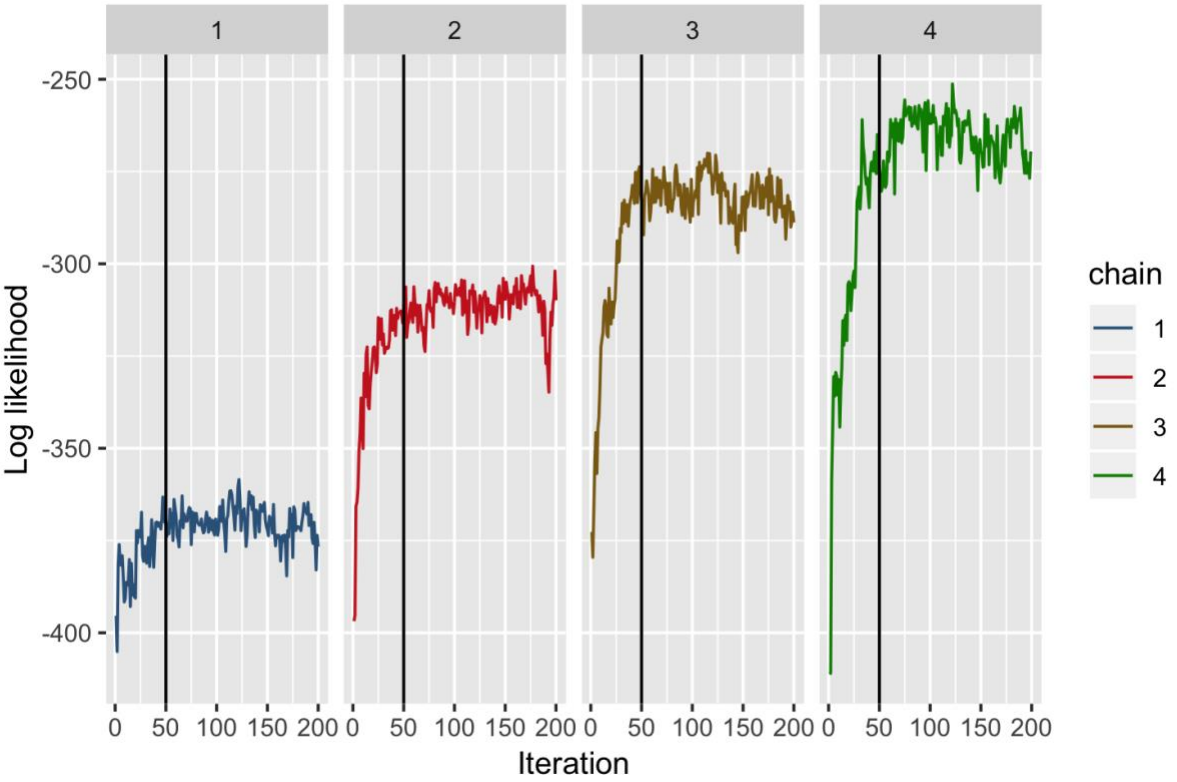
Moreover, the 95% probability interval for the true difference of means is [81.338;198.96] so it is clear, since it does not include 0, that the error obtained by the architecture chosen by cross-validation is greater than the error obtained by the DoE's architecture. In fact, if we conduct a t-test to assess difference of means with unequal variability, the result is a p-value of almost 0 and an associated confidence interval of [82.479;198.85] which is, obviously, similar to the one obtained through the Bayesian model. Therefore, our conclusion is that, for

this case, just with one efficient phase in the DoE methodology, we have found a superior architecture than with cross-validation.

Fitting the BNNs with both architectures and comparing predictive performance

Even though the DoE architecture is better than the one found with cross-validation, we are going to fit two BNN with each architecture, just to reveal a relevant result that we have obtained, which is that for BNNs, the choice of the architecture is not as relevant as in ANNs. These BNNs are fitted with four MCMC chains, meaning that, maybe, we are not able to approximate very well the posterior distribution since we are using just four local optima. In both cases we used 1,000 simulations and took only one simulation every five, because in BNNs the correlation of the MCMC simulation is very high and, therefore, it can be that we do not honestly sample from the posterior distribution.

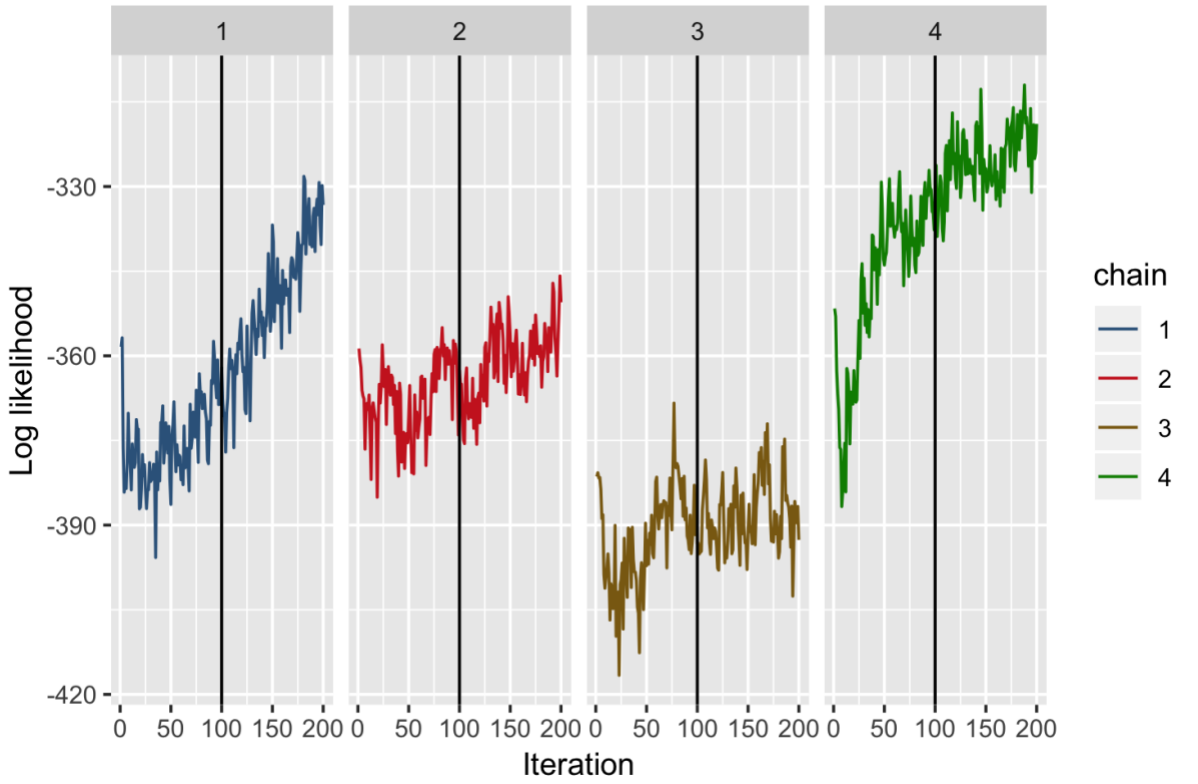
In the case of the BNN based on the DoE architecture, the chain of the log-likelihood is the following:



Annex Figure 7: Log-likelihood chains for the BNN with the best architecture according to DoE

According to these chains, the log-likelihood seems to become steady after the 50th simulation and, since we took one every five, this would mean after the 250th original simulation. If we use these simulations considered to come from the jointly posterior distribution, the RMSE on the validation set is of 4,648.19, using the mean of the predictive posterior as the punctual

prediction. For the BNN based on the cross-validation architecture the log-likelihood chains are:



Annex Figure 8: Log-likelihood chains for the BNN with the best architecture according to 5-fold cross-validation

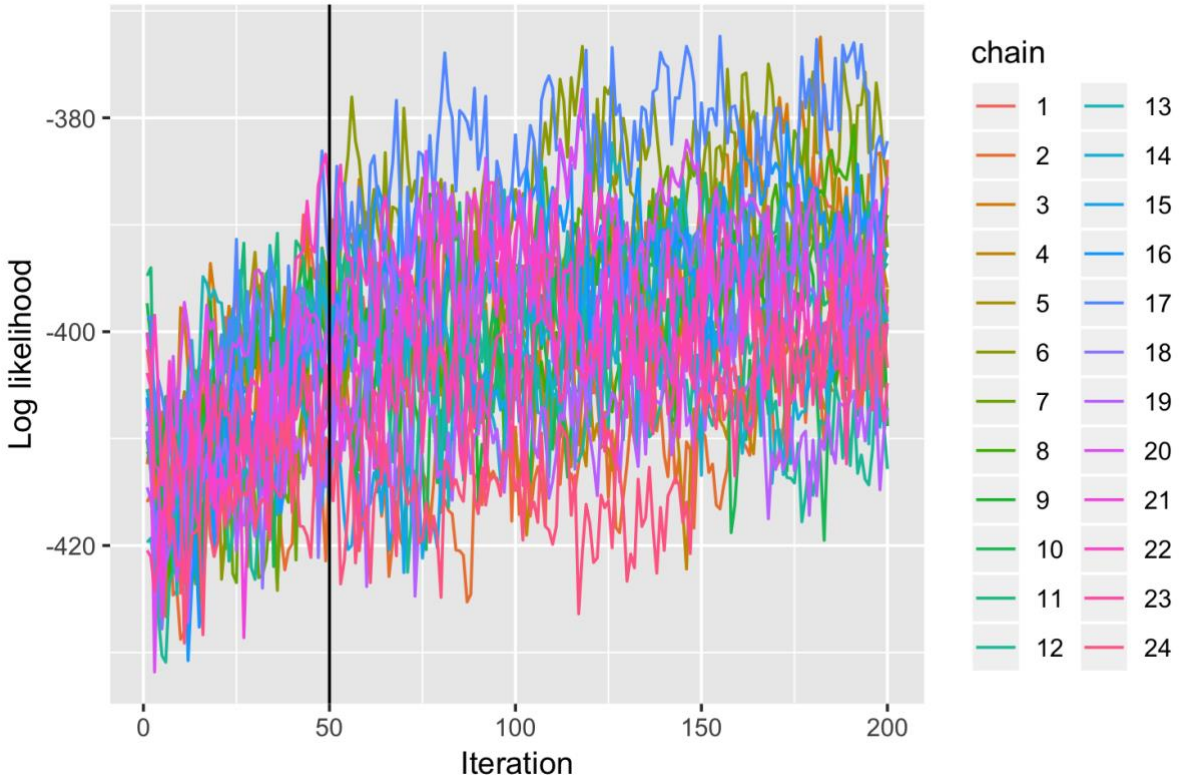
Like in the previous plot, the vertical black bar represents at which iteration of the MCMC we have decided that the simulations come from the posterior distribution. However, it is interesting to state that the first chain does not seem to have entered any steady state and, maybe, more simulations should be carried out. However, instead of allowing more simulations we will discard this first chain and work only with the simulations obtained from the three other chains, which yield a RMSE on the validation set of 4,651.8.

After obtaining these results, it is relevant to state that the error obtained by the BNN is considerably smaller than the one obtained by the ANNs and, moreover, that the two BNN yielded a similar predictive performance which indicates that, maybe, the architecture is not as relevant in BNN as it is in ANN.

According to the train data and the BIC criterion, the best linear model, considering up to third degree polynomials and two-way interactions between all variables, yields an RMSE on the validation set of 5,125.668, which is clearly superior to those obtained by the ANNs and, moreover, to those obtained by the BNNs. Therefore, in this example it is clear that the BNN seems a suitable model, since its ability to capture the behavior of the response variable is higher than the linear model because its predictions are more accurate.

Fitting the best BNN with more chains during MCMC

Before entering the validation of our BNNs, we are going to fit the BNN with the DoE architecture with different number of chains, in order to see if the more chains, the better the approximation of the posterior, which means that the results will be more reasonable. In particular, we are going to fit the BNN with 24 and 100 chains. With many chains, the graphical representation of the log-likelihood chains becomes too crowded and is usually difficult to assess convergence. Just as an example, we present the log-likelihood for the case of 24 chains.



Annex Figure 9: Log-likelihood chains for the BNN with the best architecture according to 5-fold cross-validation

In these cases with lots of chains, in order to assess the simulation in which we believe that the joint posterior distribution starts (vertical black line), we have analyzed each chain per separated because, obviously, in the previous graph it is complicated to extract it. In particular, it has been established that after the 50th simulation the following log-likelihood simulations become steady, as it happened with the case of only 4 chains.

The RMSE obtained with 24 and 100 chains in the validation set are 4,643.53, and 4,639.99, respectively. Even though these values are smaller than the case with only chains (4,648.19), we believe that the difference between them is not sufficiently big to believe that the more chains, the better punctual prediction. However, the reason why we started several chains

was to fully capture the posterior distribution and, therefore, enhance our interval-based predictions, rather than our punctual prediction.

After comparing all the BNNs and the linear model in the validation set, now the test set should be used as an indicator of the average error of the model. Nevertheless, this will not be done in this example because our goal is just to summarize the results of the experiments that helped us devise the proposed methodology.

Comparing uncertainty treatment in BNN against Linear Model and bootstrapped ANN

In order to see how the uncertainty is treated in each BNN (i.e. BNNs with 4, 24 and 100 chains with the DoE architecture), we are going to compute the 95% probability interval of the response variable for every individual of the training set in each BNN and compare them with the 95% probability interval of the best linear model obtained before and, moreover, with the 95% confidence interval obtained through bootstrap.

To obtain bootstrap confidence intervals for the response variable we are going to resample our training dataset (b) and, for each resample, we are going to estimate the parameters of our model (this estimate will be θ_b). For all the weights and biases of the ANN, the only thing required is to fit the ANN using the observations contained in b (this estimation of weights will be called W_b) and, for the deviation, we are going to use the standard deviation of the errors when predicting the resample b with W_b, which yields us σ_b. Afterwards, we will simulate from a normal distribution for all individuals of the training set (included or not in b) in which the location parameter $(\mu_{i,b})$ will be the value predicted through W_b and, the deviation parameter, will be σ_b. Therefore, for each resample b we are obtaining a plausible value of the response variable for each individual. Once we have finished all the bootstrap resamples, we will have an estimate of the distribution of the response variable for all individuals and, by taking the percentiles 0.025 and 0.975 of this distribution we will approximate the 95% confidence interval for the response variable of each individual of the training set.

It is relevant to state that if, for each resample we only calculated the predicted value for each training resample (instead of estimating σ_b and taking a random draw from a Normal distribution) then we would be delivering a confidence interval for the predicted value (i.e. for the μ_i in our model) so it would not capture the desired percentage of the response variable. In particular, this confidence interval for μ_i, only in the 66% of the training observations the observed value is captured by the interval because, as explained, this interval is for the

location parameter (predicted value) and not for the response variable, which is what is relevant for us and what we are capturing with the intervals of the BNNs and the linear model.

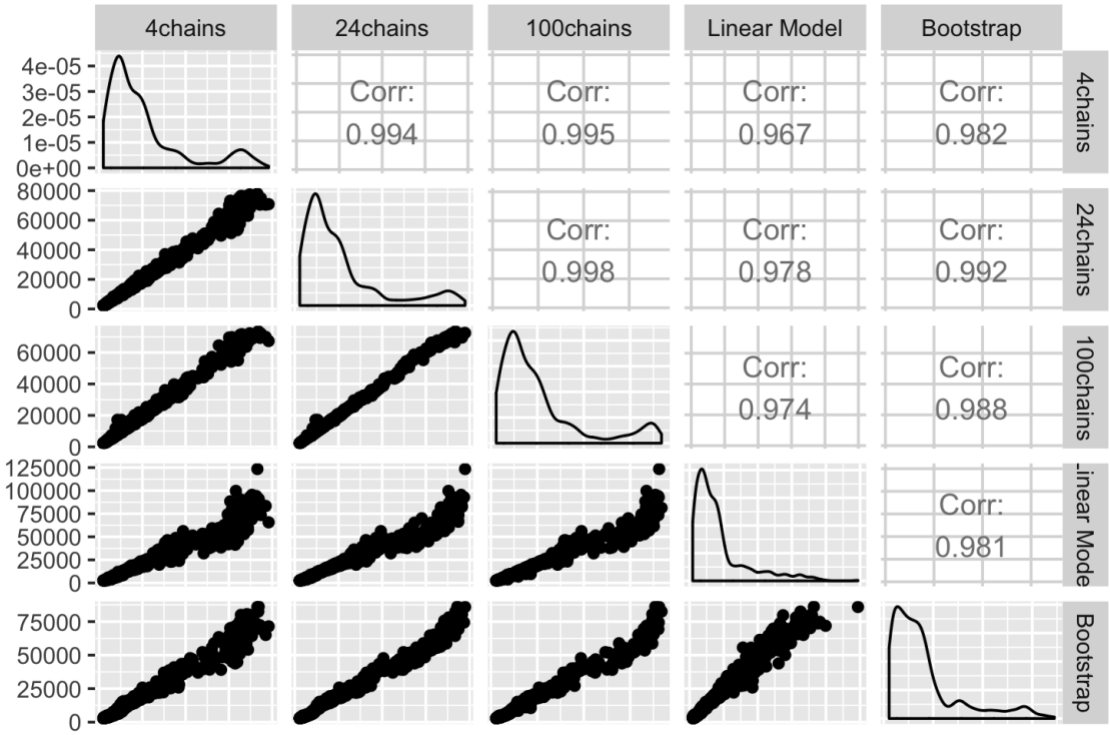
In order to see whether if our BNNs are delivering similar uncertainty than the other methods, we have calculated the average width of the intervals explained in the previous paragraphs. The results are the following:

Method	Average width 95% Interval on the response variable	% train captured in the interval	% validation captured in the interval
BNN 4 chains	20,526.88	94.89	93.66
BNN 24 chains	20,760.09	94.65	94.02
BNN 100 chains	20,663.3	94.65	94.02
Linear Model	21,173.68	94.4	94.78
Bootstrap	20,305.66	94.02	92.91

Annex Table 4: Uncertainty treatment according to each model.

Observing these results, one can conclude that our BNN is actually yielding the same global amount of uncertainty as ANN (obtained through bootstrap) which means that the interval-based prediction offered by the BNN is reasonable. However, the Bayesian framework constitutes a more general and grounded approach than using bootstrap to offer intervals for the ANN and, as a consequence, it is why we encourage the reader to use BNN to assess uncertainty in Neural Networks models. In particular, while bootstrap is an approximate technique to offer almost $1-\alpha$ confidence intervals, in BNN, thanks to all that has been explained in this thesis, the intervals are not an approximation but exact ones since we fully use our posterior distribution. In fact, as a direct consequence of this, we can observe in the previous table that the BNNs are able to capture more observation in its intervals than the ones offered by bootstrap. However, those percentages should not be taken as very relevant, since there are only 268 in the validation set and, as a consequence, the difference between the number of observations captured by the Bootstrapped ANN and the BNN with 100 chains is just three.

Even though these measures the global uncertainty of our BNN, at a given probability percentage, there is yet another check that must be conducted. In particular, we are going to test if those observations that the BNN is more uncertain (i.e. the interval is wider) are also those that are more uncertain for the other methods. Before we compared the global amount of uncertainty and, in this case, we analyze how this uncertainty is distributed.



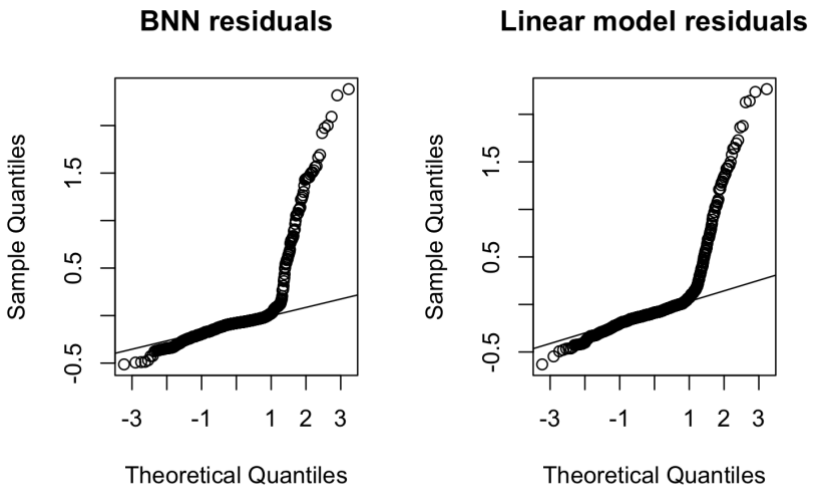
Annex Figure 11: Scatterplots of the 95% Interval width for each pair of models. In the diagonal there is a representation of the density of the model at the column.

After observing these plots, it becomes clear that all the models are distributing the uncertainty in a similar way, meaning that our BNN is, again, yielding reasonable results. It is relevant to state how the uncertainty given by the BNNs are very similar between them (correlation higher than 0.994), which could mean that maybe it is not that relevant to launch several chains because all the modes seem to be similar.

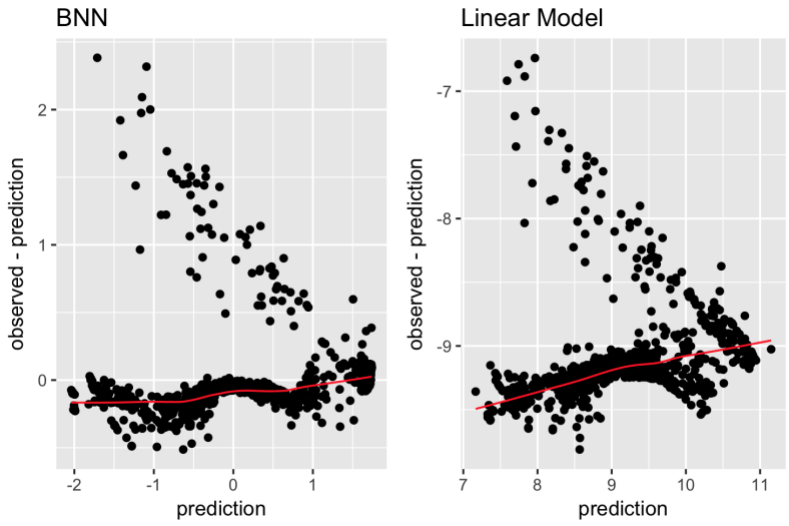
As a conclusion of this analysis of how uncertainty is treated in BNNs, we want to state that their results are grounded and similar to the linear model, which is the basic model used to conduct interval-based predictions. Moreover, since the BNN is able to yield better punctual predictions than the linear model, this would mean that the BNN is able to better generalize the behavior of the response variable and, as a consequence, that the probability intervals given by the BNN could be more realistic than the ones offered by the linear model, meaning that our BNN is a highly competitive model when compared to the linear model. When comparing ANN with BNN, the argumentation is similar, and we recommend using BNN because it not only yields better punctual predictions but, moreover, offers interval-based predictions that seem to be more suitable because they are based in an almost exact method rather than in approximations methods as bootstrap, which highly depend on the training set size.

Validation of the basic model hypotheses

Apart from all these results, some validation of the normality and homoscedasticity premises should be conducted. For the normality we present in Annex Figure 11 a QQ normal plot for the residuals of both the BNN with 100 chains and the best linear model and, for the homoscedasticity we present a scatterplot of the residuals vs the predicted value (for the BNN the response variable is standardized while it is not for the linear model).



Annex Figure 12: QQplot of the BNN with 100 chains residuals (left) and the Linear model residuals (right) to assess the suitability of a Gaussian distribution for the response variable.



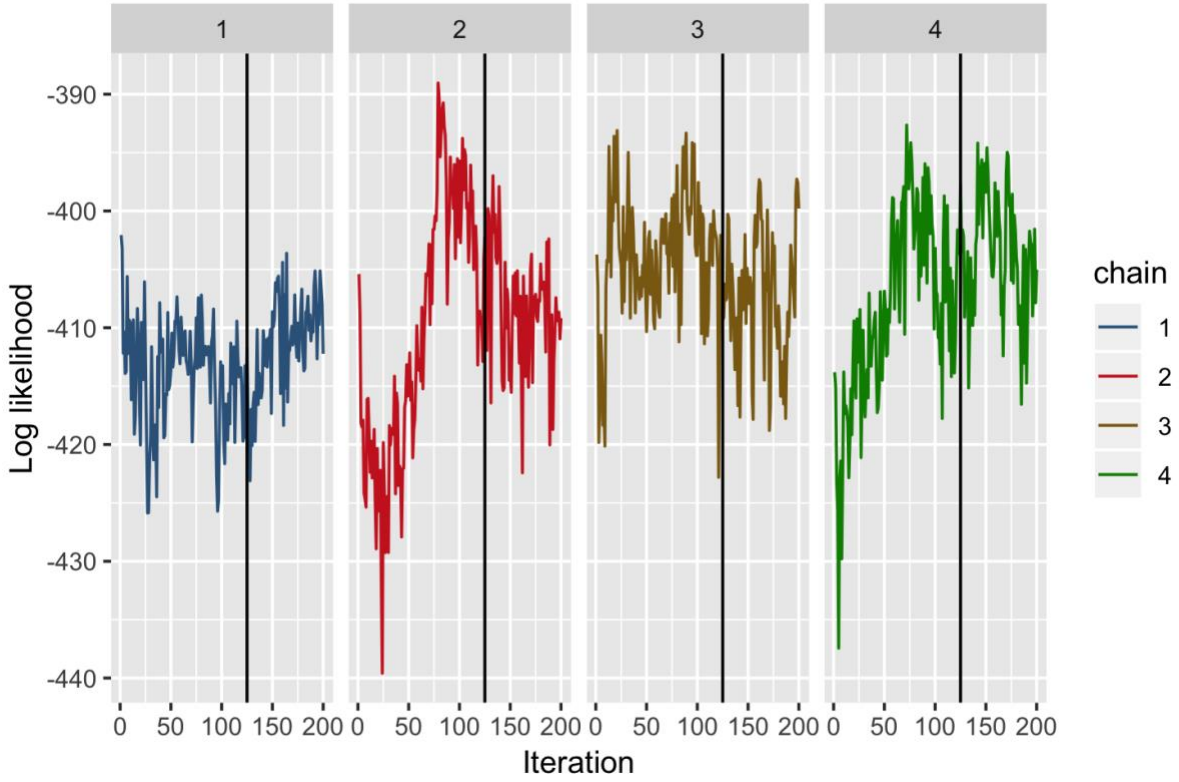
Annex Figure 13: Scatterplot of the residuals and the predicted value (scaled for the case of the BNN) to check the validity of the homoscedasticity hypothesis. The BNN is the one with 100 chains.

Neither the linear model nor the BNN are validated through the analysis of their residuals and, the main reason, is because there is a group of observations in which the real applied charges to the client are higher than the predicted ones. The conclusion about this is that we are offered only a limited set of predictors to predict the health insurance charges and there are

relevant variables, like the health history of each client, that are driving the final applied charges and that are not collected in our model.

Automatic Relevance Determination

In this part of this example we are going to use the BNN based on the DoE-selected architecture to check the implementation of Automatic Relevance Determination (ARD) and its general results. In order to fit this new BNN, we will use only four chains and the same number of simulations than before, in order to allow comparison with the already obtained BNN. The log-likelihood chain plot is the following and, according to it, we have decided to take the last 75 simulations:

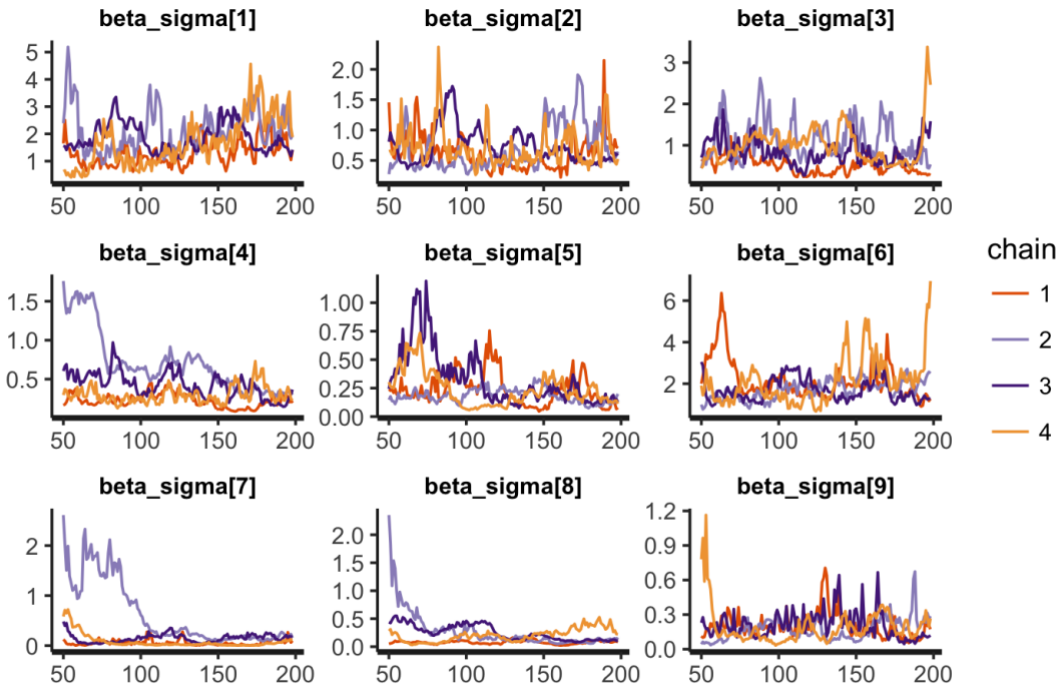


Annex Figure 14: Log-likelihood chains of the BNN with Automatic Relevance Determination and the best architecture according to DoE

The RMSE on the validation set is of 4,658.85, which is almost the same than the one obtained without ARD (4,648.19). In order to assess the relevance of each input, we need to secure that the chains of the dispersion parameters associated to each input node have converged. Those chains are plotted in Annex Figure 14:

Through Multidimensional Convergence using the log-likelihood we have assessed that after the 125th simulation start the jointly posterior distribution samples and, according to these

chains, all parameters associated to the dispersion of the weights have converged after that simulation. Therefore, we can use them to conduct an analysis of each input's relevance.



Annex Figure 15: Chains associated to each of the parameters that captures the relevance of each input node

If we take the mean of the samples from the posterior distribution of each of those parameters, for the numeric predictors we obtain:

Variable (input node)	Children (4)	Age (2)	BMI (3)
Value	0.33	0.7	0.84

Annex Table 5: Expected value from the posterior distribution of each relevance parameter (i.e. parameter beta_sigma associated to each input node) for the numeric predictors.

Therefore, the most important variable to predict the medical insurance charges are the BMI and the age of the client, while the number of children seems to be less relevant for that purpose. For the categorical variables, we obtain the following:

Category (Input node)	northwest (7)	southeast (8)	male (5)	southwest (9)	yes (6)
Variable	region	region	sex	region	smoker
Value	0.10	0.15	0.18	0.2	2.04

Annex Table 6: Expected value from the posterior distribution of each relevance parameter (i.e. parameter beta_sigma associated to each input node) for the categorical predictors.

According to this, the main feature that is driving our prediction is whether if the client smokes or not, while the other categories does not seem to drive the prediction that much.

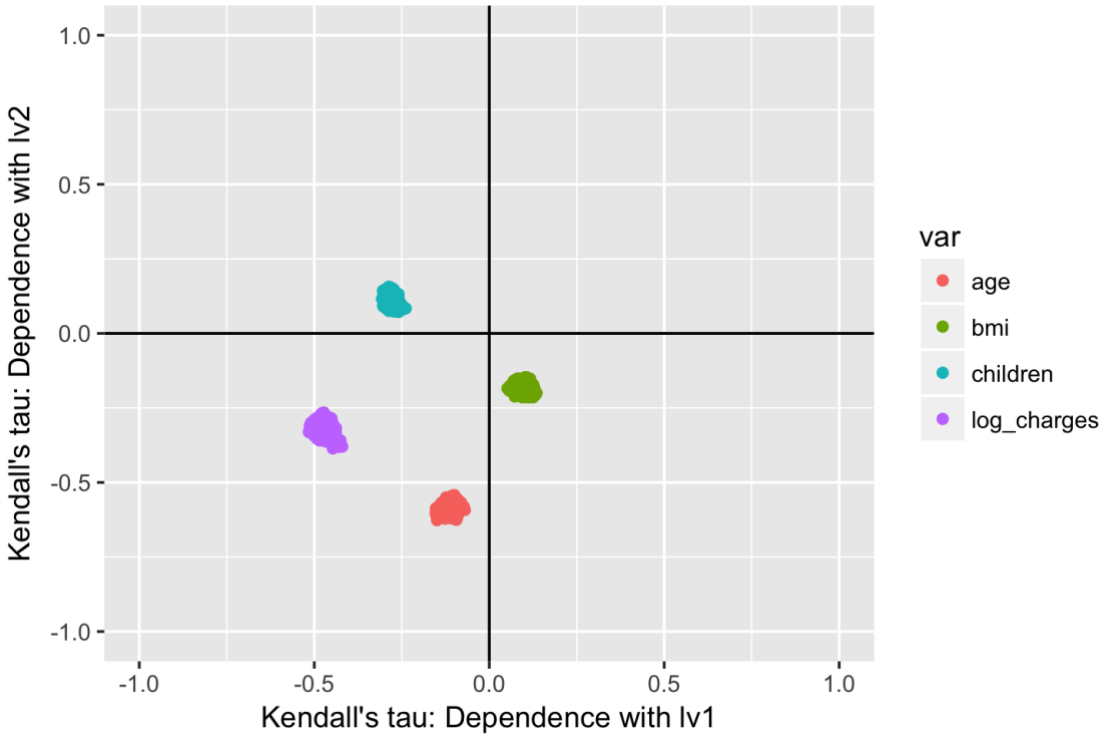
In other words, what is causing our predicted value for the medical insurance charges to be high or small is, mainly, if the client smokes or not and they Age and BMI. The other variables may have some impact, but they are not the most relevant features.

Interpretation layer

In this final part of this example we are going to review the main results of the interpretation layer that has been proposed in this thesis. In all the fitted BNNs, each chain has a total amount of 150 samples from the posterior distribution and, since in the interpretation layer only one chain can be used (because all of them refer to different local optima) we decided to fit a BNN with a unique longer chain with 600 simulations from the posterior distribution that will allow us to explain the interpretation layer. The plot of the associated log-likelihood chain is in Annex Figure 15:

Annex Figure 16: Log-likelihood chain of the BNN with the best architecture according to DoE but only with long chain.

First of all, we are going to show a scatterplot in which we represent the Kendall's τ statistic for each numeric predictor with each latent variable. Our goal is to assess which explanatory variables are not relevant in our model to predict the applied health insurance charges, because we have a circle of probability associated to those τ according to the posterior distribution. In fact, this could be used as a way to erase innocuous variables that should not be included.

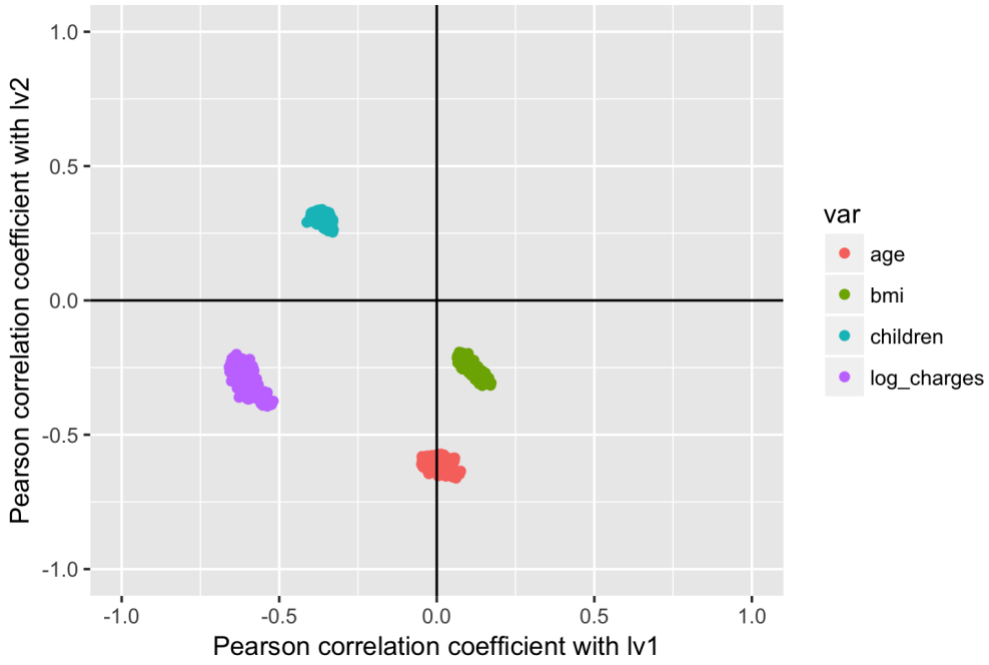


Annex Figure 17: Representation of the Kendall's τ between each numeric predictor and each simulation of the latent variables of the BNN.

Log_charges is the response variable used and it is showed just to summarize which linear combination of lv_1 and lv_2 are the ones used for prediction. Since any continuous variable crosses any axis, this means that wide probability intervals of τ do not include 0 and, as a consequence, that all the predictors are related with the latent variables, which means that all of them are useful for prediction in our BNN and should not be discarded. In fact, all explanatory variables enter in the equation of the best linear model so, again, our BNN is telling us a similar conclusion than the linear model, that all explanatory variables are useful for prediction.

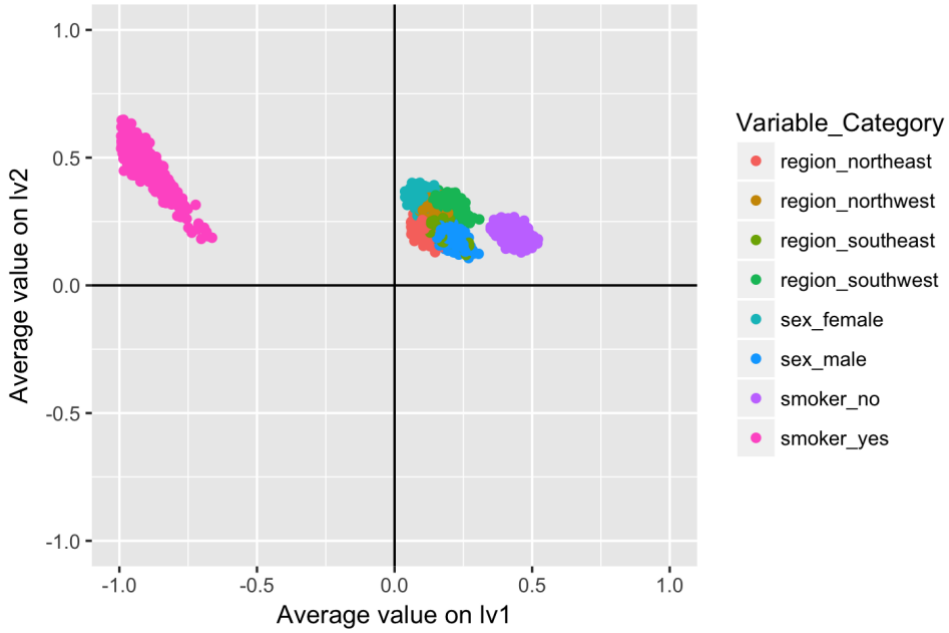
Instead of computing Kendall's τ at each simulation, we can calculate the Pearson's correlation coefficient in order to summarize the particular relationship of the predictor with the latent variables, even though, of course, that relationship is likely to be nonlinear and the Pearson's correlation coefficient only captures linear relationships. This plot is included in Annex Figure 17.

The plotted values are random samples from the posterior distribution of the Pearson's correlation coefficient of each numeric predictor with each latent variable. They tell us that the first latent variable is not linearly related with the age, since the posterior distribution simulations of the Pearson's correlation coefficient cross the y-axis of the plot.



Annex Figure 18: Pearson's correlation coefficient of each numeric predictor with each latent variable at each simulation of the latent variables.

In order to relate the categorical predictors with the latent variable, for each categorical variable we group the individuals according to its category. Afterwards, we compute the average value of the latent variables in each category in each simulation and we plot them. The associated plot is in Annex Figure 18.



Annex Figure 19: Average values of the latent variables that the individuals of a particular category take at each simulation of the latent variables.

The most important one categorical variable is whether the client smokes or not because they are the categories that take higher absolute values in the first latent variable and, for the case of *smoker_yes* also in the second latent variable. Another relevant piece of information about

the model is that it does not seem that living in a different region clearly affects the response variable, since the cloud of points associated to each region is in the same part of the scatterplot (in fact, the idea that region is not important was also captured by ARD). On the other hand, and in this graphical representation is difficult to see it, the two clouds of each gender are close, but they are not touching, meaning that our model is interpreting that there is some kind of distinction between the charges applied to women with respect to the ones applied to men.

The most relevant conclusion of these plots is that those variables that are more important for prediction are the ones that are further away from the axis origin and they are, mainly, if the client smokes and the age. The main difference with ARD is that in this plot it does not seem that BMI, even though is relevant, it is not one of the most important. Finally, the other relevant conclusion is for categorical variables and it is based on the fact that if the cloud of points associated to each category have some intersection, it means that those categories affect in a similar way the response variable, which is the case for the region but not for the sex of the client.

As a conclusion of this interpretation layer we want to explain that its goal is to enlighten what is driving the BNN prediction in order to reduce its functioning as a black box. Apart from that, since BNN automatically captures interactions between variables and nonlinearities, we believe that analyzing the effect of the predictors on the response variable is more sensible than with other methods. For instance, we believe that it is more reasonable to erase an explanatory variable from our model if it does not have a significant (i.e. different than zero) Kendall's τ with any latent variable than how they are removed in a linear model, because in a linear model we may not be considering nonlinear relationship with the predictor and relationship with other explanatory variables. Finally, the last strong feature of this representation is that it is based on areas of probability, allowing us to take more grounded decisions because we know how to define if two categories are "close" or not to each other (and, therefore, different) in the graphical representation.

Implementing BNN as a pricing model for Airbnb in Barcelona

Summary about the reasons why we discarded each predictor of the dataset

1) Variables that were discarded because they are used to conduct the scraping procedure or to structure the Airbnb's information, so they are not meaningful for the user:

scrape_id, last_scraped, listing_url, id, host_id, thumbnail_url, medium_url, picture_url, xl_picture_url, host_url, calendar_last_scraped.

2) Variables that are not meaningful for the user:

Host_name: The user will not expect any effect of his/her name on the price of its listing and, therefore, it will not be used as a predictor.

Host_location: Similar than before, the host is renting the apartment and he/she will not expect this to be meaningful to assess the price of his/her product.

Host_has_profile_pic: Same reason than the previous two predictors.

Host_acceptance_rate: In advance, the user does not know the percentage of guests that he/she will be prone to accept, so it is discarded as a predictor.

Property_type: There are many levels and they are not well-defined, so including it would be confusing for the user and would reduce the usefulness of the pricing model.

Guests_included: It captures the number of guests currently hosted on 7th February. If the user is using the pricing model to find the price of its apartment is obvious that he will not have any guests in the apartment yet, so this variable will not be included.

Calendar_updated: Apart from the fact that this variable has been used to filter non-relevant suppliers, it is not useful as a predictor of the pricing model, because asking the user "how often will you update the calendar?" could be a misleading and confusing question.

Availability: Just as it happened with *guests_included*, the user does not know how many clients his/her apartment will have because he/she is using the pricing model to publish it for the first time.

License: It is just a code of the apartment's license; the user will not expect this to have any impact on the price for his/her apartment.

3) Variables that require text mining in order to be relevant for our BNN, so they are discarded:

name, summary, space, description, experiences offered, neighborhood_overview, notes, access, interaction, house_rules, host_about, first_review, last_review.

4) Variables that capture a concept that is better encapsulated in other predictors:

Host_listings_count: It includes all the other listings that the same host has currently published. Instead of using this predictor, we will use **host_total_listings_count**, since it includes all the host apartments, including the specific observation.

Host_verifications and **host_has_profile_pic:** Those variables are more complicated than **host_identity_verified** which captures a similar concept than them in a more straighforward way and, as a consequence, they will not be used.

Street, **Neighborhood**, **Neighborhood cleansed, zipcode** and **smart_location:** They all refer to the location of the apartment, but they have more levels and are more confusing than **Neighborhood_group_cleansed**, which includes the districts of Barcelona. Therefore, in other to both facilitate the use of this tool by the user and avoid having a large model matrix (i.e. input layer), we decided to use the summarized variable **Neighborhood_group_cleansed**.

City, state, market, country_code, country, requires_license, jurisdiction_name: All those variables would be useful if more than one city was compared, but since we are only using Barcelona and its surroundings they are discarded.

Transformation and missing values treatment for the selected predictors

Transit:

Description: Text with information about the public transport connections with the apartment.

Transformation: We converted this variable into a Boolean that states whether if the apartment has some information about the transit or not. The main reason is that those that have some information are the ones that have good connections because the user spent some time explaining it, while the others do not.

Missing values: According to the previous transformation, those apartments with missing values in this variable constitute a whole category which relates to “badly connected apartments”.

Square_feet:

Description: Square feet of the whole apartment or the room that is being rented.

Missing values: This value presents a 96% of missing values and, moreover, there is some misunderstanding about it because some users employ the square feet of all the apartment while others only the square feet of the actual room that is being rented. Therefore, we have decided to dismiss this variable as a predictor.

Is_business_travel_ready:

Description: This variable collects whether if the apartment is willing to host Airbnb users that stay at Barcelona from business rather than holidays.

Missing values: There are not missing values.

This service is rather new in Airbnb and maybe the users are not fully aware of the consequences of implementing it. Therefore, we have decided to keep this variable out of the set of predictors.

Host_since:

Description: Date of the first listing that the host published.

Transformation: We converted it to the variables *days_as_host* in which we aim to capture the experience of the host publishing apartments and, therefore, the know how about Airbnb. For the user, it would be easy because if he/she is new this would be 0, while for users that are already hosts it would be automatically captured from the Airbnb database.

Missing values: It has a total of 48 missing values. After analyzing these 48 observations we observed that they have also missing values in some other relevant variables and, as a consequence we decided to discard those observations from the dataset. Therefore, our dataset after treating this variable has a total of 14,172 listings and we will continue our analysis with these observations.

Host_response_time:

Description: Average time that the host awaits before responding a message. It is a categorical variable with levels *N/A*, *a few days or more, within a day, within a few hours, within an hour*.

Transformation: Since there are too many levels, we decided to convert into *N/A* for users that have never been hosts, *Less than a day* and *more than day*. This variable would be automatically filled by the Airbnb dataset, so the user would not need to respond any question referring to this. It aims to estimate the activity of the host on Airbnb.

Missing values: The missing values (1,539) are captured under the category *N/A* which refers to the fact that the host does not have an average response time because he/she has never received a message before.

Host_response_rate:

Description: Percentage of messages that the hosts answers.

Missing values: There are 1,539 missing values, which are the same observations than in the variable *host_response_time*. Therefore, to avoid multicollinearity and high dependency between some input nodes of the BNN, we will discard this variable and it will not be included.

Host_is_superhost:

Description: It is a categorical variable that indicates if the host is a distinguished member of Airbnb because of his/her experience and average qualification from the guests. Our goal is to use is it as a variable to assess the market placement of the apartment.

Missing values: There are no missing values.

Host_total_listings_count:

Description: Variable that counts all the apartments that the host has published. With the variable *host_since*, this aims to capture the experience of the host in Airbnb.

Transformation: Since the relevant is the experience that the host has in the market of Barcelona, we have converted this variable into one that only counts the number of listings published by the host in Barcelona, i.e. in the dataset.

Missing values: There are no missing values.

Host_identity_verified:

Description: Categorical variable with two levels that indicates wheter if the host has been verified by Airbnb or not.

Missing values: There are no missing values.

Neighborhood_group_cleansed:

Description: Categorical variable with nine levels that captures the district at which the apartment is located.

Missing values: There are no missing values.

Is_location_exact:

Description: Categorical variable with two levels that indicates if the location published in Airbnb is exact or just an approximation. This variable reduces the uncertainty of the customer.

Missing values: There are no missing values.

Room_type:

Description: Categorical variables with three levels that shows the type of the listing. The categories are: *Entire home/apt, Private room* and *Shared room*. Those three types of listings are three almost substitutive markets, especially the first two types.

Missing values: There are no missing values.

Accommodates:

Description: Number of individuals that can live concurrently in the listing. The maximum value is 16 and it indicates 16 or more.

Missing values: There are no missing values.

Bed_type:

Description: Categorical with five levels that indicates the kind of beds that the listing offers. The levels are *Airbed, Couch, Futon, Pull-out Sofa* and *Real bed*.

Transformation: Since 14,036 of 14,172 observations are from the category *Real bed* we decided to transform this variable into a categorical named *has_real_bed* that indicates whether if the apartment has Real beds or not, aiming to capture the comfortability of the listing.

Beds:

Description: Number of beds available for the guests.

Transformation: We converted this variable into *Beds_per_accommodate* in order to have a comparable index of the comfortability between the listings, since those listings with more accommodates are obviously the ones with more beds.

Missing values: There are 8 missing values. However, for those observations that the number of beds is missing, it is stated that, for the variable *bed_type* they have real beds. Therefore, it is concluded that there is some kind of error and these observations are discarded. Moreover, there are 64 listings that state that they have 0 beds. However, if we observe the variable *bed_type* for them, all the 64 are in the category *real_bed*, meaning that there is, also, some kind of error in those listings because it is not possible that they do not have beds nor any other type of furniture for sleep (like couch). Therefore, as we did before, we decided to remove those observations to avoid including strange listings when computing the market price. As a consequence, after dealing with this variable our dataset has a total of 14,100 listings.

Bathrooms:

Description: Number of bathrooms available for the guests.

Transformation: Like we did with *beds*, we created the variable *bathrooms_per_accommodate.*

Missing values: There are 18 missing values and we considered that the value is missing because the apartment does not have any bathroom and the host did not want to publish it

to purposely misinform potential customers and increase the chances of renting the listing. Therefore, we converted those 18 missing values into 0 bathrooms, because there are other apartments that have stated that they have 0 bathrooms, so it is possible.

Bedrooms:

Description: Number of the bedrooms available for the guests. There can be 0 bedrooms meaning that the guest must sleep, for instance, in the living room.

Transformation: Since this variable is related also to the comfortability of the apartment, we transformed into *bedrooms_per_accommodate* in order to build a comparable index.

Missing values: There are 5 missing values and our rationale has been the same that for *bathrooms*, so we converted those missing values into 0.

Amenities:

Description: Text variable that contains all the amenities (i.e. complement services such as Internet, TV, dishwasher…) of the apartment.

Transformation: We parsed the variable and created a categorical variable for each amenity, stating whether if the listing has that amenity or not. We discarded those amenities that only less than 1% of the observations had it. All the taken amenities are:

24_hour_check_in, Accessible_height_bed, Accessible_height_toilet, Air_conditioning, Baby_bath, Babysitter_recommendations, Bathtub, Beach_essentials, Bed_linens, Breakfast, Buzzer_wireless_intercom, Carbon_monoxide_detector, Cleaning_before_checkout, Coffee_maker, Cooking_basics, Crib, Dishes_and_silverware, Dishwasher, Doorman, Dryer, Elevator, Essentials, Ethernet_connection, Extra_pillows_and_blankets, Family_kid_friendly, Fire_extinguisher, First_aid_kit, Flat_path_to_front_door, Free_parking_on_premises, Garden_or_backyard, Hair_dryer, Handheld_shower_head, Hangers, Heating, High_chair, Host_greets_you, Hot_tub, Hot_water, Indoor_fireplace, Internet, Iron, Kitchen, Laptop_friendly_workspace, Lock_on_bedroom_door, Long_term_stays_allowed, Luggage_dropoff_allowed, Microwave, Oven, Patio_or_balcony, Pets_allowed, Pets_live_on_this_property, Pocket_wifi, Pool, Private_entrance, Private_living_room, Refrigerator, Room_darkening_shades, Safety_card, Self_Check_In, Shampoo, Single_level_home, Smoke_detector, Smoking_allowed, Step_free_access, Stove, Suitable_for_events, TV, Washer, Waterfront, Well_lit_path_to_entrance,

Wheelchair_accessible, Wide_clearance_to_bed, Wide_clearance_to_shower, Wide_doorway, Wide_entryway, Wide_hallway_clearance, Window_guards, toilet

Missing values: There are 31 observations with missing value, meaning that the apartment does not have any amenity (or the host did not spend any time on detailing all the amenities). Therefore, for those 31 observations the value at each categorical variable associated to each amenity will be the category that states that they do not have the amenity.

Weekly_price:

Description: The price that the host has published for one-week stays.

Transformation: Even though this is related to the price, which is the response variable, we will transform it in order to become a predictor, because we will not work with more than one response variable. In particular, we calculated the percentage of discount per night by comparing the standard price with the weekly price and, with this discount we created a categorical variable called *weekly_discount* with four levels: *Negative, None, Up until 25%, More than 25%.* For the user that wants to use the pricing this would be easy, since he should only choose one of the four options above.

Missing values: There are 12,608 observations that have missing value in this variable, meaning that they are not offering a different price for long stays. With our transformation, all those observations have been collected under the category *None.*

Monthly_price:

Description: The price that the host has published for one-month stays.

Transformation: Like we did with *weekly_price*, but now the categories of the created variable *monthly_discount* are: *Negative, None, up until 50%, more than 50%.*

Missing values: Like in *weekly_price*, but now the number of observations with missing value is 12,404.

Security_deposit:

Description: Dollars that the guests must pay as security deposit before entering the apartment and that, in case that nothing is broken or spoiled, will be returned in its totality.

Transformation: This variable is not related to the final price, because usually they are completely returned. Since there is a lot of variability in this security deposit and several missing values, we decided to transform this variable into one more relevant for the user and, also, easier to work with when building the BNN or any other model. In particular, we created a categorical variable called with four levels: *None, Up until 200$, Up until 400$* and *more than 400$*.

Missing values: There are 4,468 observations with missing value that we supposed that are missing values because the host did not bother to fill this information since he did not want to include any security deposit. Therefore, all these observations were included in the *None* category.

Cleaning_fee:

Description: Dollars that the guests must pay besides the price per night. This a fixed quantity that must be paid without considering the number of nights that the apartment has been rented and that is not returned at the end of the stay. Therefore, since it an extra charge, it will be included in the response variable.

Transformation: As explained, we will add the cleaning fee to the price per night of the apartment, so we will compare the prices of staying one night. Moreover, we will create a categorical predictor call *has_cleaning_fee* which will indicate whether if the user wants to charge extra with a cleaning fee or not.

Missing values: There are 3,353 missing values. The rationale is the same than with the security deposit and, therefore, we considered that those apartments do not include any extra charge for cleaning purposes.

Extra_people:

Description: Extra quantity that must be paid per night if the guests is accompanied by any other person not stated when reserving the apartment. For instance, if the guests made a reservation for two people and one night three people sleep in that apartment, that third person will need to pay the charges stated by this variable.
Transformation: We have transformed it into a categorical variable that states whether if there is an extra charge for inviting over more people or not.

Missing values: There are no missing values.

Minimum_nights:

Description: Number of minimum nights that the listing must be rented. This does not influence the price since the stated price is dollars per night, not per minimum stay.

Missing values: There are no missing values.

Maximum_nights:

Description: Number of maximum nights allowed to rent the listing.

Transformation: We fixed a maximum at 365 days because several hosts used values like 2,147,483,647 days because the Airbnb platform did not have an option of “no limit”.

Missing values: There are no missing values.

Number_of_reviews:

Description: Total number of reviews from guests that the listing has received from its publication.

Missing values: There are no missing values.

Review_scores:

Description: When a guest publishes a review about the listing he/she is asked to write down a small comment and, moreover, to specify a grade for all these aspects: Overall satisfaction, accuracy of the information, check-in service, cleanliness of the apartment, communication with transport, location and the value for money. Each listing has a score on each of these aspects and, therefore, there are a total of seven variables to collect this information.

Transformation: Due to the fact that people uses integer values and usually gives high qualifications, we decided to categorize these variables into three groups: *None, Not_excellent* (if the qualification is lower than 95 for the overall score or 9 for the others) and *Excellent.* The category *None* will be used for those users that are publishing the listing for the first time, while the others can be used for users that are employing the pricing model to change the published price of their apartment.

Missing values: All these variables have between 2,200 and 2,300 observations with missing values and all of those with missing value have been grouped under the category *None.*

Instant_bookable:

Description: Categorical variable that states whether if the host accepts automatically the guests or if he/she has to confirm that the guest is allowed to rent the listing.

Missing values: There are no missing values.

Reviews_per_month:

Description: Average number of reviews received per month.

Transformation: We have converted this variable into *months_on_market* using the variable *number_of_reviews*.

Missing values: There are 2,148 with missing value in this variable because are those that do not have any review.

Cancellation_policy:

Description: Categorical variable with five modalities that indicate the requirements that need to be fulfilled in order to cancel the reservation and be able to recover the payment.

Transformation: We decided to convert this variable into a categorical variable with only three modalities, in which the apartments with strict cancellation policies are grouped in one category, while before they were separated in three and there were few observations for some of those categories.

Missing values: There are no missing values.

Require_guests_profile_picture:

Description: Categorical variable that states whether if the host requires the guests to have profile picture in order to rent the apartment.

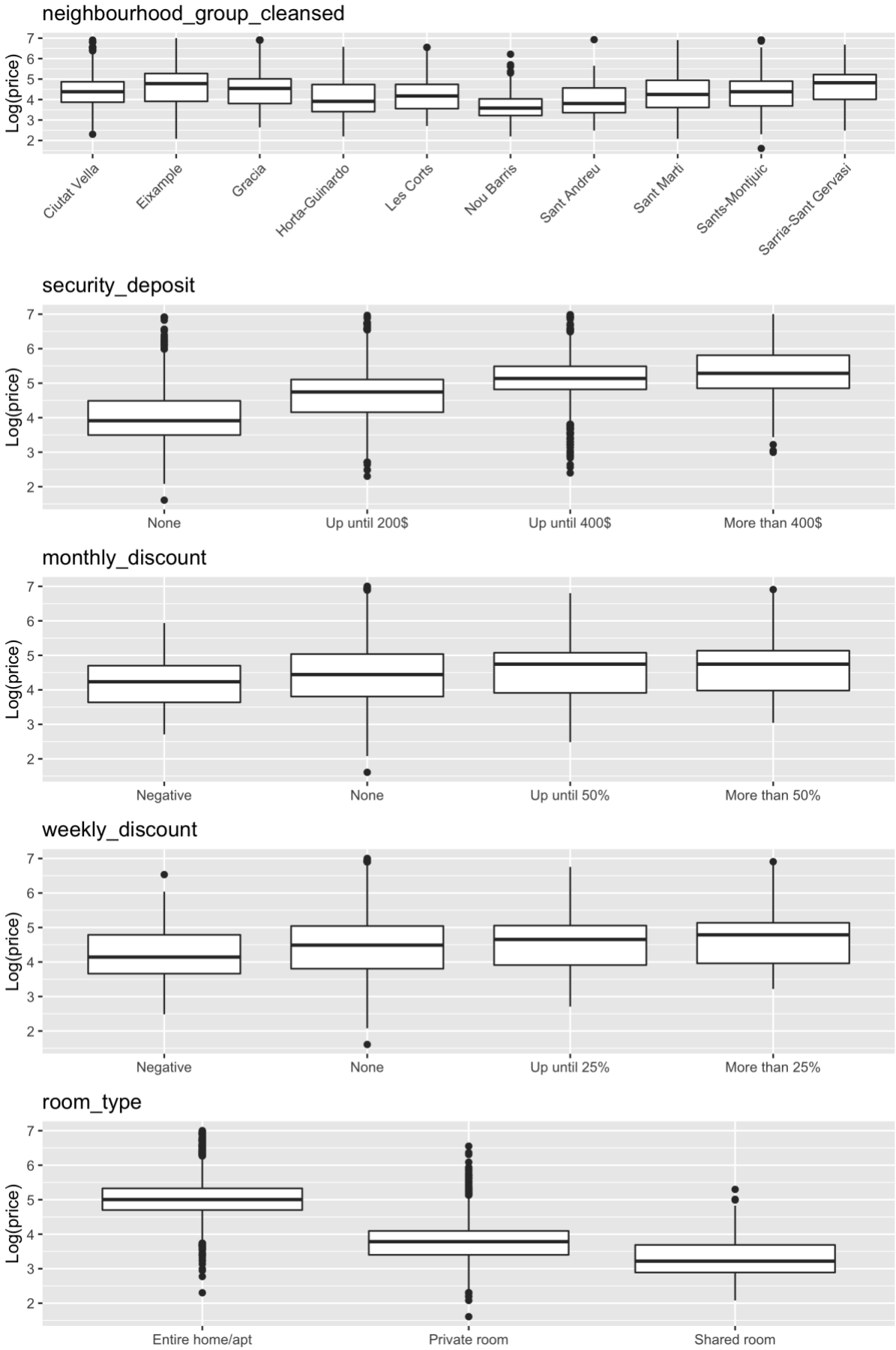
Missing values: There are no missing values

Require_guests_phone_verification:

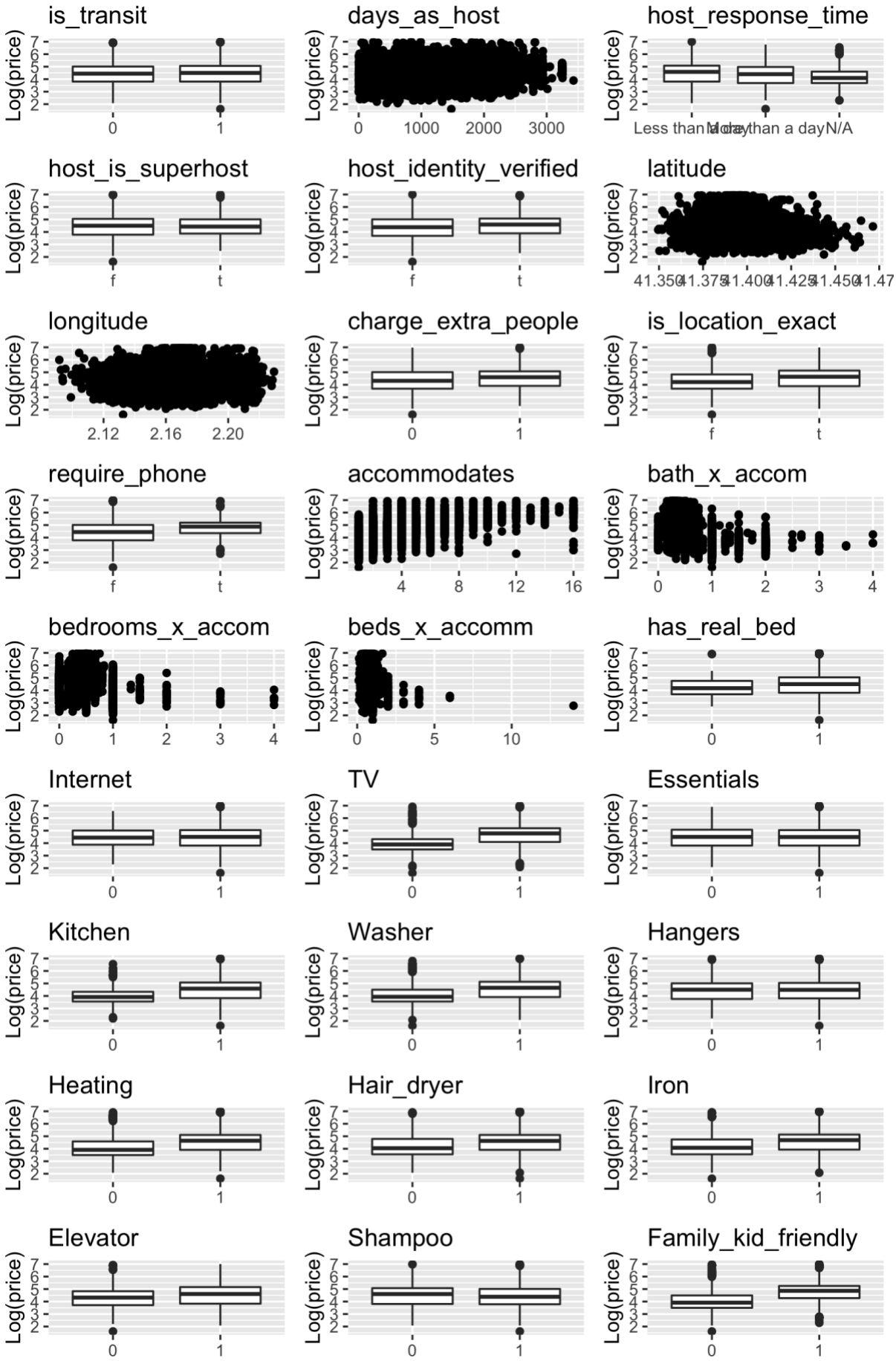
Description: Categorical variable that states whether if the host requires the guests to have a validated phone number in order to rent the apartment.

Missing values: There are no missing values.

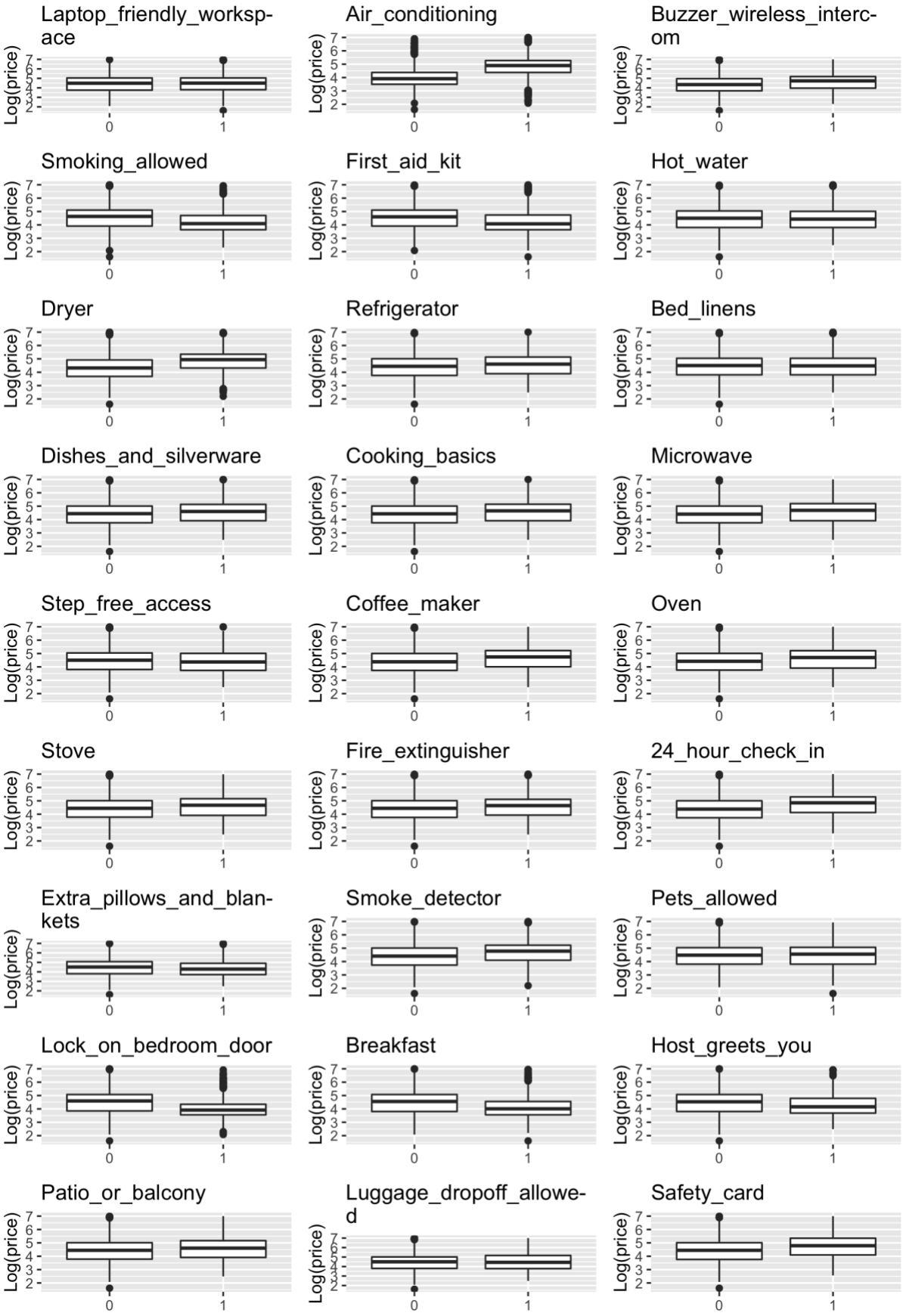
Bivariate descriptive analysis of each predictor with the logarithm of the listing price



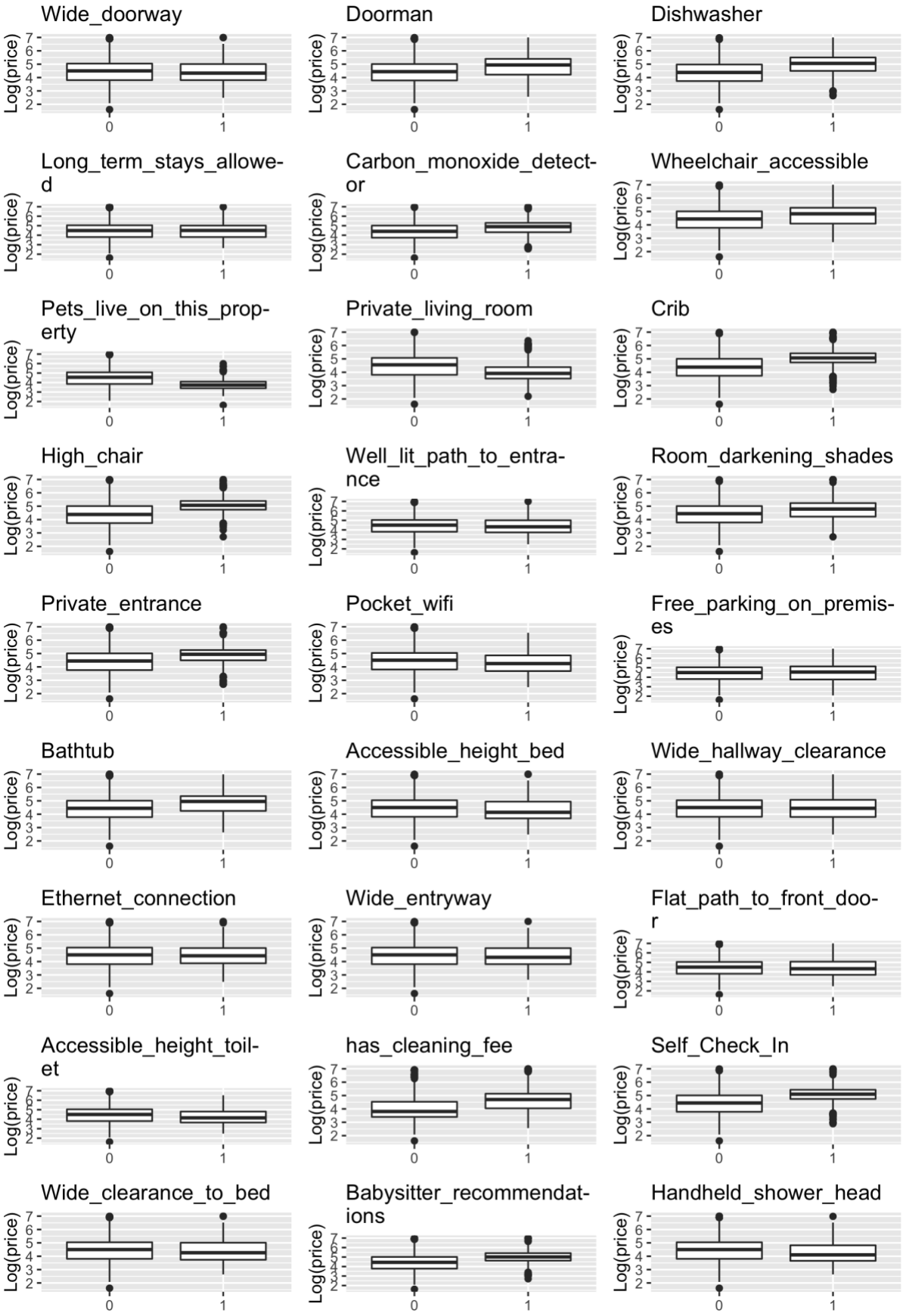
Annex Figure 20: Bivariate plots between the first 5 predictors and the response variable (i.e. the logarithm of the price with the cleaning fee).



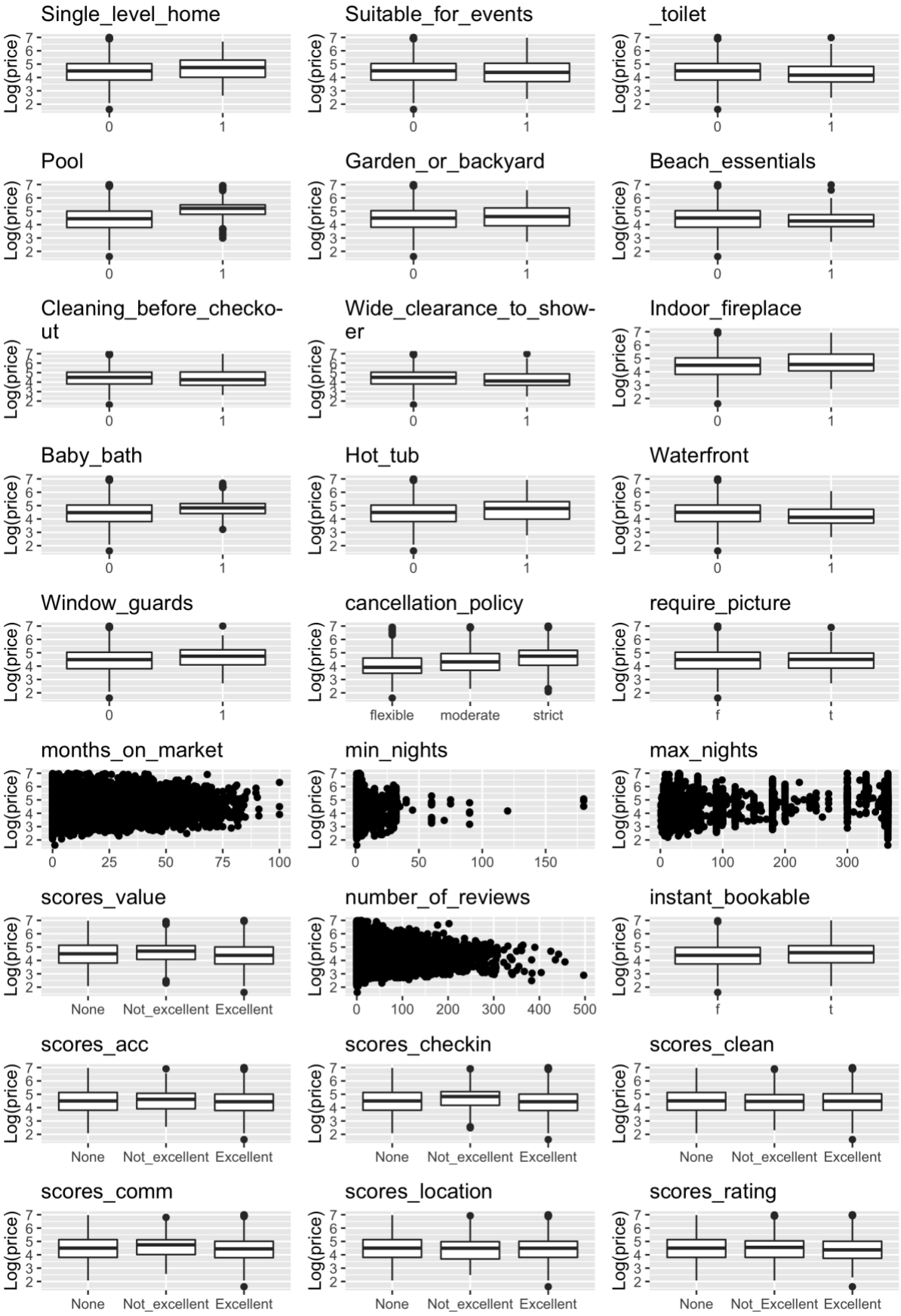
Annex Figure 21: Bivariate plots between 27 predictors and the response variable (i.e. the logarithm of the price with the cleaning fee).



Annex Figure 22: Bivariate plots between 27 predictors and the response variable (i.e. the logarithm of the price with the cleaning fee).



Annex Figure 23: Bivariate plots between 27 predictors and the response variable (i.e. the logarithm of the price with the cleaning fee).



Annex Figure 24: Bivariate plots between 27 predictors and the response variable (i.e. the logarithm of the price with the cleaning fee).

Included terms in the best linear model

The terms included in the best linear model are:

1. Room_type
2. Third degree polynomial of the number of accommodates
3. Has_cleaning_fee
4. Security_deposit
5. Scores_rating
6. Charge_extra_people
7. Amenity_Air_Conditioning
8. Host_response_time
9. Amenity_diswasher
10. Third degree polynomial of the number of days that the host has been host
11. Linear effect of the number of reviews
12. Host_is_superhost
13. Third degree polynomial of the index bath_per_accommodate
14. Amenity_Dryer
15. Third degree polynomial of the listing' latitude
16. Weekly_discount
17. Amenity_pool
18. Linear effect of the listing' longitude.
19. Instant_bookable
20. Amenity_Long_term_stays_allowed
21. Amenity_Shampoo
22. Cancellation_policy
23. Is_location_Exact
24. Linear effect of the number of minimum nights
25. Amenity_Indoor_fireplace
26. Amenity_Kitchen
27. Third degree polynomial of the number of listings that the host has in Barcelona
28. Third degree polynomial of the number of months that the listing has been published
29. Interactions: 1 and 13, 10 and 27, 6 and 27, 1 and 2, 15 and 18, 3 and 7, 11 and 28, 22 and 27, 2 and 6, 1 and 6, 17 and 26, 17 and 18, 7 and 24, 3 and 24, 19 and 23, 6 and 9, 3 and 10, 6 and 22, 3 and 27, 3 and 12, 6 and 19, 14 and 20, 3 and 6, 10 and 17.

Code required to fit a BNN with R and STAN

In this last section of the annex we provide a basic code structure in order to fit a BNN according to the methodology proposed and obtain predictions from it. The first thing will be separating the dataset into train, validation and test:

```
inds<-1:nrow(dd)
train<-sample(size=round(0.60*nrow(dd)),inds)
validation<-sample(size=round(0.2*nrow(dd)),inds[-train])
test<-inds[c(-train,-validation)]
dd_train<- dd[train,]
```

Afterwards, we need to standardize the train dataset, including the logarithm of the response variable *y*:

```
dd_train$log_y<-log(dd_train$y)
dd_train <- dd_train[,-which(names(dd_train)%in%"log_y")]
num<-which(sapply(dd_train,class)!="factor")
cat<-which(sapply(dd_train,class)=="factor")
dd_train_stand<-cbind(scale(dd_train[num]),dd_train[cat])
```

Once this has been done, the next step is creating the model matrix and collecting the mean and standard deviation of the logarithm of the response variable in order to rescale the predictions:

```
mat.mod<-model.matrix(log_y~.,dd_train_stand)
p<-ncol(mat.mod)
y_mean_train <- mean(dd_train$log_y)
y_sd_train <- sd(dd_train$log_y)
```

Then, we create a function able to fit an ANN with the package *mxnet* with two hidden layers and *tanh* activation function. Its arguments will be the data on which the ANN will be fitted and the number of nodes in each hidden layer and the learning rate.

```
fit_ANN <- function(train.x,train.y,num_hidden,l.rate,...){
  array.layout="rowmajor"
  data <- mx.symbol.Variable("data")
  label <- mx.symbol.Variable("label")
  fc1 <- mx.symbol.FullyConnected(data, num_hidden=num_hidden[1],
name="fc1")
  tanh1 <- mx.symbol.Activation(fc1, act_type="tanh", name="tanh1")
  fc2 <- mx.symbol.FullyConnected(tanh1, num_hidden=num_hidden[2],
name="fc2")
  tanh2 <- mx.symbol.Activation(fc2, act_type="tanh", name="tanh2")
  fc3 <- mx.symbol.FullyConnected(tanh2, num_hidden=1, name="fc3")
  lro3 <- mx.symbol.LinearRegressionOutput(data=fc3, label=label,
name="lro3")
  mxModel <- mx.model.FeedForward.create(lro3, X=train.x, y=train.y,
                ctx=mx.cpu(), #num.round=100,
                array.batch.size=128,
                eval.metric=mx.metric.rmse,
                verbose=FALSE,initializer = mx.init.normal(0.2),

array.layout=array.layout,learning.rate=l.rate,
                                          optimizer="rmsprop",...)

  return(mxModel)
}
```

The next step will be deciding the number of nodes per hidden layer and the learning rate. In this case we decided to fit ANNs with 10 nodes in the first hidden layer and 2 in the second one. Afterwards, we will fit 4 ANNs according to those ML-hyperparameters in order to obtain the initial values for the BNN:

```
best_num_hidden<-c(10,2)
best_lrate <- 0.1
for (i in 1:4){
  assign(paste0("mynnet",i),
      fit_ANN(train.x=mat.mod,
      train.y=dd_train_stand$log_y,
      num_hidden = best_num_hidden,
      l.rate=best_lrate,num.round=50))
   assign(paste0("sig_net",i),
      sd(predict(get(paste0("mynnet",i)),mat.mod[,-1],array.layout   =
      "rowmajor")-dd_train_stand$log_y))
}
```

Just as an extra note, in order to predict with one of those obtained ANNs and obtain the RMSE of that prediction, in the original scale, the code required would be:

```
RMSE(exp(predict(mynnet_DoE1,mat.mod[,-1]
      ,array.layout="rowmajor")*y_sd_train +
      y_mean_train),exp(dd_train$log_y))
```

As we were explaining, once we have obtained the ANNs, we need to collect the initial values:

```
Initials_4_chains<-list()
for (i in 1:4){
Initials_4_chains[[i]]<-list(beta =
      rbind(as.array(get(paste0("mynnet",i))$arg.params[[2]]),
      as.matrix(get(paste0("mynnet",i))$arg.params[[1]])),
      w_hid1=as.array(get(paste0("mynnet",i))$arg.params[[3]]),
      b_hid1=as.array(get(paste0("mynnet",i))$arg.params[[4]]),
      w_out=as.array(get(paste0("mynnet",i))$arg.params[[5]])[,1],
      bias_out=as.array(get(paste0("mynnet",i))$arg.params[[6]]),
      sigma=get(paste0("sig_net",i)))
}
```

Before starting the MCMC simulation we need to specify a file with extension *.stan* with the Bayesian model, i.e. the BNN. The content of this file is:

```
data{
      int<lower=0> N;
      int<lower=0> p; //number of columns of the model matrix
      int<lower=0> K_hid1; //number of hidden nodes of the first
hidden
      int<lower=0> K_hid2; //number of hidden nodes of the second
hidden layer
      vector[N] y;
      matrix[N,p] x;
}
parameters{
      matrix[p,K_hid1] beta;
      matrix[K_hid1,K_hid2] w_hid1;
      vector[K_hid2] b_hid1;
      vector[K_hid2] w_out;
      real bias_out;
      real<lower=0> sigma;
}
transformed parameters{
      matrix[N,K_hid1] hidden_1;
      matrix[N,K_hid2] hidden_2;
      for (n in 1:N){
            for (k in 1:K_hid1){
                  hidden_1[n,k] = tanh(x[n]*col(beta,k));
            }
            for (k in 1:K_hid2){
                  hidden_2[n,k] =
tanh(hidden_1[n]*col(w_hid1,k)+b_hid1[k]);
            }
      }
}
model{
      for (n in 1:N)
            y[n] ~ normal(bias_out + hidden_2[n]*w_out,sigma);
}
```

Before starting the MCMC simulation with Stan we still need to prepare the input information for this model:

```
N<-nrow(dd_train_stand)
y<-dd_train_stand$log_y
p<-ncol(mat.mod)
dat_BNN<-list(N=N,y=y,x=mat.mod,p=p,
           K_hid1=best_num_hidden[1],K_hid2=best_num_hidden[2])
```

Now we have all the elements required to fit the BNN with Stan. It will be done with the package to interact between R and Stan: *rstan*. Just as a note for the reader, we want to warn about the computation cost that launching the following lines of code can have.

```
fitted_BNN<- stan(file = 'BNN_model.stan', data = dat_BNN,
                      iter = 1000,warmup=250,
                      chains= 4,init=Initials_4_chains,
                      pars=c("hidden_1","hidden_2"),thin=5,
                      include=FALSE,cores=4,refresh=10)
```

All the simulations are stored in *fitted_BNN* and, the following code extracts those simulations after warmup in order to yield a prediction of some observations. In this case, we will use the same observations from the training set (i.e. the object *mat.mod*).

```
posterior <- as.matrix(fitted_BNN)
nsim<-nrow(posterior)
N<-nrow(mat.mod)
p<-ncol(mat.mod)
sim_used<-1:nsim
scaled_preds<-matrix(nrow=nrow(mat.mod),ncol=length(sim_used))
i<-1
for(j in sim_used){
  pre_hidden_1<-mat.mod%*%matrix(posterior[j,
           1:(p*best_num_hidden[1])],nrow=p,ncol=best_num_hidden[1])
           hidden_1 <- tanh(pre_hidden_1)
  pre_hidden_2 <- hidden_1%*%matrix(posterior[j,
           grep("w_hid1",colnames(posterior))],nrow=best_num_hidden[
           1],ncol=best_num_hidden[2])
  pre_hidden_2_bias <- pre_hidden_2 + matrix(rep(posterior[j,
      grep("b_hid1",colnames(posterior))],N),nrow=N,ncol=best_num_hid
      den[2],byrow=T)
  hidden_2 <- tanh(pre_hidden_2_bias)
  pre_mu <- hidden_2%*%matrix(posterior[j,
      grep("w_out",colnames(posterior))],
      nrow=best_num_hidden[2],ncol=1)
  mu <- pre_mu +posterior[j,"bias_out"]
  scaled_preds[,i]<-mu
  i<-i+1
}
```

This last piece of code stores, for each simulation of the MCMC method after warmup, the sample of the expected value for each individual in the matrix *scaled_preds*. In other words, this object *scaled_preds* contains the samples from $\Pi(\mu_i|y)$ in the standardized scale of the logarithm. Therefore, the only last thing required to do is to compute the expected value for each $\Pi(\mu_i|y)$, reescale it to the original scale, and use that as punctual prediction. We also compute the RMSE of that prediction in the training set:

```
Punctual_prediction_BNN<- exp(apply(scaled_preds,1,mean)*y_sd_train
                              + y_mean_train)
RMSE(Punctual_prediction_BNN,exp(dd_train$log_y))
```

The only last thing that would be left with this BNN, apart from extracting samples from the posterior predictive for each individual in order to offer the interval for the response variable, is to check the convergence of the log-likelihood chains. According to the normal distribution for the response variable, the likelihood depends on the expected value (i.e. prediction in *scaled_preds*) and the deviation (simulations are stored in *fitted_BNN*, in the parameter *sigma*). First of all we extract all the simulations after warmup of *sigma* and we save the number of real iterations after warmup (in our case: (1000 – 250)/5 = 150) per each chain and, also, the number of chains:

```
num_chain <- 4
num_it<- 150
sigma_BNN <- as.data.frame(fitted_BNN)$sigma
```

After this, we have two relevant objects: *scaled_preds* which is a matrix of dimentions N*(num_it*num_chain) and *sigma_BNN*, which is a vector of dimension num_it*num_chain. For each simulation associated to each chain, we will compute the log-likelihood:

```
chain_likelihood<-matrix(nrow=num_chain*num_it,ncol=3)
for (j in 1:num_chain){
  it<-1
  for (i in (1+150*(j-1)):(150+150*(j-1))){
    error_sum_it<-sum((scaled_preds[,i]-y)^2)
    chain_likelihood[i,1]<-j
    chain_likelihood[i,2]<-it
    it<- it + 1
    chain_likelihood[i,3]<-error_sum_it*(-1/(2*sigma_BNN[i]^2))-
N*log(sigma_BNN[i]) -(N/2)*log(2*pi)
  }
}
```

The object *chain_likelihood* contains the 150 values of the log-likelihood per each chain, so now it can be used to plot the log-likelihood chains and assess convergence. For the interval-based predictions the only thing required is to simulate 600 randoms draws of a Normal distribution (one for each simulation of *scaled_preds* and *sigma_BNN*) for each individual and taking the desired quantiles (2.5% and 97.5%) of that posterior predictive distribution.