TREBALL DE FI DE GRAU

Grau en Estadística

Títol: Rellevància de variables en models algorítmics predictius

Autor: Oriol Rovira Tauler

Director: Pedro Delicado

Departament: Estadística i Investigació Operativa

Convocatòria: Juny 2020

UNIVERSITAT DE BARCELONA

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

AGRAÏMENTS

Especialment al meu tutor, Pedro Delicado per la seva paciència, dedicació i per l'eficàcia i rapidesa en resoldre tots els meus dubtes. La seva gestió i planificació del treball ha estat molt clara i de gran ajuda.

En darrer lloc, a la meva família per tot el recolzament donat durant aquest període.

RESUM

Machine Learning és un conjunt de mètodes que permeten als ordinadors aprendre de les dades per fer i millorar prediccions. Aquest conjunt de mètodes és un nou enfoc modern i té una millor precisió en la predicció comparada amb els models estadístics tradicionals.

Aquest treball pretén, mitjançant mètodes de Machine Learning, desenvolupar una llibreria a partir de l'article de Delicado i Peña (2020). També, s'ha treballat a partir de dades reals d'habitatges de lloguer procedents de la pàgina web de la Inmobiliaria Idealista i s'han introduït nous models per estudiar dues rellevàncies descrites en l'article. Per últim, es comparen els resultats obtinguts amb la rellevància implementada de cada model i entre les rellevàncies descrites per l'article.

Paraules clau: Rellevància, Variables Fantasma, Permutacions aleatòries, Llibreria, Predicció

Classificació AMS: 03C65 *Models of other mathematical theories*, 62J12 *Generalized linear models*

ABSTRACT

Machine Learning is a set of scientific methods which allows computers to learn from data in order to improve all the predictions. This combination of scientific methods is a new modern approach and it has a better accuracy in terms of prediction when it comes to compare it with traditional statistical models. This university research is an attempt to develop a bookstore based on Delicado i Peña's article (2020), by making use of Machine Learning methods.

Furthermore, this research was accomplished by using rental housing real data from Inmobiliaria Idealista webpage. In addition, there were also inserted new models in order to study two important facts described in the article above. To sum up, the results obtained using implemented measures in each model and among relevances described in the article are compared.

Keywords: Relevance, Ghost Variables, Random Permutation, Library, Prediction.

AMS Classification: 03C65 *Models of other mathematical theories*, 62J12 *Generalized linear models*

ÍNDEX

LLISTAT DE FIGURES

INTRODUCCIÓ

El món de Data Science ha despertat un gran interès en tota la comunitat estadística. En aquest TFG s'intenta aprofundir en alguns aspectes de la ciència de dades que han anat evolucionant amb rapidesa en aquests darrers anys.

L'enfoc de les estadístiques tradicionals ja no són la única forma d'arribar a conclusions a partir de les dades. Després de fer un breu incís en l'estadística tradicional, s'exposen els diferents enfocs que permetran realitzar anàlisis alternatives.

En estadística tradicional, la predicció i la interpretació són dos objectius en l'anàlisi de les dades. La predicció permet preveure quines seran les respostes a futurs valors de les variables d'entrada, és a dir, un vector de variables X independents. La interpretació consisteix en extraure alguna relació que associï les variables resposta amb les variables d'entrada. En el cas de que les dades siguin generades per un model estadístic paramètric determinat, els valor dels paràmetres s'estimen a partir de les dades i el model s'utilitza després tant per a la predicció com per a la interpretació. Aquests models tradicionals engloben la gran majoria de tots els models estadístics.

En Breiman (2001) l'autor parla per primera vegada d'una nova cultura de modelatge algorítmic que avui en dia forma part de la ciència de dades. Aquesta cultura, on hi entren tècniques com ara el Random Forest o neural networks, proporciona usualment més precisió en la predicció comparada amb els models estadístics tradicionals, encara que aquests últims habitualment expliquen millor la relació entre les variables resposta i les predictores, és a dir, tenen una millor interpretació. Per aquesta nova cultura, l'objectiu és trobar una funció que operi en les variable/s predictores per predir les respostes.

Un compromís entre la potència predictora de la cultura algorítmica i la interpretabilitat de la cultura estadística consisteix en millorar la interpretabilitat dels algorismes de predicció. En aquest sentit, Breiman (2001) ja feia alguna proposta basada en la permutació aleatòria dels valors de cada variable predictora en un conjunt test. Delicado i Peña(2020) proposen un mètode alteratiu (que es basa en el que ells anomenen "variables fantasma") que serà descrit a la secció 4.

Aquest treball es basa en l'article de Delicado i Peña (2020). L'objectiu principal d'estudi és crear una llibreria d'R que implementi les mesures de rellevància descrites en l'article de Delicado i Peña (2020) per a respostes com les del model lineal de regressió. A partir de l'exemple d'un cas real, s'ha testejat la rellevància per variables fantasma i per permutacions aleatòries. S'ha fet servir el modelatge algorítmic, com els arbres de decisió Random Forest o la regressió lasso.

Com a metodologia emprada, s'ha recopilat diverses fonts bibliogràfiques pel desenvolupament dels continguts teòrics ja sigui per la introducció del Machine Learning o per la Interpretabilitat. Pel que fa al tractament de les dades, tant les simulades com les dades reals, s'ha utilitzat el software RStudio. També s'ha fet servir paquets necessaris per el desenvolupant dels exemples i gràfics.

I. PART TEÒRICA

Abans d'exposar la part pràctica, es considera important realitzar una part teòrica que serveixi com a base per a l'estudi. Així doncs, en aquest apartat s'adquirirà totes les nocions necessàries per al desenvolupament del projecte.

Aquests apartats es divideixen en l'exposició dels continguts bàsics de Machine Learning, els conceptes d'interpretabilitat i rellevància de variables, els mètodes fets servir per a mesurar-la, la seva implementació com a llibreria del software R, ja sigui les comandes que s'ha fet servir com la pujada del projecte en la pàgina web de Github i per últim s'inclou un resum de l'article de Delicado i Peña que ha servit de pilar per desenvolupar la llibreria del software R.

1. Machine Learning

Primer de tot, es defineix què és un algorisme com a concepte per entendre Machine Learning. Un algoritme és un conjunt de regles que segueix un ordinador per trobar un objectiu determinat. Són algorismes, per exemple, les receptes de cuina. Les entrades són els ingredients, la sortida és el menjar cuinat i els passos de preparació i cocció són les instruccions de l'algorisme.

El Machine Learning és un conjunt de mètodes que permeten als ordinadors aprendre de les dades per fer i millorar prediccions. Aquets mètodes suposen un canvi respecte a la programació estàndard. Els passos del Machine Learning són els següents. Primer es recopilen les dades, com més millor. Aquestes han de contenir el resultat que es vol predir i la informació addicional per a realitzar la predicció. Seguidament, s'introdueix aquesta informació en un algorisme d'aprenentatge automàtic que genera un model. Finalment, s'utilitza el model amb dades noves i s'integra a un producte o procés.

2. Interpretabilitat

La interpretabilitat és el grau en què un ésser humà pot predir el resultat d'un model. Així es dedueix que, com més alta sigui la interpretabilitat d'un model, més fàcil és que un individu comprengui les determinades decisions o prediccions. Un model té millor interpretabilitat que un altre si les decisions del primer són més fàcils d'entendre. Amb altres paraules, la interpretabilitat és molt important ja que ens pot proporcionar una explicació comprensible de les prediccions d'un model per guanyar més enteniment del problema que es tracta.

En alguns casos no té importància saber per què es pren una decisió, mentre que el rendiment predictiu sigui bo, però, en d'altres casos el saber el per què ens pot ajudar a conèixer millor el problema, les dades i la raó per la què un model podria fallar. La necessitat de la interpretabilitat sorgeix quan en el model també ha d'explicar com va arribar a la predicció.

3. Mètodes de predicció

3.1 Regressió Lineal (LM)

És un model que relaciona de manera lineal una variable resposta amb una o més variables predictores o explicatives.

3.2 Models lineals generalitzats (GLM)

Els GLM (Generalised Linear Models) serveixen per a dades que no compleixen amb el supòsit de normalitat. Permeten modelar dades de comptatge, dades binàries, dades de proporcions i dades inflades per zeros.

3.3 Models additius generalitzats (GAM)

Els GAM (Generalised Additive Models) s'utilitzen quan els nostres dades no compleixen amb el supòsit de normalitat ni el de linealitat.

3.4 Arbres de decisió

Un arbre de decisió és una estructura similar a un diagrama de flux. Els models basats en aquests arbres divideixen les dades diverses vegades segons certs valors de tall en les característiques. A través de la divisió, es creen diferents subconjunts del conjunt de dades. Els subconjunts finals es denominen nodes terminals o fullesa i els subconjunt entremitjos es denominen nodes interns o nodes de divisió. Els arbres poden utilitzar-se per la classificació i la regressió. En el treball s'usen els arbres de regressió.

3.4.1 Arbre de regressió

L'objectiu és predir la variable resposta y en funció de covariables. En el treball s'ha utilitzat el random forest que consisteix en un gran nombre d'arbres que permeten assolir una millor precisió i estabilitat del model.

3.5 Ridge Regression

És una extensió de la regressió lineal en que la funció de pèrdua es modifica per minimitzar la complexitat del model. Aquesta modificació es fa afegint un paràmetre de penalització que equival al quadrat de la magnitud dels coeficients.

3.6 Lasso Regression

És una modificació de la regressió lineal. La funció de pèrdua es modifica per minimitzar la complexitat del model. A la suma de quadrats dels residus s'afegeix la suma dels valors absoluts dels coeficients del model multiplicada per un paràmetre de penalització lambda.

4. Rellevància de les variables (Delicado i Peña, 2020)

L'article de Delicado i Peña (2020) és la base principal d'aquest TFG i, en particular, del paquet *ghostvar*. En Delicado i Peña (2020) es proposa una forma d'assignar una mesura de rellevància per a cada variable explicativa en un model predictiu complex. Per aquest model, es té un conjunt d'entrenament (*training set*) i un conjunt de proves (*test set*). Per tal de mesurar la rellevància individual de cada variable es comparen les prediccions del model en el conjunt de proves obtingudes de dues formes. Primer es fan les prediccions al conjunt de proves de la forma habitual, sense modificar aquest conjunt de proves. Anomenarem $\hat{y}$ al vector de prediccions de la variable resposta al conjunt de proves obtingudes d'aquesta manera.. Segon, per mesurar la rellevància de la variable explicativa *j* aquesta és substituïda al conjunt de proves per la predicció d'aquesta variable a partir de la resta de variables explicatives. Aquesta predicció de la variable explicativa *j* s'anomenarà variable fantasma *j*. Denotarem com $\hat{y}_j$ a la predicció de la variable resposta quan es fa servir la variable fantasma *j* en comptes de la variable explicativa *j.* Si $\hat{y}_j$ és semblant a $\hat{y}$ aleshores deduïm que la variable explicativa *j*, per sí mateixa, aporta poc a la predicció, donat que la seva aportació és semblant a la que fa la seva variable fantasma, que és funció de les altres variables explicatives. Per contra, quan $\hat{y}_j$ i $\hat{y}$ són molt diferents tenim una senyal clara de que la variable explicativa *j* aporta informació rellevant sobre la resposta que no és compartida amb les altres variables explicatives i, per tant, la mesura de la rellevància de la variable *j* ha de ser alta. Als vectors ($\hat{y}_j$ - $\hat{y}$**)** els anomenarem vectors dels efectes individuals de la variable *j*, i a la matriu que té per columnes els vectors d'efectes individuals de totes les variables explicatives l'anomenarem matriu de rellevància. A continuació, es compara la rellevància conjunta de les variables utilitzant els valors propis de la matriu de covariàncies de la matriu de rellevància. Aquesta forma de mesurar rellevància de variables és vàlida per a qualsevol model predictiu, tant si és un model estadístic (models lineals, generalitzats o additius, per exemple) com si és un model algorítmic (xarxes neuronals, per exemple) S'exposarà a continuació un exemple simulat.

4.1 Exemple simulat

En la realització del paquet s'ha exposat l'exemple 1 de l'article del Delicado i Peña (2020). En aquest exemple s'utilitza un model lineal amb tres variables explicatives com a mecanisme de generació de dades. La primera és independent de les altres dos i aquestes últimes estan altament correlacionades. Es genera una mostra d'entrenament i una mostra de prova per tal de calcular la rellevància individual de cada variable i la matriu de rellevància utilitzant les variables fantasma. Després de la realització dels gràfics pertinents es conclou, a partir del primer gràfic, que la primera variable explicativa és la més rellevant. Les altres dues variables tenen una rellevància similar i fortament relacionada.

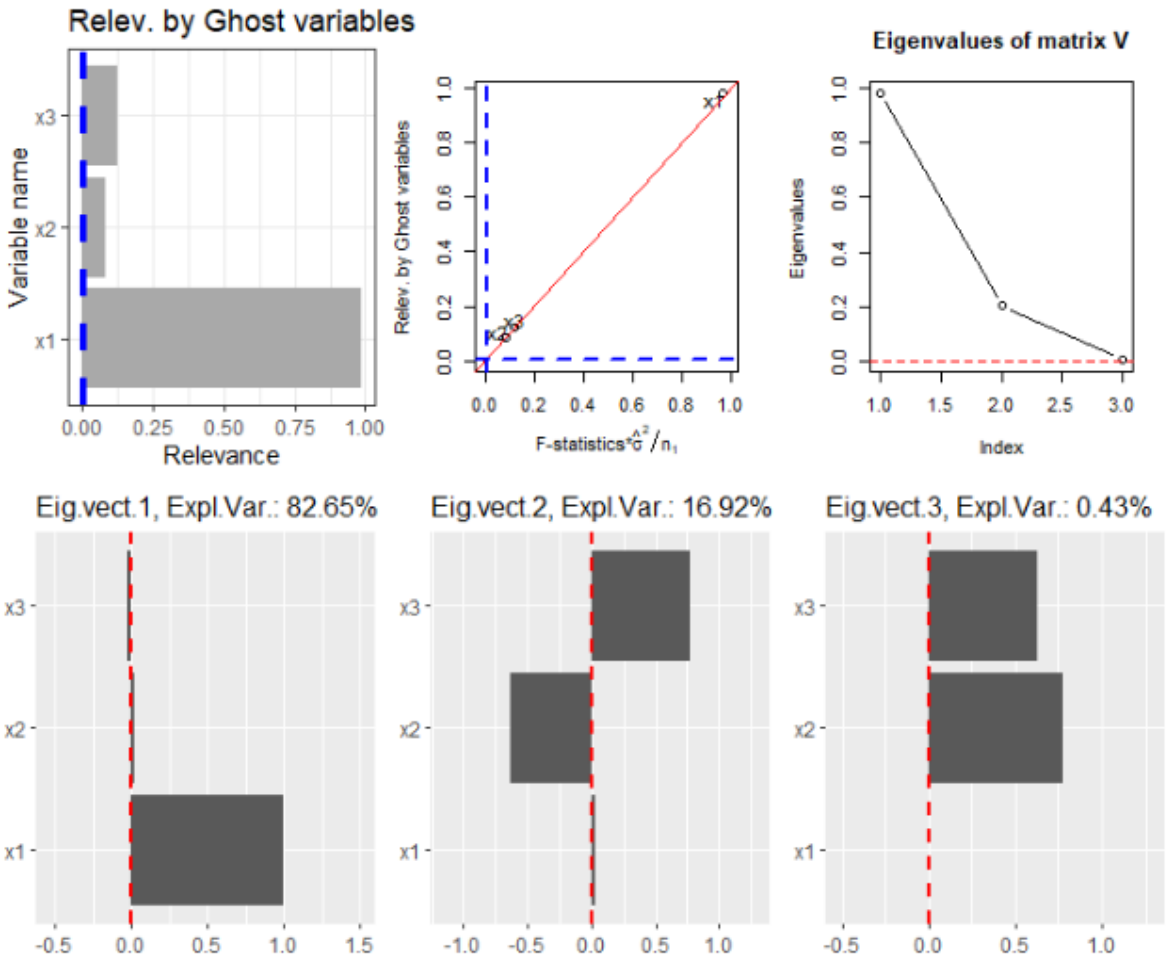

Figura 1: Rellevància per Variables Fantasma exemple simulat

En el segon gràfic, es mostra la rellevància per permutacions aleatòries. Les tres variables tenen una rellevància similar.

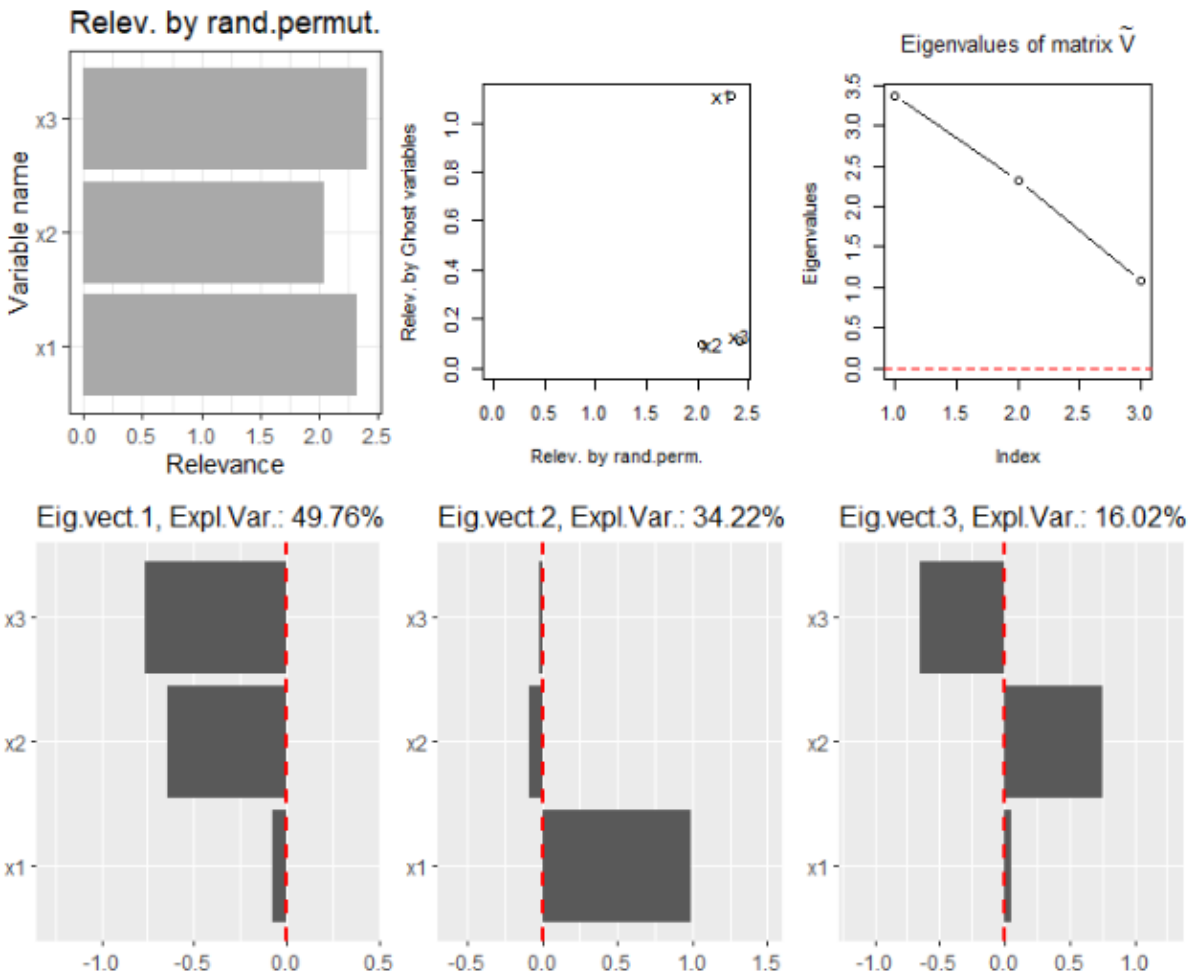

Figura 2: Rellevància per Permutacions Aleatòries exemple simulat

Finalment s'arriba a la conclusió en els gràfics exposats anteriorment que la informació que proporcionen les variables fantasma és més útil ja que s'ha trobat una associació entre dues variables explicatives. També la variable X_1 és la que té més rellevància en comparació a l'obtinguda mitjançant permutacions aleatòries.

5. Implementació dins de l'entorn R

Abans de descriure el paquet de R on s'implantaran les mesures de rellevància de variables donaré una breu definició de què és un paquet o llibreria. Un paquet és una col·lecció de funcions, dades i codi d'R que es troben en una carpeta amb una estructura ben definida. Mentre que una llibreria és un directori que conté paquets instal·lats. Es pot tenir com a mínim dos tipus de llibreries diferents a l'ordinador. Un pels paquets que s'instal·len i altres pels paquets que venen definits pròpiament pel software R, com per exemple, bases de dades.

En aquest treball s'ha creat una llibreria d'R que implementa les mesures de rellevància descrites a l'article de Delicado i Peña (2020) per a respostes com les del model lineal de regressió. Aquesta llibreria s'ha anomenat *ghostvar*, i es pot trobar al sistema de gestió de

projectes Github. A continuació, s'esmenten tots els passos realitzats fins al final de la creació del paquet d'R.

Al crear un projecte tots els fitxers queden vinculats directament al projecte. Per tal de la realització del paquet d'R es crea un, dins de Rstudio en el *file* seguit de *New Project*, en el directori que s'escull i es selecciona també l'opció *R Package*. Més tard s'introdueix el nom que tindrà el paquet i es clica a *Create Project*. En el nou directori apareixerà una carpeta anomenada *R* on s'ha de posar totes les funcions necessàries per a la realització del paquet, una altra carpeta anomenada *man* on es crearà automàticament, després de diverses comandes que s'esmentaran més tard, la funció d'R en format Rd. També s'haurà d'omplir els arxius de *DESCRPITION* i de *NAMESPACE*.

En tercer lloc, es va fer una descripció, *DESCRIPTION*, del paquet necessari per la compilació. Els camps necessaris són el nom, tipus, títol, versió, autor/autors, persona de manteniment, importacions, descripció, llicencia i el tipus de codificació de caràcters. Fent un breu resum, eren necessaris tots aquest camps per tal de saber qui és el creador, contribuïdor i la persona que mantindrà el paquet. També s'havien d'importar tots els paquets creats per altres usuaris per poder realitzar gràfics i altres funcions. Per últim, la versió del paquet també havia de ser indicada. Inicialment, la primera versió d'una llibreria d'R ha de ser la 0.0.0.9000, a partir de diversos canvis l'usuari pot anar incrementant el nivell de versió quan ho cregui convenient.

En quart lloc, s'ha creat el fitxer *NAMESPACE*. Aquest arxiu és important ja que diferencia dos paquets que possiblement podrien tenir el mateix nom, però, funcions totalment diferents. Aquest fitxer s'ha creat automàticament, quan s'han executat les instruccions de creació del paquet.

Després de la descripció i el *Namespace,* s'ha de crear l'ajuda d'R del paquet a partir de totes les funcions necessàries. En les diverses funcions ha estat necessari escriure amb format Roxygen2. Són comentaris especialment estructurats abans de cada definició de funció i es processen per produir posteriorment els fitxers *.Rd*. Els comentaris de Roxygen2 són només comentaris d'R precedits per el símbol *#*' (és a dir, del coixinet i seguidament d'un apòstrof) per tal de distingir els comentaris de Roxygen2 dels comentaris d'R estàndards. S'ha fet servir un arrova(@) i posteriorment els noms adequats per tal d'obtenir la "description", "usage", "arguments", "values", "see also," "examples" necessaris en l'ajuda del paquet.

5.1 Creació del paquet

El primer pas s'executa si ja es té implementat versions anteriors del paquet. Així doncs, el primer que es fa és esborrar els arxius Rd de la carpeta *man* i també esborrar l'arxiu *NAMESPACE* perquè l'R els creï a partir de tot el codi implementat. Després, es crea els arxius .Rd a partir del següent codi*: devtools::document()*. Posteriorment, es carrega la llibreria devtools. Per últim, s'instal·la el paquet i es construeix en el meu directori l'arxiu *.tar.gz* amb les següents comandes: *install()* i *build().* A partir de tots aquest passos s'ha obtingut la llibreria.

A continuació es comenta com s'ha pujat al sistema de gestió de projectes Github.

5.2 Github

Per tenir un millor seguiment amb el tutor del TFG es van pujar el paquet i totes les funcions al web de Github, mitjançant el qual es permet compartir el codi amb altres usuraris, poder sol·licitar millores i resoldre problemes. Es va descarregar el programa Git, que permet, a partir de comandes de Git Bash pujar tots els arxius a Github. Posteriorment, totes les actualitzacions de les funcions i el paquet es van pujar al web en l'opció de Github de RStudio. Es van seleccionar els arxius que es volien afegir al repositori i es va clicar el botó Commit, entrant una breu descripció on deia Commit message. Es puja (*push*) el resultat de l'enviament (*commit*) a Github de forma que es té una copia de seguretat i també és accessible al tutor del TFG.

II. PART PRÀCTICA

Al segon exemple es tracten dades reals d'habitatges de lloguer procedents de la pàgina web de la Inmobiliaria Idealista, que permet als clients buscar habitatges en funció dels diferents criteris entre les ofertes publicades per altres clients. S'ha predit el logaritme dels preus de lloguer en funció de 16 variables explicatives. A l'article de Delicado i Penya (2020) s'ha provat tres models predictius (regressió lineal, model additiu i una xarxa neuronal). En aquest TFG s'han afegit tres models més. Per a cada model s'ha calculat la rellevància de les variables tant per a variables fantasma com per a permutacions aleatòries. El conjunt d'entrenament conté el 70% de les dades i el 30% restant formen el conjunt de proves. En el cas dels models de l'article, tant per el model lineal com per a l'additiu l'anàlisi de la rellevància representa una informació complementaria ja que el resultat estàndard ofereix una bona informació sobre la significació estadística de cada variable explicativa. Aquests dos mètodes no són molt diferents l'un de l'altra i, per tant, parlarem aquí només del mètode lineal. Pel cas del model de la xarxa neuronal l'anàlisi de rellevància proporciona nous coneixements que s'exposaran a continuació. Per últim, per a cada model es mesura la qualitat de predicció en la mostra. Aquesta mesura és molt similar a la del coeficient de determinació i, com més proper a 1 sigui el resultat es podrà dir que la predicció és millor i per tant, el model és millor. Seran en aquest models els més idonis per fixar-se en les rellevàncies de les variables descrites pel article de Delicado y Peña (2020).

Model lineal

S'ha estudiat la rellevància per a les variables fantasma i també la rellevància per a permutacions aleatòries per el model lineal.

En el primer cas, després de realitzar totes les comandes necessàries per a la realització dels gràfics pertinents s'observa en la figura 3, les 7 variables més rellevants la més destacada és la variable *log.size*. En el gràfic situat a la segona fila i primera columna mostra el primer vector propi que representa quasi el 60% de la rellevància total relacionat majoritàriament al *log.size*. El segon vector propi representa un 15% de la rellevància total relacionat al nivell de preus del districte, mentre els cinc següent vectors propis representen menys del 25% de la rellevància total.

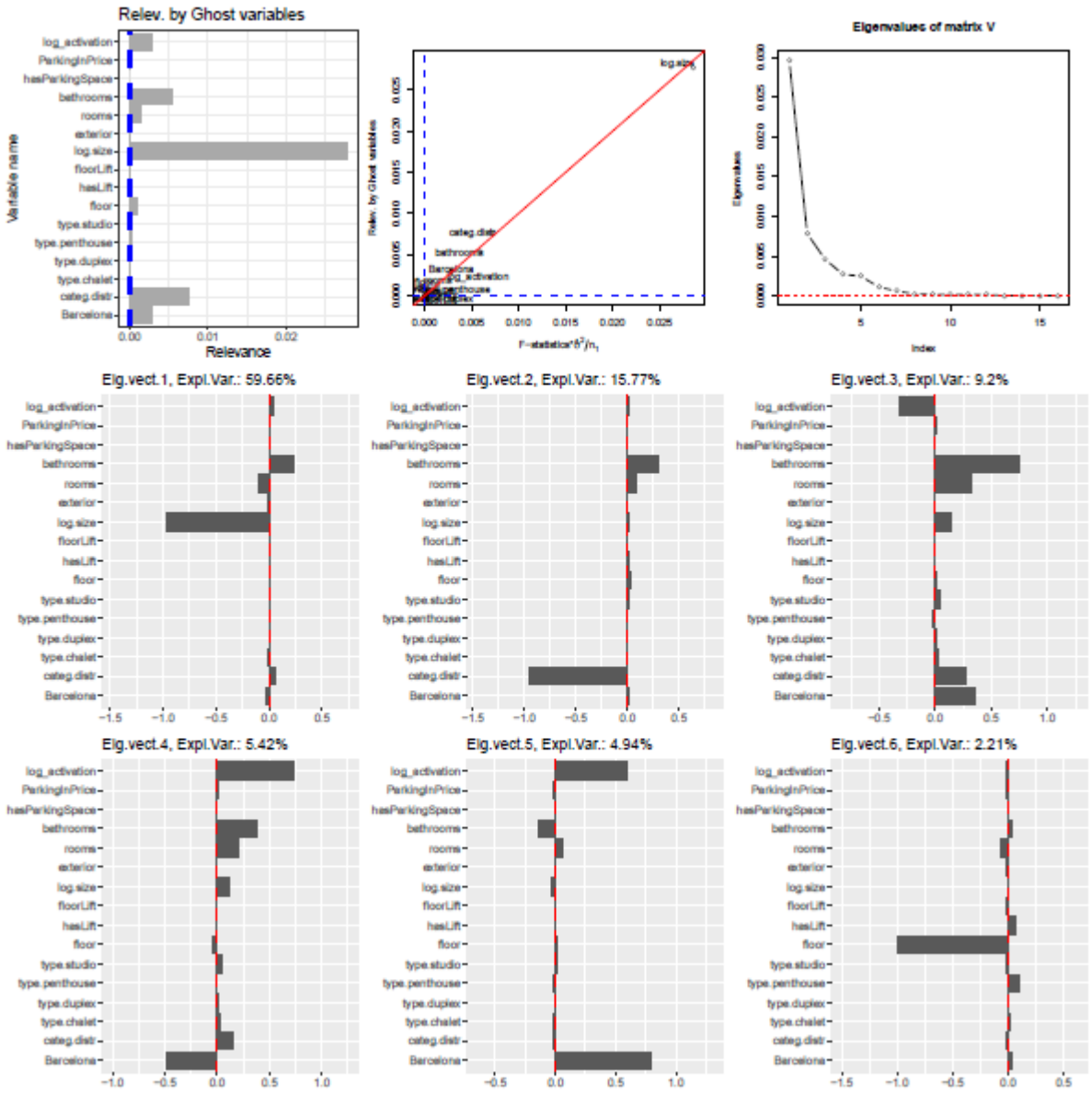

Figura 3: Rellevància per Variables Fantasma model lineal exemple Idealista

En el cas de la rellevància per permutacions aleatòries s'observa en la figura 4 que el resultats són lleugerament similars que els anteriors. Ara es té 6 vectors propis que tenen més de l'1% de rellevància total. S'ha exclòs el setè vector propi ja que representa menys de l'1% de la rellevància total.

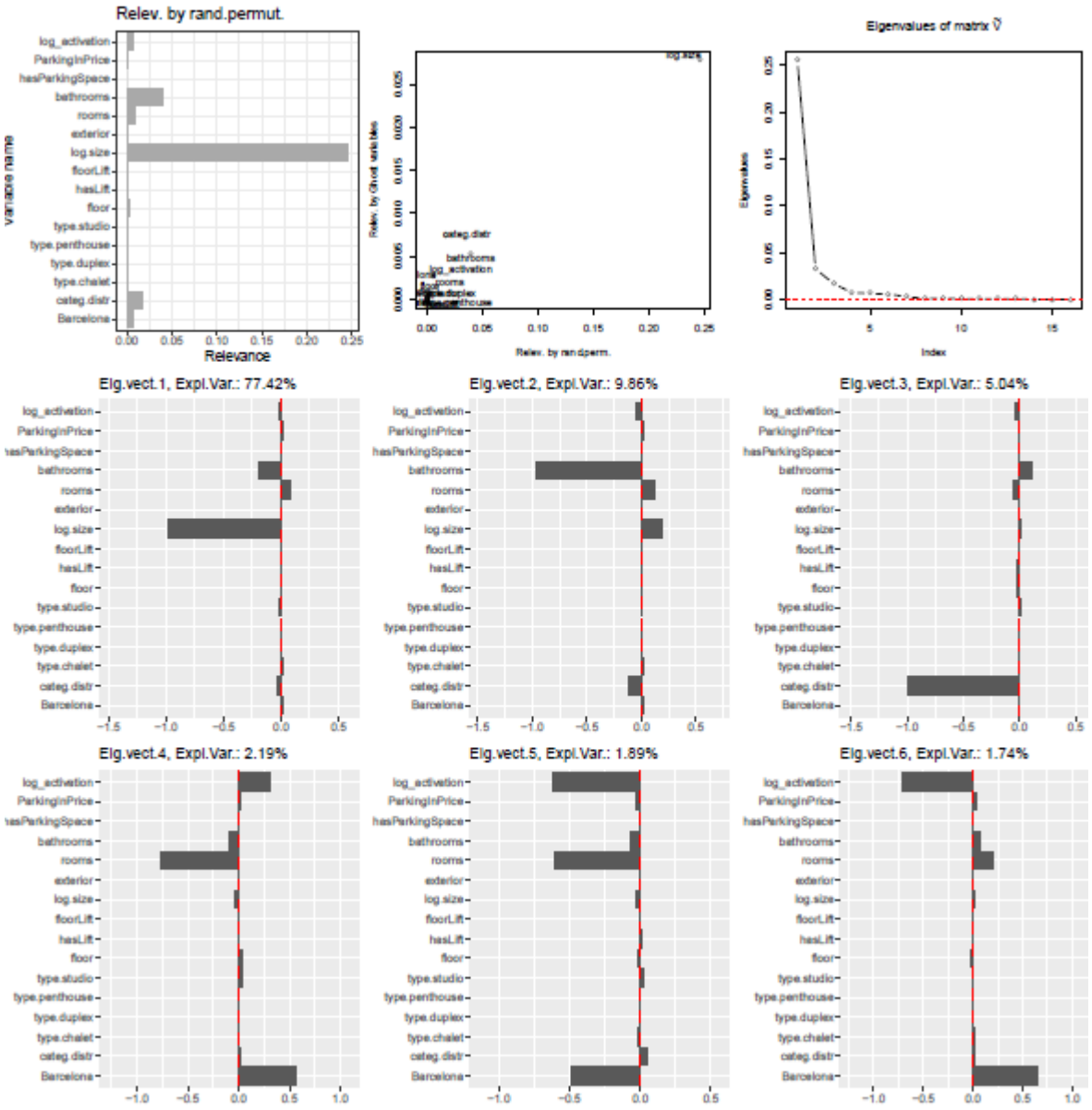

Figura 4: Rellevància per Permutacions Aleatòries model lineal exemple Idealista

Fent la comparativa entre rellevàncies, la principal diferencia de la rellevància de les permutacions aleatòries respecta a la rellevància per a les variables fantasma és que en la primera no s'ha detectat interaccions entre les variables explicatives.

Per últim, la mesura de qualitat de predicció per aquest model és de 0.79, valor proper a 1, per tant, la predicció del model és bastant bona.

Xarxes neuronals

Els *tunning parameters*, el nombre de neurones i el *decay parameter* s'han escollit utilitzant la llibreria de caret mitjançant el *10-fold cross validation* en el conjunt d'entrenament.

Els resultats obtinguts sobre la rellevància per variables fantasma per aquest model es mostren a la figura 5. Partint de les 7 variables ja rellevants per el model lineal i additiu, s'han afegit 4 més. També, per aquest model hi ha 10 vectors propis amb una rellevància total explicada superior al 1% per cada vector propi. El primer vector propi representa gairebé un 45% de la rellevància total relacionat majoritàriament al log.size. El segon vector propi representa un 15% de la rellevància total relacionat a la categoria distr. El tercer vector propi representa aproximadament un 10% de la rellevància total explicada relacionat, en gran part, a la variable tipus d'habitatge. Els altres 7 vectors propis corresponen aproximadament a un 25% de la rellevància total.

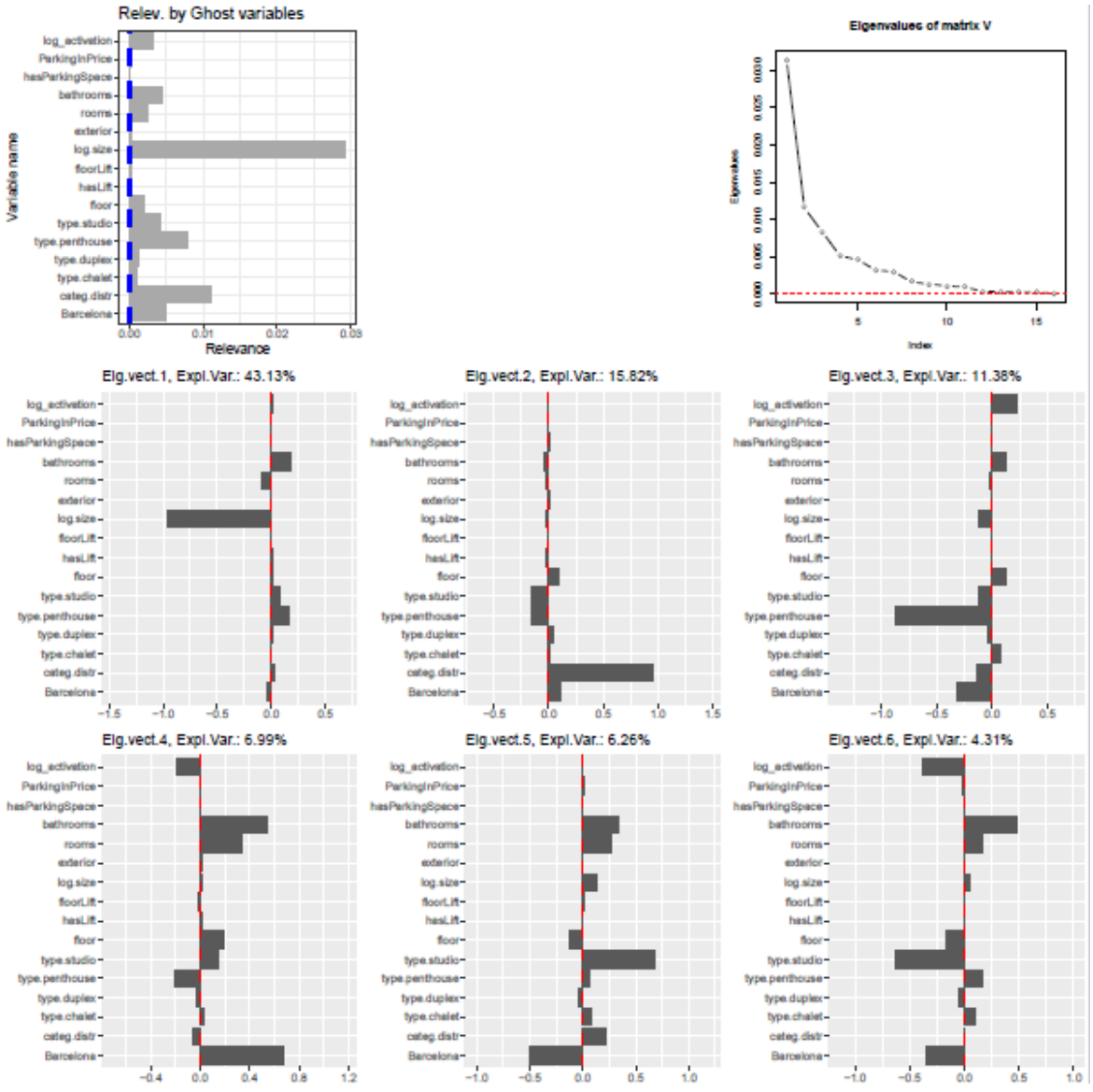

Figura 5: Rellevància per Variables Fantasma xarxes neuronals exemple Idealista

En el cas de la rellevància per permutacions aleatòries s'observa en la figura 6 que les dos variables més rellevants són floor i bathrooms respectivament. Les variables floor i bathrooms dominen els dos primers vectors propis de la matriu de rellevància. També s'observa com ara es té 12 vectors propis mentre l'últim explica menys del 1% de la rellevància total.

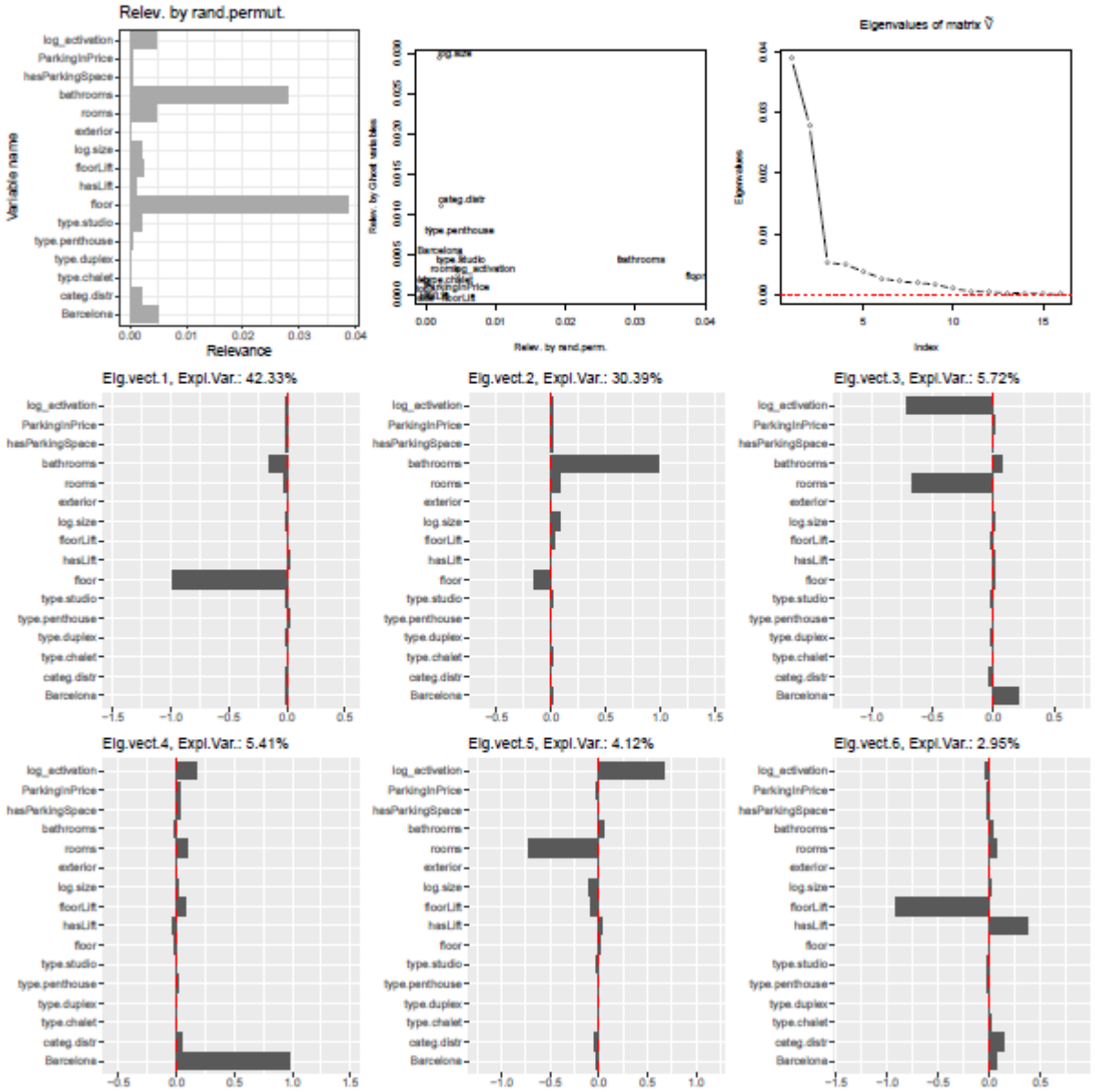

Figura 6: Rellevància per Permutacions Aleatòries xarxes neuronals exemple Idealista

Per últim, la mesura de qualitat de predicció per aquest model és de 0.8, valor proper a 1, per tant, la predicció del model és bastant bona.

S'ha introduït quatre nous models per l'exemple de l'idealista.

Regression Tree

S'ha utilitzat el Recursive Partitioning and Regression Trees de la llibreria rpart.

Després d'obtenir els gràfics adients per la rellevància per variables fantasma s'observa que les variables *log.size*, i *categ.distr* són les més rellevants i en aquest mateix ordre.
Per altra banda, el primer vector propi, que representa aproximadament el 80% de la rellevància total, fa referencia a la variable *log.size*. El segon vector propi representa un 20% de la rellevància total i fa referencia exclusivament a la variable *categ.distr.* S'observa com tots els vectors propis restants representen un percentatge nul de rellevància total.

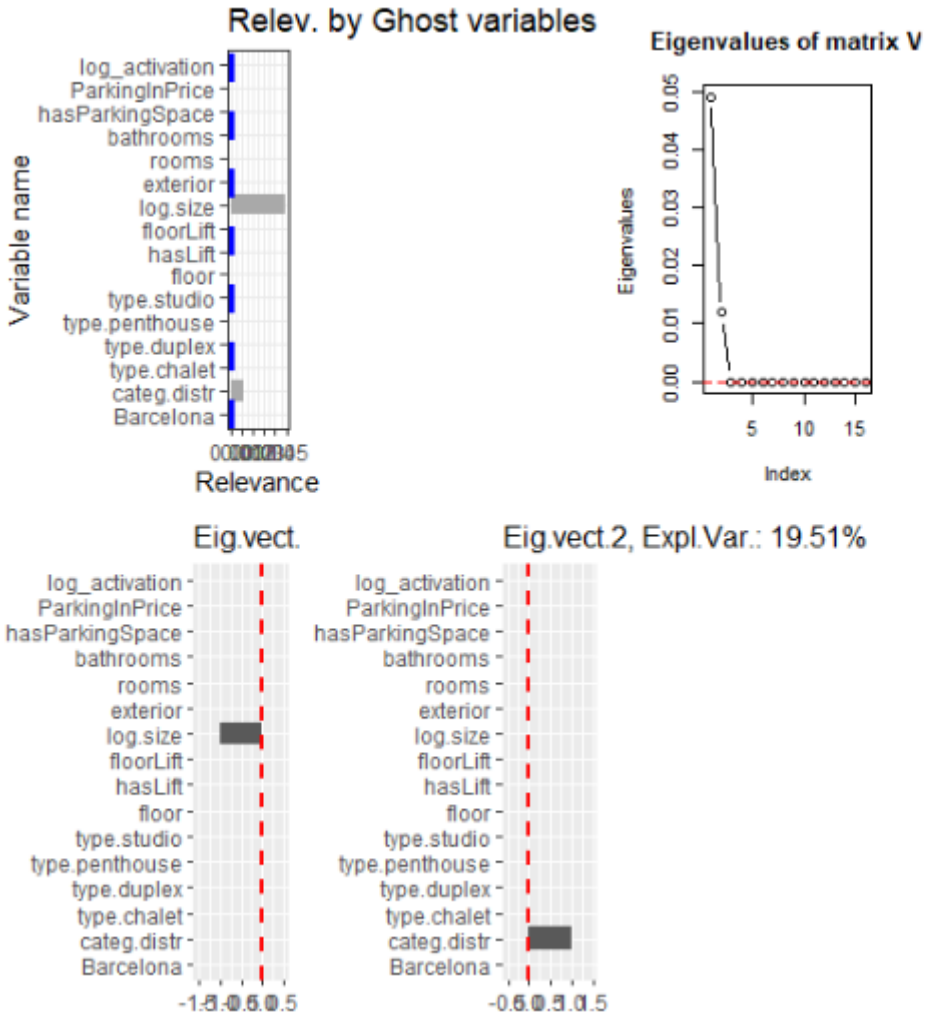

Figura 7: Rellevància per Variables Fantasma Regression Tree exemple Idealista

En el cas de la rellevància per permutacions aleatòries la variable *log.size* és la més rellevant, seguidament de la variable *categ.distr.* En el gràfic de la segona fila i primera columna de la figura 8, es pot apreciar com, gairebé tota la rellevància total és explicada per la variable *log.size* (gairebé un 100%). El segon vector propi conté un 5% de la rellevància total representada per la variable *categ.distr.* Així doncs, hi ha dos vectors propis que aporten informació important.

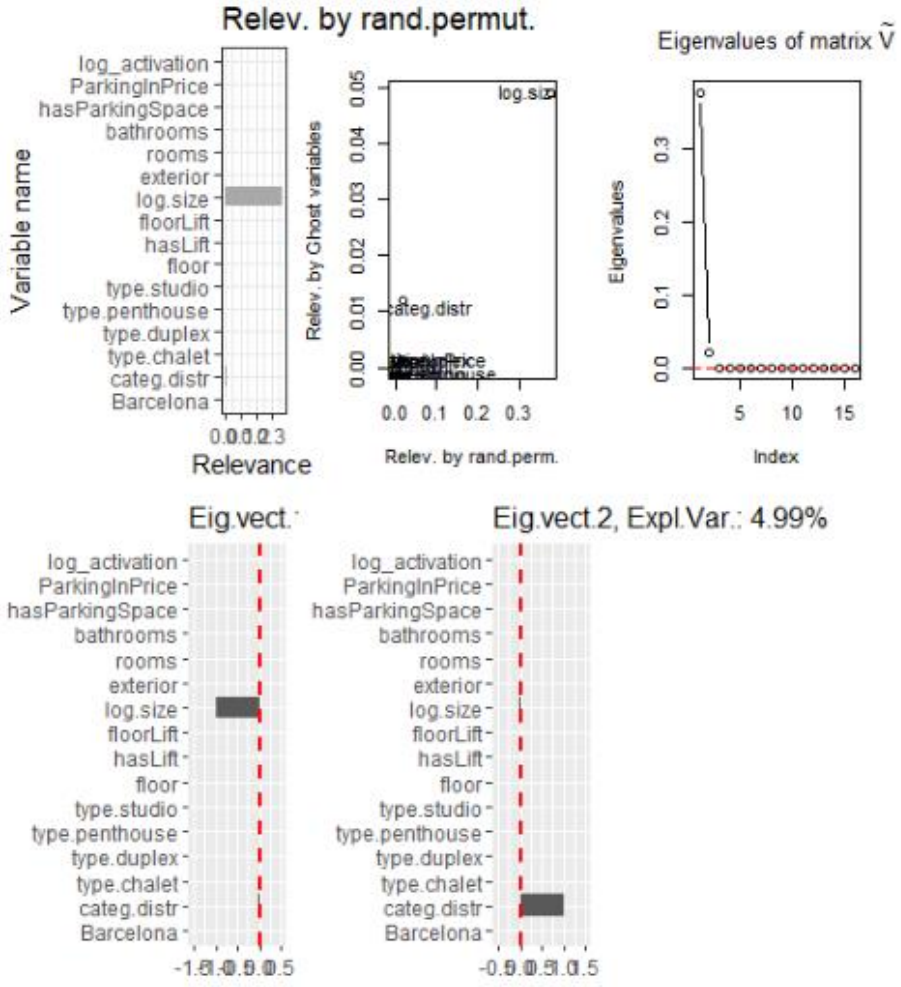

Figura 8: Rellevància per Permutacions Aleatòries Regression Tree exemple Idealista

Per últim, la mesura de qualitat de predicció per aquest model és de 0.69, valor proper a 1, per tant, la predicció del model és bona.

Random Forest

S'ha fet servir la llibreria Random Forest per tal de predir la variable resposta en funció de covariables.

Els resultats obtinguts sobre la rellevància per variables fantasma per aquest model es mostren a la figura 9. Es pot observar que la variable *log.size* és la més rellevant mentre que les sis restants prenen poca importància en comparació a aquesta. Fent referència al requadre del primer vector propi, aquest representa aproximadament un 50% de la rellevància total i correspon en gran part a la variable *log.size*. Els sis següents vectors propis representen aproximadament un 40% de la rellevància total.

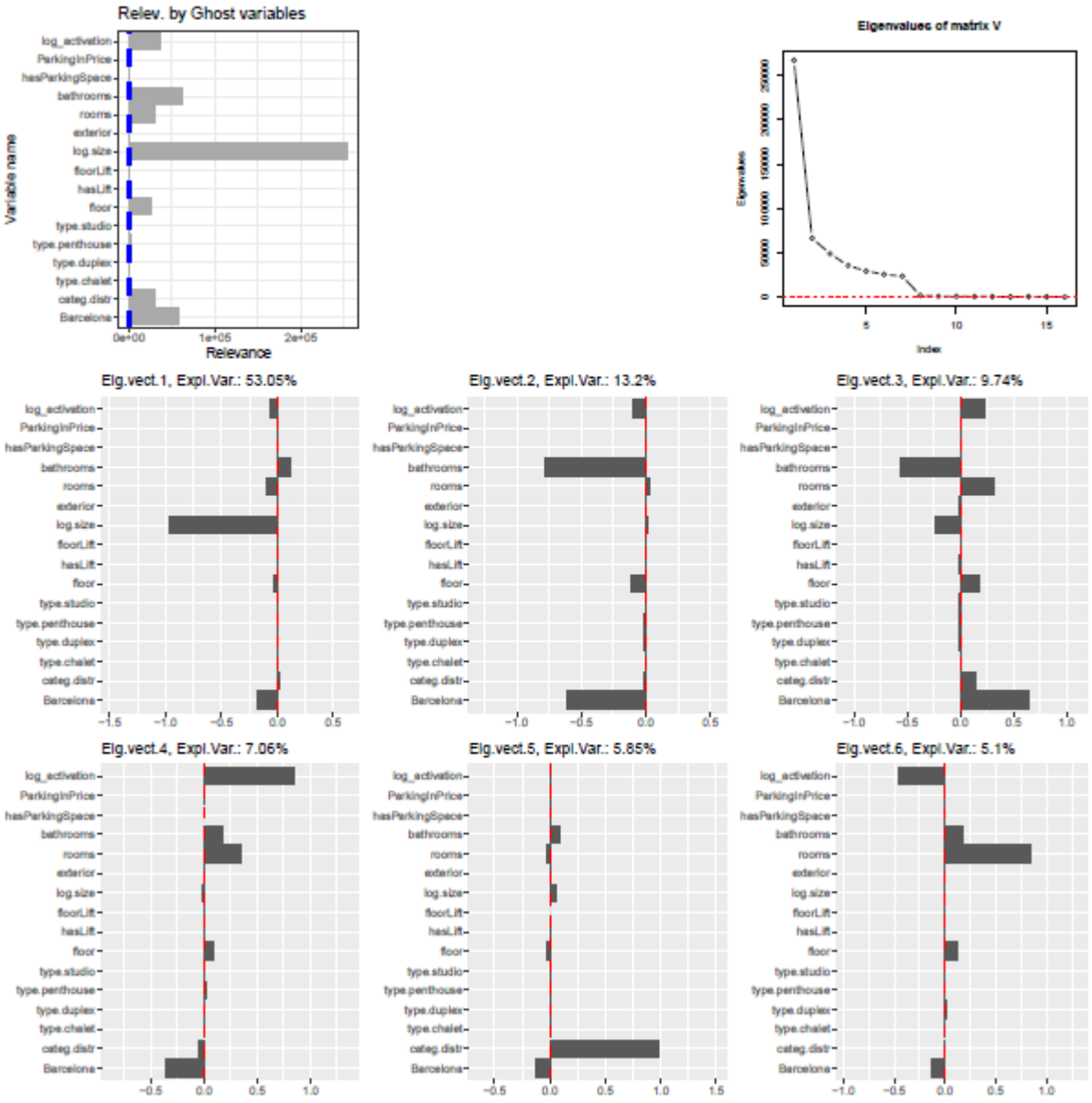

Figura 9: Rellevància per Variables Fantasma Random Forest exemple Idealista

En el cas de la rellevància per permutacions aleatòries la variable *log.size* és la més rellevant, seguidament de la variable *bathrooms* de manera semblant en el cas de la rellevància per variables fantasma. El primer vector propi engloba més d'un 70% i els respectius vectors propis representen menys del 10% cadascú.

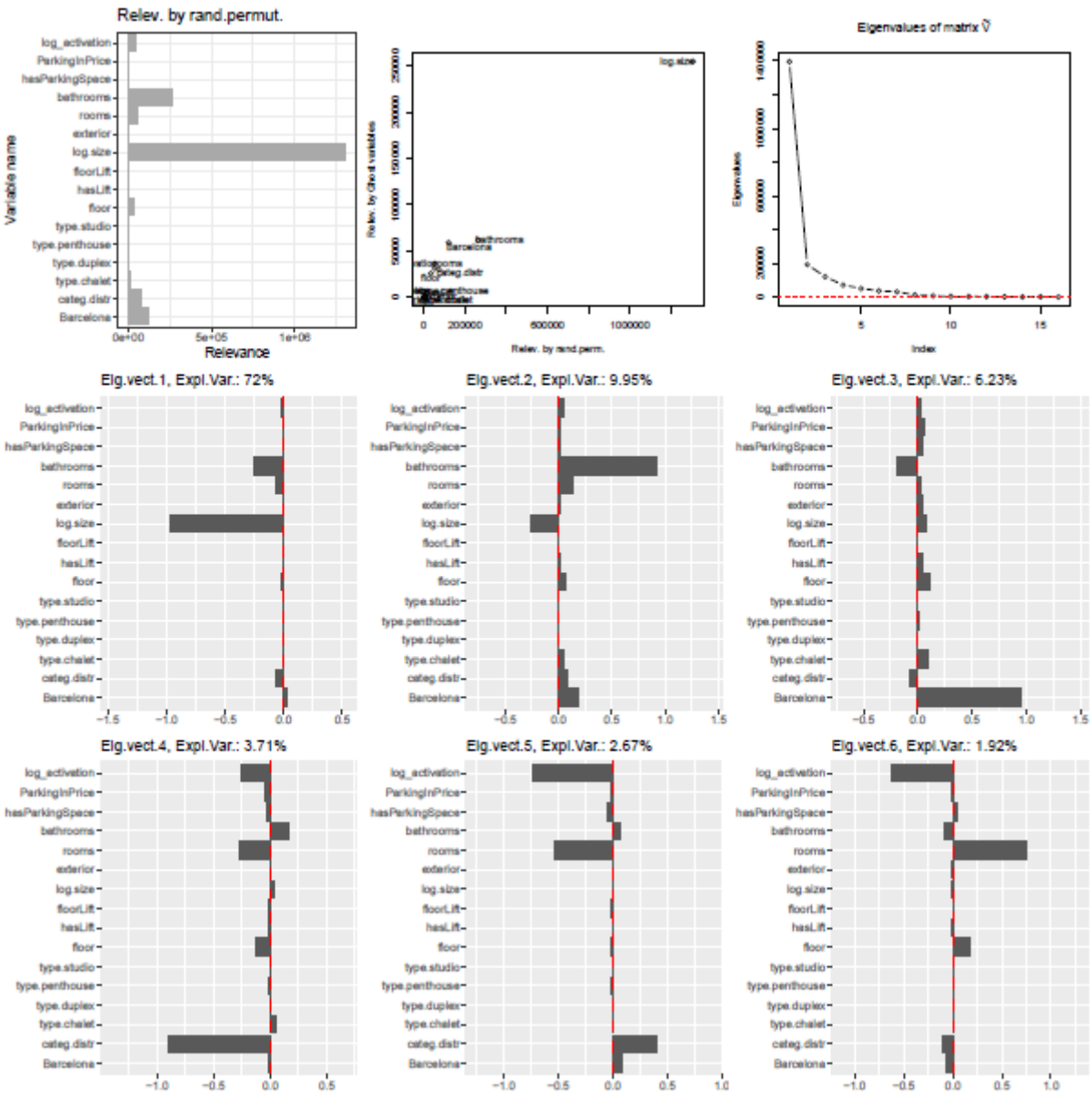

Figura 10: Rellevància per Permutacions Aleatòries Random Forest exemple Idealista

Per últim, la mesura de qualitat de predicció per aquest model és de 0.67, valor proper a 0.5, per tant, la predicció del model és regular.

Lasso Regression

Per tal d'estudiar la rellevància per variables fantasma i permutacions aleatòries en els casos de Lasso regression i de Ridge Regression s'ha fet servir el paquet d'R glmnet. A continuació es mostren les dues figures que corresponen al model Lasso Regression. El Ridge regression no s'ha explicat ja que els resultats són similars, però, en l'annex està el codi d'R per tal de desenvolupar i poder testejar les possibles rellevàncies. Alhora d'ajustar el paràmetre de penalització lambda d'aquest model, és va triar en un principi la millor lambda, és a dir, la que minimitzava el Mean Squared Error estimat per k-fold cross validation. Més tard, es va observar que els resultats eren molt similars al del model lineal i, per tant, es va escollir la lambda que la llibreria glmnet anomena One Standard Error: és més gran que la lambda òptima però el seu Mean Squared Error està només a una desviació estàndard del corresponent al lambda òptim. Com que el lambda One Standard Error és més gran que l'òptim és habitual que la solució corresponent tingui més variables explicatives amb coeficients nuls i, per tant, que el model estimat sigui més fàcilment Interpretable. En l'exemple de l'Idealista només hi ha dues variables amb coeficient no nul i són *log.size* i *bathrooms*.

Pel que fa a la rellevància per variables fantasma, hi ha moltes variables rellevants, encara que les que més destaquen són *bathrooms*, *log.size* i *rooms* respectivament. També es pot observar que hi ha molts vectors propis superiors a l'1% (en la figura 11 es mostren només els sis primers). El primer vector propi engloba només un 34% de la rellevància total corresponent majoritàriament a la variable *bathrooms.* El següent vector propi correspon quasi un 25% mentre que a partir del cinquè ja correspon menys del 10% de la rellevància total.

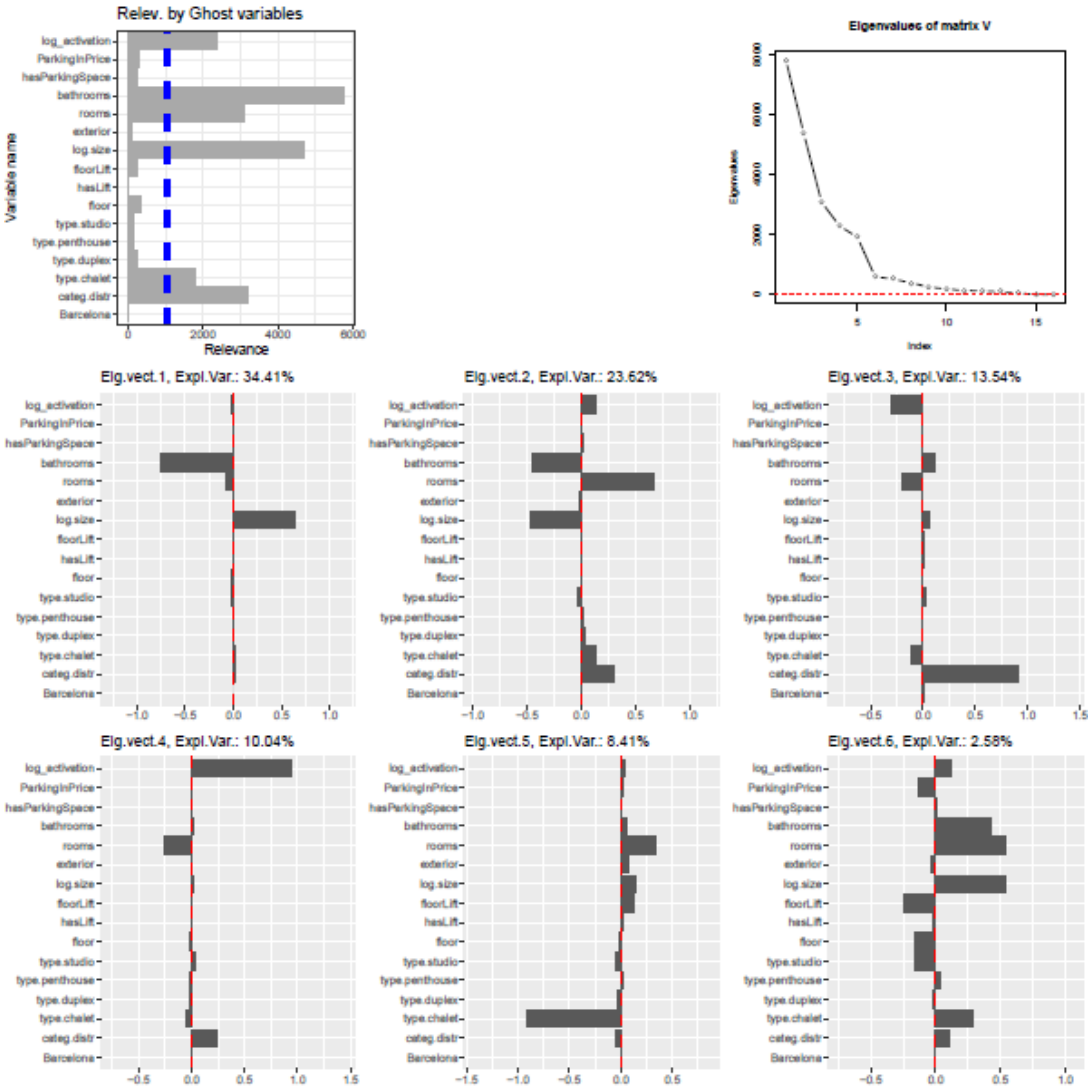

Figura 11: Rellevància per Variables Fantasma Lasso Regression exemple Idealista

En el cas de la rellevància per permutacions aleatòries, els resultats són bastant diferents que els anteriors. Ara només hi ha dues variables rellevants que són *bathrooms* i *log.size* per ordre de variables més rellevants. Ara doncs, hi ha dos vectors propis. El primer representa aproximadament un 75% de la rellevància total mentre que el segon representa gairebé un 25% de la rellevància total.

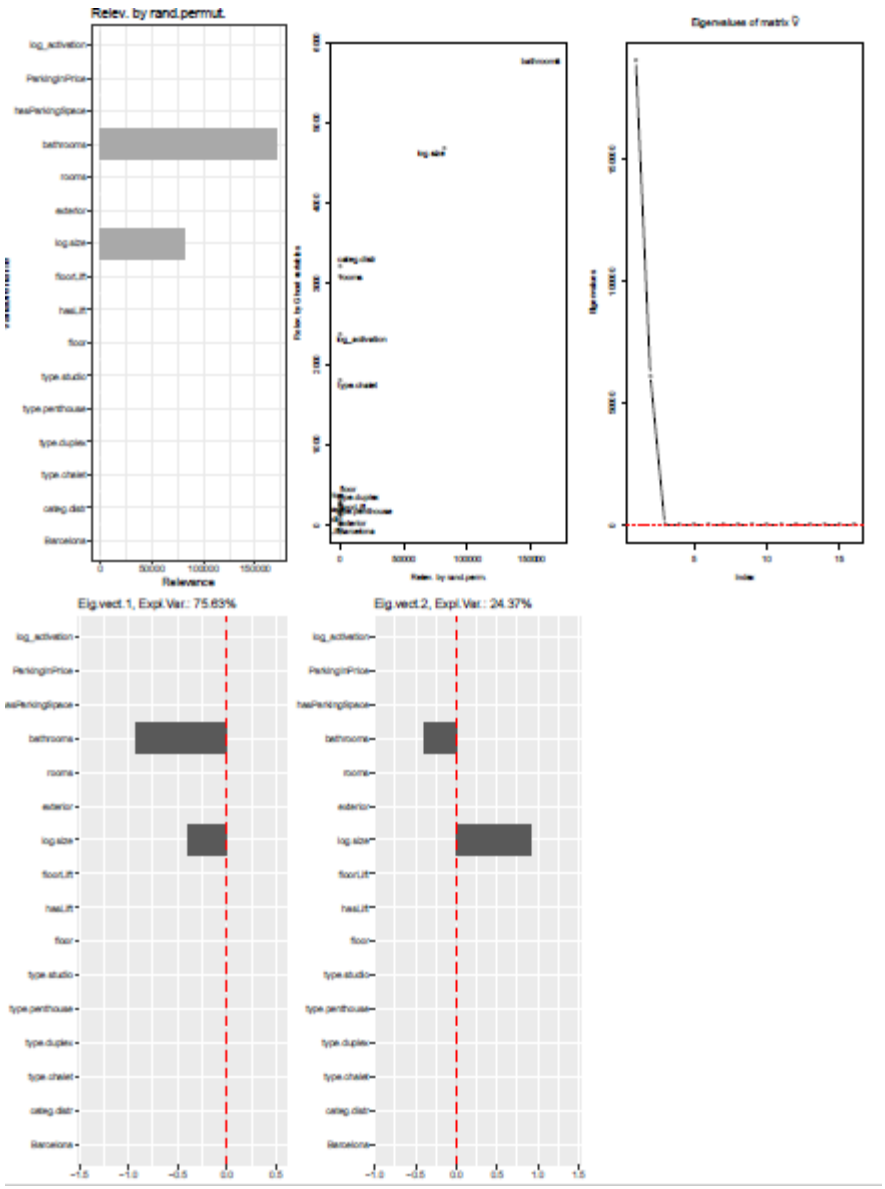

Figura 12: Rellevància per Permutacions Aleatòries Lasso Regression exemple Idealista

Per últim, la mesura de qualitat de predicció per aquest model és de 0.66, valor proper a 0.5, per tant, la predicció del model és regular.

CONCLUSIONS

S'ha posat en pràctica una nova forma de mesurar la rellevància d'una variable en un model predictiu complex. Es compara les prediccions fora de la mostra del model que inclouen aquesta variable amb les d'un model en què aquesta variable és substituïda per la seva variable fantasma, és a dir, la predicció de la variable utilitzant les altres variables explicatives.

En la comparativa entre la rellevància per variable fantasma i la rellevància per permutacions aleatòries hi ha resultats completament diferents. Els resultats de la rellevància per a la xarxa neuronal presenten certes diferencies respecte als models lineal i additiu. Per al model de xarxa neuronal, les variables més importants per a les permutacions aleatòries són les menys importants en els models de regressió o additiu. Cal recalcar que la rellevància per variable fantasma s'ha considerat més variables com a rellevants respecte a les permutacions aleatòries. Es pot arribar a la conclusió que la rellevància per variables fantasma és un bon indicador en els models vistos anteriorment per aquest conjunt de dades.

També s'ha observat, per a tots els models tractats en l'exemple de l'Idealista, que la qualitat de predicció en la mostra és bastant bona. Els models amb una capacitat predictiva més elevada són les xarxes neuronals i el model lineal en aquest ordre.

BIBLOGRAFIA I WEBGRAFIA

BREIMAN, Leo. “Statistical Modeling: The Two Cultures (with comments and a rejoinder by the autor)”. Statist. Sci. 16 (2001), no. 3, 199--231

BROMAN, karl. *Building and installing and R Package* [En línia]. <https://kbroman.org/pkg_primer/pages/build.html> [Consulta: febrer 2020]

DELICADO, Pedro; PEÑA, Daniel. Understanding complex predictive models with Ghost Variables [En línia]. 2020. <https://arxiv.org/abs/1912.06407> [Consulta: juny 2020].

MOLNAR, Christoph. *Interpretable Machine Learning* [En línia]. Lulu.com, 2020. < https://christophm.github.io/interpretable-ml-book/> [Consulta: maig 2020]

WICKHAM, Hadley. *R packages* (1a edició). O’Reilly, 2015.

Annex SCRIPT R

Rpart

```
load(file="rhBM_Price.Rdata")
# rhBM.price, rhBM.priceByArea,
names(rhBM.price)
log.size <- TRUE
if (log.size){
  rhBM.price$size <- log(rhBM.price$size)
  names(rhBM.price)[11]<-"log.size"
}

names(rhBM.priceByArea)

### Training and test sets
n <- dim(rhBM.price)[1]
pr.tr <- .7
pr.te <- 1 - pr.tr
n.tr <- round(n*pr.tr)
set.seed(123456)
Itr <- sample(1:n,n.tr)
Ite <- setdiff(1:n,Itr)
n.te <- n-n.tr

library(rpart)
library(rpart.plot)
library(mgcv)

tree<- rpart(formula = log(price) ~ ., data = rhBM.price[Itr,])

y.hat.te <- as.numeric(predict(tree, newdata = rhBM.price[Ite,], type = "vector"))

plot(log(rhBM.price$price[Ite]),y.hat.te,
    main=paste("Corr^2=",round(cor(log(rhBM.price$price[Ite]),y.hat.te)^2,2)))

### Variable relevance measures
relev.ghost.out <- relev.ghost.var(model=tree,
                        newdata = rhBM.price[Ite,],
                        func.model.ghost.var= lm)

res.var.tree <- mean(residuals(tree)^2)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.ghost.var(relev.ghost.out, n1=n.tr, resid.var=res.var.tree,
                           vars=1:8, sum.lm.tr=NULL,
                           alpha=.01, ncols.plot=4)
```

```
res.var.tree <- mean(residuals(tree)^2)
pdf(file="newVarRlevIdealista_rpart_Gh.pdf", height=16, width=12)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.ghost.var(relev.ghost.out, n1=n.tr, resid.var=res.var.tree,
                     vars=1:8, sum.lm.tr=NULL,
                     alpha=.01, ncols.plot=3)
dev.off()

## Variable relevance matrix by random permutation

relev.rand.out <- relev.rand.perm(model=tree,
                    newdata = rhBM.price[Ite,])

# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.rand.perm(relev.rand.out, relev.ghost=relev.ghost.out$relev.ghost,
                     vars=1:8, sum.lm.tr=NULL,
                     alpha=.01, ncols.plot=4)

pdf(file="newVarRlevIdealista_rpart_RP.pdf", height=16, width=12)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.rand.perm(relev.rand.out, relev.ghost=relev.ghost.out$relev.ghost,
                     vars=1:7, sum.lm.tr=NULL,
                     alpha=.01, ncols.plot=3)
dev.off()
```

RandomForest

```
load(file="rhBM_Price.Rdata")
# rhBM.price, rhBM.priceByArea,
names(rhBM.price)
log.size <- TRUE
if (log.size){
  rhBM.price$size <- log(rhBM.price$size)
  names(rhBM.price)[11]<-"log.size"

names(rhBM.priceByArea)

### Training and test sets
n <- dim(rhBM.price)[1]
pr.tr <- .7
pr.te <- 1 - pr.tr
n.tr <- round(n*pr.tr)
set.seed(123456)
Itr <- sample(1:n,n.tr)
Ite <- setdiff(1:n,Itr)
```

```
n.te <- n-n.tr

library(mgcv)
library(MASS)
library(randomForest)
library(Metrics)
library(caret)
library(dplyr)
library(nlme)
library(lattice)

m <- randomForest(
  formula = log(price) ~ .,
  data    = rhBM.price[Itr,],
  ntree = 1500

plot(m)
nt = which.min(m$mse)  # Number of trees with lowest MSE
regr <- randomForest(x = rhBM.price[Itr,][-1], y = rhBM.price$price[Itr] , ntree = nt)
regr

# Predicting in the test sample
y.hat.te <- as.numeric(predict(regr, newdata = rhBM.price[Ite,], type = "response"))

plot(log(rhBM.price$price[Ite]),y.hat.te,
    main=paste("Corr^2=",round(cor(log(rhBM.price$price[Ite]),y.hat.te)^2,2)))

### Variable relevance measures
library(mgcv)
library(ggplot2)
library(grid)
library(mgcv)
library(maptools)# For pointLabel
library(sp)

source("D:/TFG/Scripts inicials/relev.ghost.var.R")
source("D:/TFG/Scripts inicials/relev.rand.perm.R")

## Variable relevance matrix by ghost variables
relev.ghost.out <- relev.ghost.var(model=regr,
                      newdata = rhBM.price[Ite,],
                      func.model.ghost.var= lm)

res.var.regr <- mean((regr$predicted - rhBM.price$price[Itr]) ^2)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.ghost.var(relev.ghost.out, n1=n.tr, resid.var=res.var.regr,
                         vars=1:7, sum.lm.tr=NULL,
```

```
                        alpha=.01, ncols.plot=4)

res.var.regr <- mean((regr$predicted - rhBM.price$price[ltr]) ^2)
pdf(file="newVarRlevIdealista_RandomForest_Gh.pdf", height=16, width=12)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.ghost.var(relev.ghost.out, n1=n.tr, resid.var=res.var.regr,
                        vars=1:7, sum.lm.tr=NULL,
                        alpha=.01, ncols.plot=3)
dev.off()

## Variable relevance matrix by random permutation
relev.rand.out <- relev.rand.perm(model=regr,
                      newdata = rhBM.price[lte,])

# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.rand.perm(relev.rand.out, relev.ghost=relev.ghost.out$relev.ghost,
                        vars=1:7, sum.lm.tr=NULL,
                        alpha=.01, ncols.plot=4)

pdf(file="newVarRlevIdealista_RandomForest_RP.pdf", height=16, width=12)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.rand.perm(relev.rand.out, relev.ghost=relev.ghost.out$relev.ghost,
                        vars=1:7, sum.lm.tr=NULL,
                        alpha=.01, ncols.plot=3)
dev.off()
```

Glmnet

```
load(file="rhBM_Price.Rdata")
# rhBM.price, rhBM.priceByArea,
names(rhBM.price)
log.size <- TRUE
if (log.size){
  rhBM.price$size <- log(rhBM.price$size)
  names(rhBM.price)[11]<-"log.size"
}

names(rhBM.priceByArea)

### Training and test sets
n <- dim(rhBM.price)[1]
pr.tr <- .7
pr.te <- 1 - pr.tr
n.tr <- round(n*pr.tr)
set.seed(123456)
ltr <- sample(1:n,n.tr)
```

```
Ite <- setdiff(1:n,Itr)
n.te <- n-n.tr

library(mgcv)
library(MASS)
library(Metrics)
library(caret)
library(dplyr)
library('glmnet')

fit.glmnet.lasso <- glmnet(as.matrix(rhBM.price[, -1]),
                           as.matrix(rhBM.price[, 1]),
                           alpha = 1)
plot(fit.glmnet.lasso)

it.glmnet.ridge <- glmnet(as.matrix(rhBM.price[, -1]),
                          as.matrix(rhBM.price[, 1]),
                          alpha = 0)
plot(fit.glmnet.ridge)

_vars <- model.matrix(price~. , rhBM.price)[,-1]
y_var <- rhBM.price$price
x_test = (-Itr)
y_test = y_var[x_test]

cv_output <- cv.glmnet(x_vars[Itr,], y_var[Itr],
                       alpha = 1, nfolds = 5)
plot(cv_output)

#Identifying best lamda
best_lam_lasso <- cv_output$lambda.min

lam.1se_lasso <- cv_output$lambda.1se
lam.1se_lasso

lasso_best <- glmnet(x_vars[Itr,], y_var[Itr], alpha = 1, lambda = best_lam_lasso)
lasso_1se <- glmnet(x_vars[Itr,], y_var[Itr], alpha = 1, lambda = lam.1se_lasso)

# Predicting in the test sample
y.hat.te <- as.numeric( predict(lasso_1se , s = lam.1se_lasso, newx = x_vars[x_test,]) )

plot(log(rhBM.price$price[Ite]),y.hat.te,
    main=paste("Corr^2=",round(cor(log(rhBM.price$price[Ite]),y.hat.te)^2,2)))

x_vars <- model.matrix(price~. , rhBM.price)[,-1]
y_var <- rhBM.price$price
x_test = (-Itr)
```

```
y_test = y_var[x_test]

cv_output <- cv.glmnet(x_vars[ltr,], y_var[ltr],
                       alpha = 0, nfolds = 5)
plot(cv_output)

#Identifying best lamda
best_lam_ridge <- cv_output$lambda.min
lam.1se_ridge <- cv_output$lambda.1se
lam.1se_ridge

ridge_best <- glmnet(x_vars[ltr,], y_var[ltr], alpha = 0, lambda = best_lam_ridge)
ridge_1se <- glmnet(x_vars[ltr,], y_var[ltr], alpha = 0, lambda = lam.1se_ridge)

# Predicting in the test sample
y.hat.te <- as.numeric( predict(ridge_1se, s = lam.1se_ridge, newx = x_vars[x_test,]) )

plot(log(rhBM.price$price[lte]),y.hat.te,
    main=paste("Corr^2=",round(cor(log(rhBM.price$price[lte]),y.hat.te)^2,2)))

### Variable relevance measures
library(ggplot2)
library(grid)
library(maptools)# For pointLabel

source("C:/Users/Oriol/Desktop/TFG/Scripts inicials/relev.ghost.var.R")
source("C:/Users/Oriol/Desktop/TFG/Scripts inicials/relev.rand.perm.R")

relev.ghost.out <- relev.ghost.var(model = lasso_1se,
                        newdata = rhBM.price[lte,],
                        func.model.ghost.var= lm)

# Plotting only eigenvectors with more than 1% of total relevance
y.hat = predict(lasso_1se, newx = x_vars[ltr,])
sigma2.res = mean((y.hat - y_var[ltr])^2)
plot.relev.ghost.var(relev.ghost.out, n1=n.tr, resid.var=sigma2.res,
                           vars=1:10, sum.lm.tr=NULL,
                           alpha=.01, ncols.plot=4)

y.hat = predict(lasso_1se, newx = x_vars[ltr,])
sigma2.res = mean((y.hat - y_var[ltr])^2)
pdf(file="newVarRlevIdealista_glmnet_Lasso_Gh.pdf", height=16, width=12)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.ghost.var(relev.ghost.out, n1=n.tr, resid.var=sigma2.res,
                           vars=1:9, sum.lm.tr=NULL,
                           alpha=.01, ncols.plot=3)
dev.off()
```

```
relev.ghost.out <- relev.ghost.var(model = ridge_1se,
                        newdata = rhBM.price[Ite,],
                        func.model.ghost.var= lm)

# Plotting only eigenvectors with more than 1% of total relevance
y.hat = predict(ridge_1se, newx = x_vars[Itr,])
sigma2.res = mean((y.hat - y_var[Itr])^2)
plot.relev.ghost.var(relev.ghost.out, n1=n.tr, resid.var=sigma2.res,
                        vars=1:9, sum.lm.tr=NULL,
                        alpha=.01, ncols.plot=4)

y.hat = predict(ridge_1se, newx = x_vars[Itr,])
sigma2.res = mean((y.hat - y_var[Itr])^2)
pdf(file="newVarRlevIdealista_glmnet_Ridge_Gh.pdf", height=16, width=12)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.ghost.var(relev.ghost.out, n1=n.tr, resid.var=sigma2.res,
                        vars=1:9, sum.lm.tr=NULL,
                        alpha=.01, ncols.plot=3)
dev.off()

## Variable relevance matrix by random permutation
pdf(file="newVarRlevIdealista_glmnet_Lasso_RP.pdf", height=16, width=12)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.rand.perm(relev.rand.out, relev.ghost=relev.ghost.out$relev.ghost,
                        vars=1:2, sum.lm.tr=NULL,
                        alpha=.01, ncols.plot=3)
dev.off()

relev.rand.out <- relev.rand.perm(model=ridge_1se,
                       newdata = rhBM.price[Ite,])

# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.rand.perm(relev.rand.out, relev.ghost=relev.ghost.out$relev.ghost,
                        vars=1:10, sum.lm.tr=NULL,
                        alpha=.01, ncols.plot=4)

pdf(file="newVarRlevIdealista_glmnet_Ridge_RP.pdf", height=16, width=12)
# Plotting only eigenvectors with more than 1% of total relevance
plot.relev.rand.perm(relev.rand.out, relev.ghost=relev.ghost.out$relev.ghost,
                        vars=1:8, sum.lm.tr=NULL,
                        alpha=.01, ncols.plot=3)
dev.off()
```