Tutor/s

Dr. Esther Chamarro Aguilera
Departament d'Enginyeria Química i Química Analítica

Javier Blanco Guillermo
Director d'Operacions
Cromogenia Units S.A.

Grau en Enginyeria Química

Treball Final de Grau

Demand forecasting using the Monte Carlo method

Àlex Vallespí Monclús

January 2021

UNIVERSITAT DE BARCELONA

"Es mucho mejor prever, incluso sin certeza, que no prever en absoluto."

Henri Poincaré

Antes de entrar de lleno en el trabajo, me gustaría dedicar unas palabras de agradecimiento a todas aquellas personas que han estado a mi lado durante los 4 meses que he estado haciendo el trabajo.

En primer lugar, quiero agradecer enormemente a la empresa Cromogenia Units por brindarme la oportunidad de realizar el trabajo con ellos. Quiero hacer mención especial a tres personas que han sido pilares fundamentales durante la realización del proyecto;

Javier Blanco, Director de Operaciones de la compañía y tutor de las prácticas, ha sido la persona que me ha guiado, ayudado y enseñado durante todo este tiempo, sin ti este trabajo no hubiera sido posible. Gracias.

Vicenç Vidal, Técnico de Producción de la compañía, fue un apoyo en resolverme todas las dudas y aconsejarme siempre, tu ayuda ha sido fundamental. Gracias.

Ventura Granados, Técnico de Planificación de la compañía, fue la persona encargada de proporcionarme los datos de la compañía además de resolverme todas las dudas que me surgían durante la realización del trabajo. Gracias.

En segundo lugar, a Esther Chamarro, tutora del trabajo, por tu ayuda, dedicación y aportaciones que me ayudaron en la realización del proyecto.

Por último, gracias a la Universidad de Barcelona, a todos los profesores y compañeros, des del primero al último, por todos estos años.

CONTENTS

SUMMARY

With the constant market's change and the competition that exists, having a robust forecasting model becomes essential for the companies to be able to compete against other companies and apply it to the Planification and Purchasing Departments to satisfy the customers trying to anticipate what will happen in the market.

In this project, a demand forecasting model has been developed using the ARIMA method. The model input is the company's historical sales data and future forecasts are obtained for the selected period.

After the demand forecasting, the Monte Carlo simulation has been designed and applied due to obtaining the predictions for products' containers, products' quantity, etc. This has been done using the company's previous years' frequencies.

At the same time, three different programs have been developed; an algorithm to find relationships between orders, a program to segment and classify customers, and, finally, an algorithm to group by similarity, in the project's case, customers, but it can be applied to other aspects.

To verify the ARIMA model and reduce the pandemic's effect, three different scenarios have been planned, and their errors have been calculated for each of them. The results obtained are very promising giving low errors meaning that the model is adequate and can be applied in the company to do little predictions.

Monte Carlo's simulation results are not very favorable due to the high errors obtained. In future projects, the company will study the possibility to add more parameters to the algorithm, such as neural networks (artificial intelligence) to improve the project and apply it.

Keywords: Demand forecasting, ARIMA, Monte Carlo.

RESUMEN

Con el constante cambio de mercado y la competencia que existe, es imprescindible que las empresas tengan un modelo de previsión robusto para poder competir y aplicarlo dentro de los Departamentos de Planificación y Compras de cara a satisfacer a los clientes intentando anticiparse al mercado.

En este proyecto se ha desarrollado un modelo de previsión de la demanda utilizando el método ARIMA. El método se alimenta de datos históricos de la compañía obteniendo las previsiones futuras para el periodo de tiempo seleccionado.

Tras la previsión de la demanda, se ha diseñado y aplicado la simulación de Monte Carlo para obtener una previsión del envasado de los productos, cantidad de los pedidos, etc. Todo ello se obtiene utilizando las frecuencias calculadas de la compañía de los años anteriores.

Paralelamente, se han desarrollado tres programas distintos; un algoritmo para encontrar relaciones entre los pedidos, un programa para segmentar y clasificar a los clientes y, finalmente, un algoritmo para agrupar por similitud, en el caso del proyecto, clientes, pero se puede aplicar en otros aspectos.

Se han planteado tres escenarios distintos para intentar reducir el efecto de la pandemia y se han calculado los errores para cada escenario. Se han obtenido resultados muy prometedores con errores bajos por lo que el modelo es adecuado y se podría incorporar a la compañía para hacer pequeñas previsiones.

Los resultados de la simulación de Monte Carlo no son demasiado favorables ya que se obtienen errores elevados, en próximos proyectos, la empresa estudiará la posibilidad de añadir más parámetros al algoritmo, por ejemplo, las redes neuronales (inteligencia artificial) para mejorar el proyecto y aplicarlo.

Palabras clave: Previsión de la demanda, ARIMA, Monte Carlo.

1. INTRODUCTION

1.1. DEMAND FORECASTING

Nowadays, constant market change causes so much uncertainty. Uncertainty appears in all business and consists of the impossibility of knowing how the situation will be. Many companies should incorporate demand forecasting systems in their planification and purchase departments to try to predict how the market will change.

Demand forecasting takes part in different areas: financial, production, personal, and commercial, each one with a specific action plan.

1.1.1. Methods

There are two different demand forecasting methods, qualitative and quantitative (Corres et al., 2009).

The qualitative method is used when there are not enough historical data about the studied phenomenon. This method uses surveys and opinion of experts instead of data, so this kind of forecasting method is subjective.

The quantitative method is used when there is a big amount of historical data, this forecasting method is objective because it is based on real data.

1.1.2. Factors

To carry out a demand forecast it is necessary to keep some factors in mind (Chambers et al., 1971):

1. Type of product, his historical data, and his lifecycle.
2. Competition level, when it is high, the demand forecasting is more difficult.
3. Product price, its variations could affect the demand forecasting indirectly.

4. Product technology level, with technological products, an advance in technology could affect product demand.

5. Economy has a very important role because it is directly related to demand forecasting.

1.1.3. Forecasting problems

There are different methods to predict and forecast the number of sales, but, most of them fail due to some factors, such as the bad use of statistic tools.

Many times, the result is disappointing because the sales only reflect the 40% of the total data provided in the demand forecasting, so the most common error by the business owners is the product innovation where the products don't have any historical data.

Another common error is the lack of accuracy of target customers, product applications, and possible competitors.

The business owners' subjective opinions are another negative factor that could affect the forecasting because they could overvalue a product and then the expectations are not exceeded.

Finally, the costumers' priorities based on surveys can also be biased. It is demonstrated by the results obtained, that shows a different sales behaviour in front of surveys result. That means that the customers' opinion in the survey is not true, so they indicate that they would buy the product, but in fact, they don't do it (Chambers et al., 1971).

1.1.4. Products Lifecycle

The product lifecycle is an important factor and it must be considered in the demand forecasting because depending on the phase that the product is it will be easier to do a demand forecasting (Hernando, 2015).

In Figure ***1*** a product's lifecycle phases are represented.

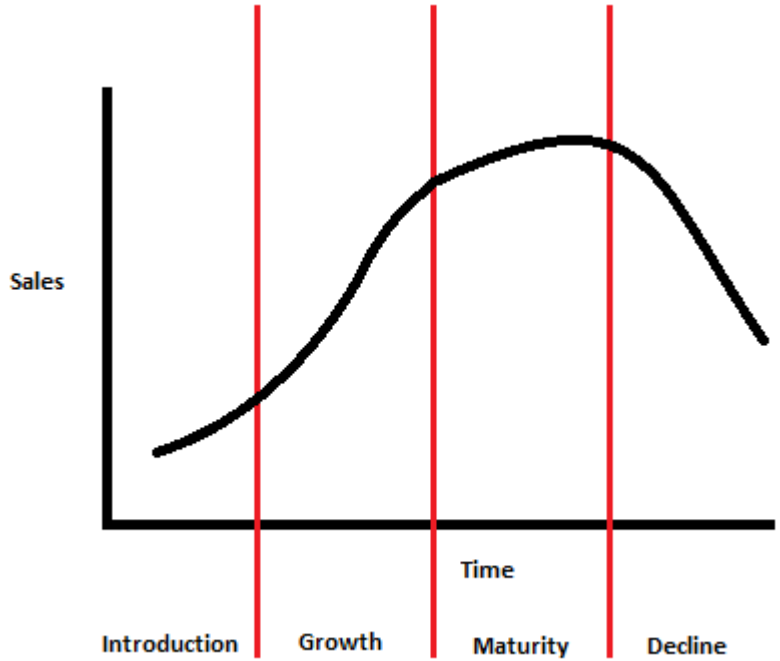


Figure 1 Lifecycle phases

The lifecycle phases are:

1. Introduction, it consists of the product launch, is a phase with too many risk and uncertainty because no one knows how the product will be adapted to the market. The demand forecasting is qualitative because there isn't enough historical data.

2. Growth, the products begin to be accepted by the customers and the sales are increasing. In this phase will appear some competitors so the demand forecasting is difficult.

3. Maturity, this phase takes more time than the other ones. The product reaches its maximum market quote and it's the best phase to do demand forecasting because there is enough historical data and the market is stable.

4. Decline, this is the last phase where the sales begin to decrease until the product is substituted by another one. The demand forecasting will announce the product decline and its possible disappearance.

1.1.5. Forecast Types

The demand forecasting can be of three different kinds (Mar Hernández et al., 2020):

1. Short-term, forecasts until one year, it allows deciding the production, prices, and distribution policy.

2. Medium-term, forecast from 5 to 10 years, it allows deciding about the product introduction, company expansion, and funds request.

3. Long term, 10 or more years of forecasting, it allows deciding the population growth, the economic development, and the policy situation of a country.

1.2. TIME SERIES

1.2.1. Introduction

A time series consists of a combination of variable observations that are taken sequentially in time. Usually, the observations are collected equally spaced.

The value can be anything measurable that depends on time (Shumway & Stoffer, 2013).

1.2.2. Types

The time series could be continuous or discrete (Rey Graña & Ramil Diaz, 2011):

- Continuous, when values are offered permanently, where each one represents the variable condition at every instant.

- Discrete, when values are offered for time slots, where are generally uniform and represent the cumulative state magnitude of the variable during that interval.

1.2.3. Components

Time series has three components and they must be studied in demand forecasting (Rey Graña & Ramil Diaz, 2011):

1. Trend, it is the long-term change that occurs in relation with the mean of the observations.
2. Seasonal component, when a time series has monthly, weekly, quarter influence. Always is a well-known and fixed period.
3. Cyclical or random component, it is an unpredictable fluctuation that occurs randomly in different time instants.

The demand's components are shown in Figure ***2*** :

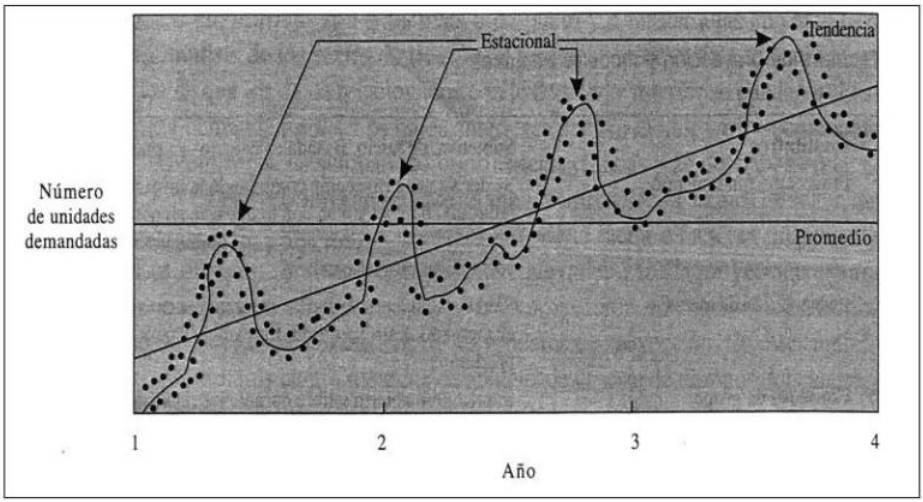


Figure 2 Demand's components (Corres et al., 2009)

The original Figure ***2*** is in Spanish and it remains as the original.

The trend and seasonal components are deterministic, and the cyclical component is random. So, it can be expressed by:

$$Xt = Tt + Et + It$$

Equation 1

Where Tt is the trend, Et the seasonal component and It is the cyclical component.

It is necessary to isolate the cyclical component and study which model will be the most adequate. Once the model will be well-known, the time series behaviour will be able to be well-known.

The cyclical component isolation could be done in two ways:

1. Descriptive perspective, Tt y Et are estimated and It is obtained.

$$It = Xt - Tt - Et$$

Equation 2

2. Box-Jenkins' perspective, the trend and seasonal component are eliminated (with transformations and filters) and only remain the cyclical component which is adjusted by parametrical models.

1.2.4. Classification

Time series can be classified in (Rey Graña & Ramil Diaz, 2011):

- Stationary series, when mean and variance are constants in time and don't have a trend.
- Non-stationary series, when mean or variance change in time. These changes in mean produce a trend to increase or decrease in long term.

1.2.5. Smoothing

The smoothing is a very useful method to eliminate irregularities and fluctuations and obtain a clear, without seasonal variations and ready for the forecast time series. The most common smoothing types are (Carrión García, 2017):

1. Moving averages, this method can be useful to calculate the time series trend without fitting to a previous function obtaining a more adjusted series. There are two types of moving averages:
    1.1. Centred moving averages, the number of observations is odd, so the central observation is the moving average. A centred moving average in t with 2n+1 length is:

$$MM(2n+1)_t = \frac{Y_{t-n} + Y_{t-n+1} + \cdots + Y_t + \cdots + Y_{t+n-1} + Y_{t+n}}{2n+1}$$

Equation 3

1.2. An asymmetric moving average (non-cantered), each moving average is assigned to the period corresponding to the most advanced observation of all observations that are involved in the calculation. The asymmetric moving average for n points and associated with the observation t is:

$$\mathrm{MMA(n)_t} = \frac{\mathrm{Y_{t-n+1}} + \mathrm{Y_{t-n+2}} + \cdots + \mathrm{Y_{t-1}} + \mathrm{Y_t}}{n}$$

Equation 4

2. Simple exponential smoothing, this method smooths all the irregularities that the series has, the weight of the observation decreases exponentially as it moves away from the current moment t and it is simply because the variable is smoothed one time. The smoothed variable St is:

$$\mathrm{S_t} = (1-\mathrm{w})\mathrm{Y_t} + (1-\mathrm{w})\mathrm{wY_{t-1}} + (1-\mathrm{w})\mathrm{w^2Y_{t-2}} + (1-\mathrm{w})\mathrm{w^3Y_{t-3}} + \cdots$$

Equation 5

Where w is a parameter that takes values between 0 and 1.

3. Holt-Winters method, this technique uses a set of recursive estimations from the historical data. These estimations use a level constant, α, a trend constant, β, and a multiplicative seasonal constant, γ.

These recursive estimations are based on three equations:

$$\hat{Y}_t = \alpha\left(\hat{Y}_{t-1} - T_{t-1}\right) + (1-\alpha)\frac{Y_t}{F_{t-s}} \quad with\ 0 < \alpha < 1$$

Equation 6

$$T_t = \beta T_{t-1} + (1-\beta)\left(\hat{Y}_t - \hat{Y}_{t-1}\right) \quad with\ 0 < \beta < 1$$

Equation 7

$$F_t = \gamma F_{t-s} + (1-\gamma)\frac{Y_t}{\hat{Y}_t} \quad with\ \ 0 < \gamma < 1$$

Equation 8

Where the s is the period, Yt the series smoothing level, Tt the series smoothing trend and Ft the seasonal smoothing adjusts of series.

1.2.6. Stationarity estimation

To do a time series analysis, it must be determined if the data follow a stationary or non-stationary process. The most appropriate test for doing this is the Dickey-Fuller Test, which is useful to verify if data is stationary or non-stationary.

This test allows to know if there is a significant trend in time series by a hypothesis contrast, the invalid 0 hypothesis tells that the time series is non-stationary, and the valid hypothesis 0 tells that the time series is stationary.

To find out if the time series is stationary or non-stationary, the p-value is used. This value indicates the probability that hypothesis 0 is not valid. If the p-value is very close to 0, it means that the hypothesis 0 isn't probable, and it should be rejected.

1.2.7. Autocorrelation

Represents the correlation between a sequence and itself. Measures the similarity level between several periods ago sequence (lags) and the actual data. Figures ***3*** and ***4*** are represented the graphics (de la Fuente Fernández, n.d.).

1.2.7.1. Autocorrelation function (ACF)

Measures the correlation between two variables separated by k periods.

The ACF function is used to identify the moving averages (MA) parameter in the ARIMA method and is showed in Figure ***3***.

The MA parameter will be defined by the vertical lines that are bigger than the discontinued horizontal blue lines.

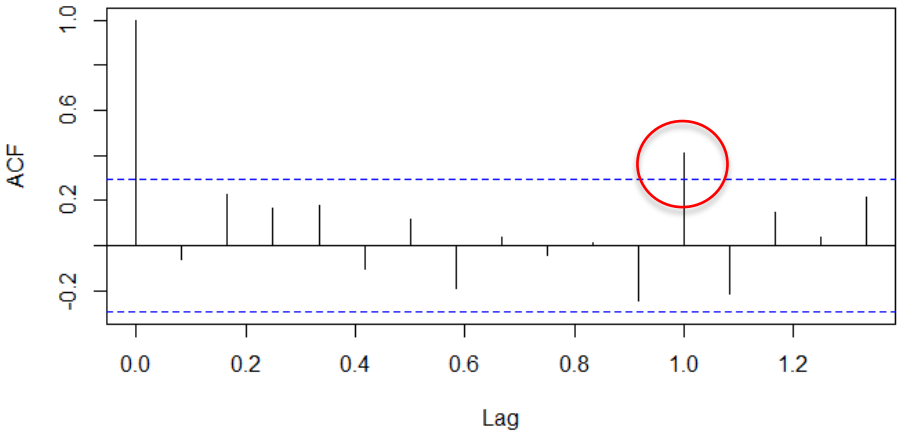


Figure 3 Autocorrelation graphic

1.2.7.2. Partial autocorrelation function (PACF)

Measures the correlation between two variables separated by k periods when the dependency created by the intermediate delays between both isn't considered.

The ACF function is used to identify the autoregressive (AR) parameter in the ARIMA method and is showed in Figure ***4***.

The AR parameter will be defined by the vertical lines that are bigger than the discontinued horizontal blue lines.

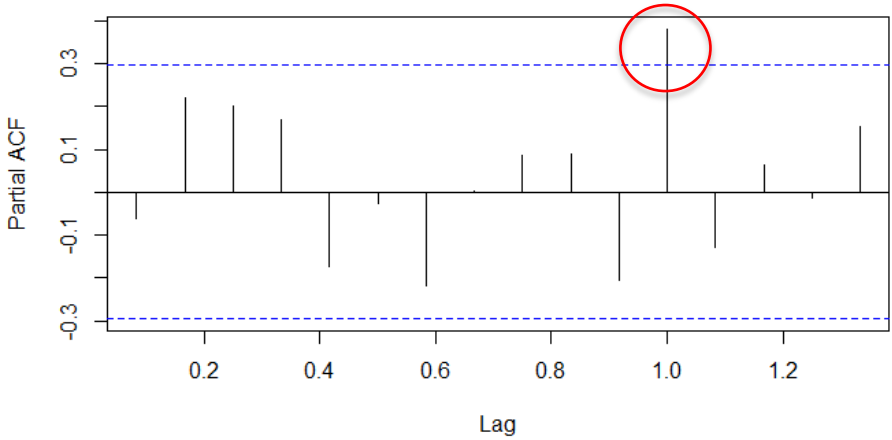


Figure 4 Partial autocorrelation graphic

1.2.8. White noise

It is a special time series where data doesn't follow any pattern. To consider a series as white noise, it must satisfy these conditions:

1. Constant mean
2. Constant variance
3. Not having significant autocorrelations in any period.

1.2.9. Forecasting error calculation

The forecasting errors contain valuable information that should be analysed for two reasons: on the one hand, it allows to detect whether the method predicts accurately, and, on the other hand, all contingency plans should consider the forecast error (Pavolini, 2019).

The forecast accuracy error is the difference between the actual value and the forecast value for a specific period. It is expressed by:

$$E_t = Y_t - T_t$$

Equation 9

Where Yt is the real value and Tt is the forecasted value.

There are different ways to calculate the forecasting error, the most common are:

1. Mean Absolute error (MAE):

$$MAE = \frac{\sum_{t=1}^{N} |E_t|}{N}$$

Equation 10

2. Mean absolute percentage error (MAPE):

$$\mathrm{M}APE = \frac{\sum_{t=1}^{N} \left|\frac{E_t}{Y_t}\right|}{N}$$

Equation 11

3. Root mean square error (RMSE):

$$RMSE = \sqrt{\frac{\sum_{t=1}^{N} E_t^2}{N}}$$

Equation 12

1.3. MONTE CARLO METHOD

The method can be used to solve many complex mathematical problems generating random values (Terejanu, 2013).

1.3.1. Method's history

It was created in 1944 by John Von Newmann and Stanislav Ulam. In the beginning, it was used as an investigation tool during the atomic bomb development in the Second World War.

In 1946, Ulam observed while he was playing Solitaire that it was easier to have an idea about the game's solution if he made multiple trials with cards, analysing all the card combinations and the frequencies of each result. This observation was applied to his neutron's diffusion work because it was impossible to solve the project equations.

Von Newmann wrote a card in 1947 in which he explained the Monte Carlo method and the applications. He suggested that this method could be applied to evaluate complex equations and multiple integrals.

Enrico Fermi, Ulam and Von Newmann applied the method to a deterministic problem in 1948 obtaining estimators to the Schrödinger equation values.

With the computer's development, the method began to increase its precision in the 70's decade (*Monte Carlo, El Origen Del Método*, n.d.).

1.3.2. Applications

Nowadays, the method has different applications, some of the most important are:

- Multiple integral evaluations
- Demand forecast
- Planning
- Risk analysis in projects
- Inventories

1.4. R STUDIO

1.4.1. Program's history

R is a free programming language that comes from S, a previous programming language. This program, S, was developed by John Chamber, Rick Becker, and Allan Wilks in Bell Laboratories in 1976.

In 1988, the S language was rewritten becoming a new language, C, and started to look like the program we know nowadays.

In the early 90s, S was being used less and less due to it was the property of Bell Laboratories. When Ross Ihaka and Robert Gentleman created a new S free variation, it started to increase its use. In 1992, they started to create R, obtaining their first version in 1995 and a more stable version in 2000 (Ihaka, 2009).

1.4.2. Applications

It is a statistic program with many statistical tools. The program allows us to generate high-quality graphics and we can use R as a mathematical tool or mining data.

Also, with R it's possible to define our functions and algorithms. The program has many packages developed by users and accessible to everyone. Moreover, we can connect R to a database that permits to work with a high quantity of data.

1.5. THE COMPANY

Cromogenia Units was created in 1942 and was established in Barcelona.

The headquarters and central laboratories are in Barcelona, there are nine different production plants, two in Barcelona and two in China, one in Tarragona, Murcia, Galicia, Argentina, and Mexico, and three different sales offices in Brazil, Chile, and United States.

Nowadays, the company manufactures many chemical specialties around the world. There are different business areas such as industrial coatings, rubber, leather, adhesives, construction, textile, performance chemicals, water treatment, lubricants, and metalworking and coating for synthetic substrates (*Cromogenia*, n.d.).

Due to confidentiality matters, all the company's data are occulted using a grey square or rectangle.

2. OBJECTIVES

The purpose of this project is to find, choose, and develop a demand's prevision mathematical model using the R Studio program.

The program will use historical sales data to obtain results for future sales. With these future sales results, Monte Carlo's method will be applied to obtain the number of containers and the number of kilograms demanded each product.

These results will help to reduce the stock level, predict future breaks of stock, guaranteeing good quality service and knowledge of the probability of future machinery stops.

Moreover, this project could be used in the planning and purchasing department to buy raw materials, simulate and segment the customers, and know the associations between two or more products in the orders.

3. DEMAND FORECASTING STUDY

3.1. METHODS

Four different forecasting models are proposed for demand forecasting, three of them classic methods and the other one newer. One of these must be chosen to do the project (Otero, 1993).

The four methods are briefly introduced:

- Simple Exponential Smoothing (ETS), it is a classic forecasting method that can be used with data without a stationary pattern or don't have a clear trend (Kourentzes & Petropoulos, 2016).

  The method works considering that the current demand will be the same as the previous demand, but it will apply more weight to the closing values.

- Holt-Winters Method, it is a classic forecasting method, from historical data do the previsions for future values. This method is based on an iterative algorithm that forecasts the behaviour of the series at each time based on the weighted averages of the historical data provided (Carrión García, 2017).

- Trigonometric Exponential Smoothing Method (TBATS), it is a Simple Exponential Smoothing variation, it uses a trigonometric representation based on the Fourier series (Kourentzes & Petropoulos, 2016).

- Autoregressive Integrated Moving Average (ARIMA), it is a statistical model that uses statistical variations and regressions to find patterns for a prediction (de Arce & Mahía, 2001).

3.2. METHOD SELECTION

A demand forecasting for 2019 is done using the historical data from 2014 to 2018, the results are compared with the real values and the errors between the four methods are compared.

The obtained errors are in Table ***1***:

Table 1 Obtained errors from different forecasting methods

METHOD	ERROR (%)
EXPONENTIAL SMOOTHING	4,94
HOLT-WINTERS	4,95
TBATS	5,01
ARIMA	**4,64**

The less error is obtained by the ARIMA method so this will be the method used for the project.

3.3. ARIMA METHOD

A little explanation has been made before but now it will go deeper explaining the components of the model, the procedure, and its programming in RStudio.

In order to obtain estimations with adequate statistic proprieties, the series must be stationary in mean and variance.

The components of the model are (de Arce & Mahía, 2001):

3.3.1. Autoregressive model, AR i MA (p)

The current value is predicted based on past values.

The polynomial is:

$$AR(p) = X_t = \Phi_1 X_{t-1} + \Phi_2 X_{t-2} + \cdots + \Phi_p X_{t-p} + \varepsilon_t$$

Equation 13

Where Xt is the predicted value, ϕ the autoregressive value, ɛt the current period error and Xt-1, Xt-2… the series values in previous periods.

The following examples show the specific correlogram for the autoregressive (AR) parameters.

Examples correlograms, AR(p)

For an autoregressive model with parameter p=1 the correlograms are in Figure ***5***:

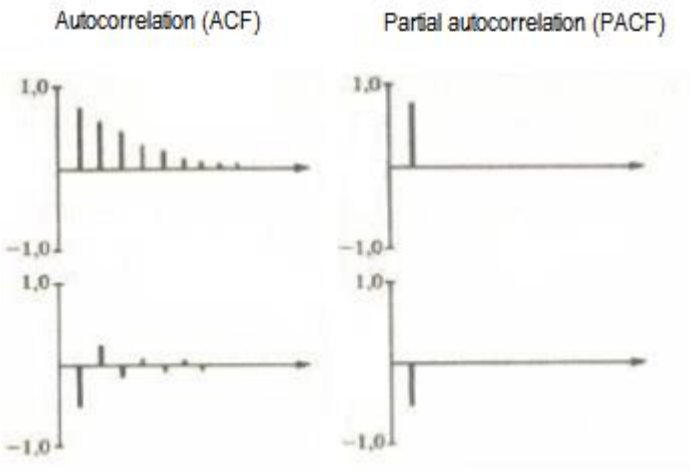


Figure 5 Autocorrelation and partial autocorrelation graphics for an autoregressive model with order 1 (Peña Sánchez & De Rivera, 2005)

For an autoregressive model with parameter p=2 the correlograms are in Figure ***6***:

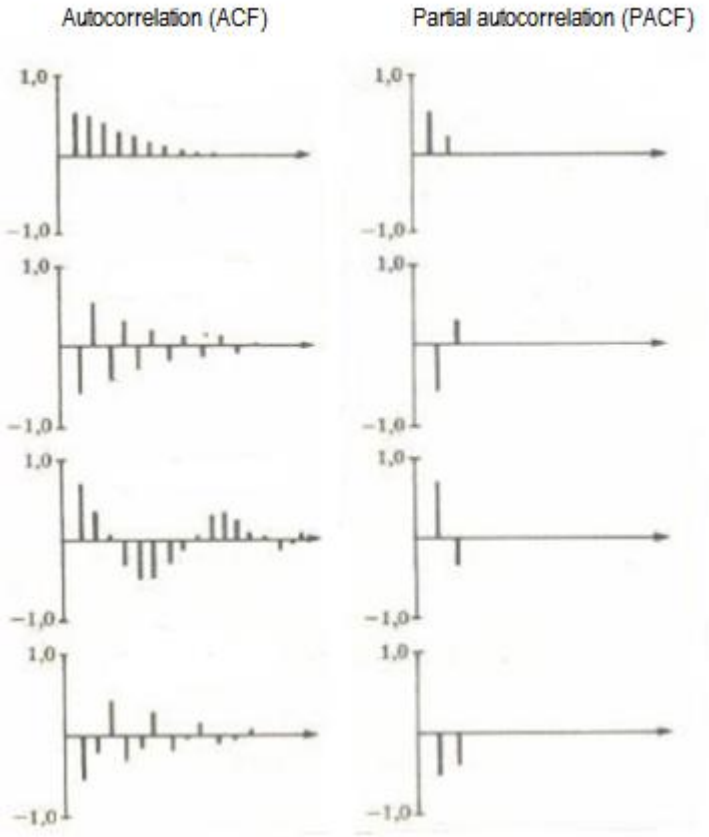


Figure 6 Autocorrelation and partial autocorrelation graphics for an autoregressive model with order 2 (Peña Sánchez & De Rivera, 2005)

3.3.2. Moving average model, AR i MA (q)

The current value is predicted based on the errors of the previous values.

The polynomic for this model is:

$$MA(q) = X_t = \varepsilon_t - v_1\varepsilon_{t-1} - v_2\varepsilon_{t-2} - \cdots - v_p\varepsilon_{t-p}$$

Equation 14

Where Xt is the predicted value, v the moving average value, ɛt the current period error and ɛt-1, ɛt-2… the error values in previous periods.

The following examples show the specific correlogram for the moving averages (MA) parameters.

Examples correlograms, MA (q)

For a moving average model with parameter q=1 the correlograms are in Figure ***7***:

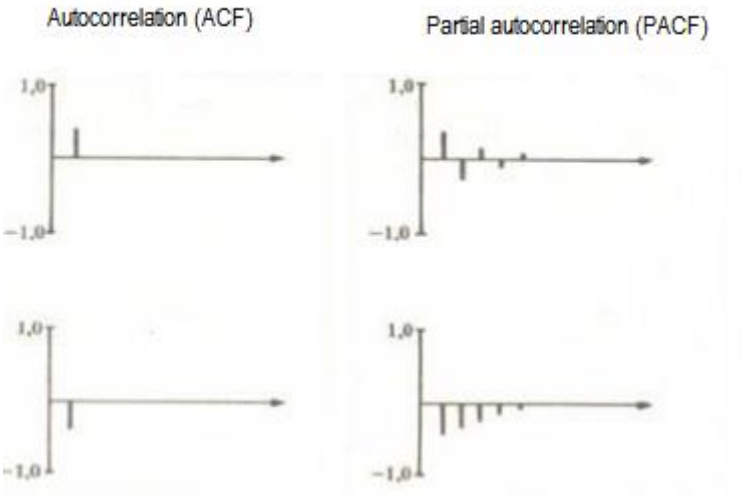


Figure 7 Autocorrelation and partial autocorrelation graphics for a moving averages model with order 1 (Peña Sánchez & De Rivera, 2005)

For a moving average model with parameter q=2 the correlograms are in Figure ***8***:

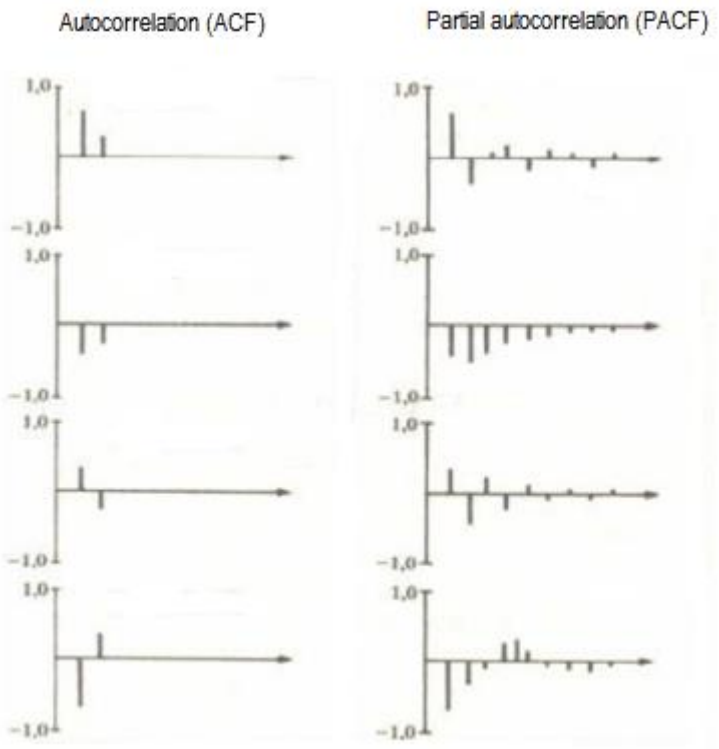


Figure 8 Autocorrelation and partial autocorrelation graphics for a moving averages model with order 2 (Peña Sánchez & De Rivera, 2005)

3.3.3. Autoregressive Moving average model, ARMA(p,q)

The ARMA models can only be applied when the time series don't have a trend.

It is a combination of the two models explained before, the autoregressive and the moving average model. This model has two parameters; p for the autoregressive part and q for the part of the moving average.

The polynomic for this model is:

AR(p) MA(q)

$$X_t = \Phi_1 X_{t-1} + \Phi_2 X_{t-2} + \cdots + \Phi_p X_{t-p} + \varepsilon_t - v_1 \varepsilon_{t-1} - v_2 \varepsilon_{t-2} - \cdots - v_p \varepsilon_{t-p}$$

Equation 15

3.3.4. Autoregressive Integrated Moving average model, ARIMA(p,d,q)

Usually, time series is not stationary, and the series must be transformed with differentiations or logarithmic to make them stationary (de la Fuente Fernández, 2016), (Carrión García, 2017).

In the ARIMA model, the d parameter indicates the times that the series has been transformed. The parameters p and q are for the autoregressive and moving average models.

When differentiation is applied to a time series, the first value of the series gets lost due to differentiations are calculated making the difference between two consecutive values.

The polynomic for this model is:

AR(p) MA(q)

$$X_t^d = \Phi_1 X_{t-1}^d + \Phi_2 X_{t-2}^d + \cdots + \Phi_p X_{t-p}^d + \varepsilon_{t-1}^d - v_1 \varepsilon_t^d - v_2 \varepsilon_{t-2}^d - \cdots - v_p \varepsilon_{t-p}^d$$

Equation 16

Where the X_t^d is the series of the differences of order d.

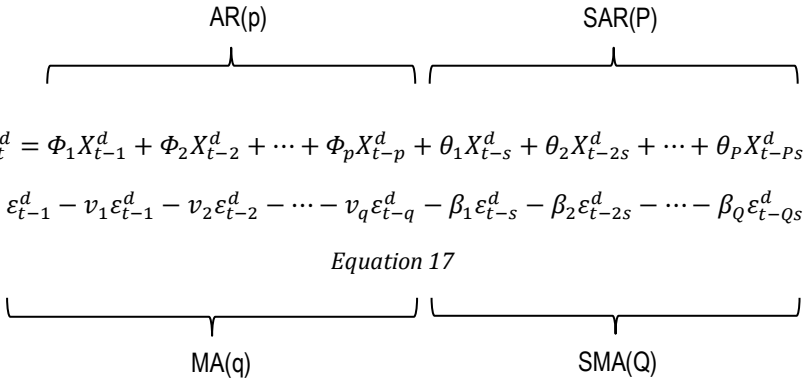
3.3.5. Seasonal Autoregressive Integrated Moving average model, sARIMA(p,d,q)(P,D,Q)[12]

When working with data with a periodicity under a year (quarterly, monthly, or daily), the seasonal factor usually occurs, so correlations between the same months, quarters, etc. of successive years should be analysed (Universidad Nacional Medellin, 2019).

Models with a seasonal component will be obtained when the data series have trend and periodic variations. In this case, the autoregressive model will not predict the data correctly because it only considers the previous period values to estimate the current value.

The SARIMA model has the following parameters; the (p,d,q) parameters for the non-seasonal or regular part and the (P, D, Q) parameters for the seasonal orders related to the errors added accordingly to the seasonal order s.

The polynomic for this model is:



3.3.6. ARIMAx models

The ARIMAX models are an extension of the ARIMA model with one, or more, exogenous variables added.

These exogenous variables could be the price of any product or raw material. This model will show the orders depending on the prices so the results will be more precise and realistic.

Exogenous variables do not depend on the evolution of the model variables and can be added in order to improve the prediction when the model terms don't give an explicative value.

The exogenous series added to the model must be stationary or previously transformed with differentiations or logarithms (Hyndman, 2010).

3.3.7. ARIMA procedure

The procedure to do the ARIMA and obtain the predictions for a determinate period is (de la Fuente Fernández, 2016):

1. Analyse and transform the data; transform the data to time series and look for patterns or irregularities, clean up the atypical and missing values, decompose the time series and observe if exists trend and seasonality.

2. Model adjustment; inspect the seasonality, if the time series needs, apply transformations to convert the time series to stationary.

3. Model evaluation-validation; if the parameters are correct and the remainders don't have significant correlations, the model will be verified, and the prediction can be made.

4. Prediction; when the model is verified, predict for a period.

3.3.8. Programming ARIMA in R

To program the ARIMA model in RStudio, some packages must be installed. These packages include different functions that will be used during the model's programming (Kourentzes & Petropoulos, 2016).

The most used packages are:

- Tseries package: this package is used to analyse the time series.
- Forecast package: this package is used to model the ARIMA and make predictions.
- Greybox package: this package is used to calculate the forecasting errors.

The procedure followed for the programming in R is:

1. Load the data and transform it into time series using the ts function.
2. Make a seasonal decomposition of the time series using the stl function and make a plot with the results, obtained in the graph: trend, seasonality, and a random component. It will be useful to know how to evaluate the time series.
3. Adjust the model and determine the autoregressive (p) and moving average (q) parameters using the autocorrelation and partial autocorrelation charts. Determine the regular

differences with the Dickey-Fuller test using the adf.test function and the stationary differences using the ndiffs function.

If the 0 hypothesis from the Dickey-Fuller is rejected, the model must be transformed by differences and perform the Dickey-Fuller test again to check if the 0 hypothesis is reached or if another differentiation is needed.

4. Once the model and parameters are determined, apply the ARIMA method to get the model by auto.arima function.

5. When the model is determined, analyse the remaining autocorrelations and partial correlation using the Box-Cox test to analyse if the data are independently distributed (present correlation among them with the Box.test function).

If the 0 hypothesis of the Box-Cox test is rejected, make the process and model again.

6. Make the prediction and evaluate the prediction error using the MAPE.

The program can be found in Appendix 1 (Kourentzes & Petropoulos, 2016).

The whole procedure for the ARIMA model is attached in Appendix 2.

The ARIMA procedure in R Studio with its functions is illustrated in figure *9*.

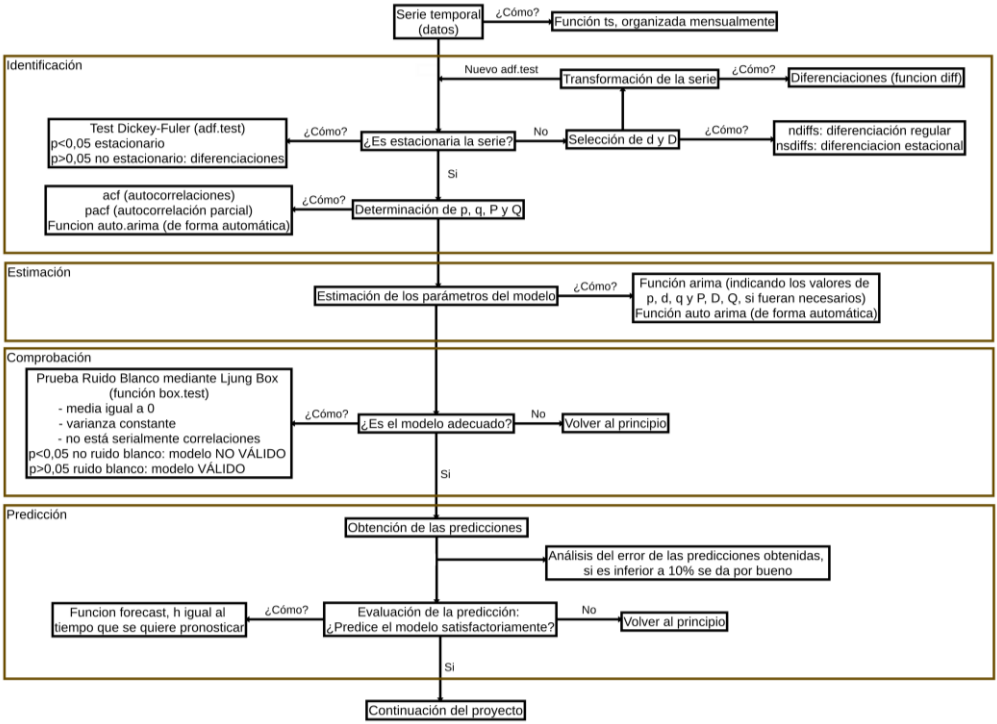


Figure 9 ARIMA procedure in R Studio (de la Fuente Fernández, 2016)

4. TYPE A PRODUCTS

Once the model has been programmed, the most suitable model is determined for each of the company's 50 best-selling products (type A products). The model polynomials are also determined.

The forecasts are also graphed to found which of these products present errors and must be specifically studied.

4.1. ABC ANALYSIS

The ABC Analysis, also called Law 80-20 or Pareto Rule, is a very useful method to accelerate the storage processes of goods in companies (Olivos Aarón & Penagos Vargas, 2013)(*Método Análisis Inventarios ABC*, n.d.).

The Pareto Principle was described by the economist and sociologist Vilfredo Pareto, and it specifies that 80% of earnings are based on 20% of the products.

This method allows identifying the items that have a significant impact on the overall value (inventory, sales, costs...) to focus on them. In general, using this analysis enables the categorization of all the products depending on their importance and profits generated.

An example would be:

1. Type A products; represents 80% of total sales and 20% of total products.
2. Type B products; represents 15% of total sales and 30% of total products.
3. Type C products; represents 5% of total sales and 50% of total products.

Figure ***10*** is shown an ABC analysis and with their percentages of sales and type of product.

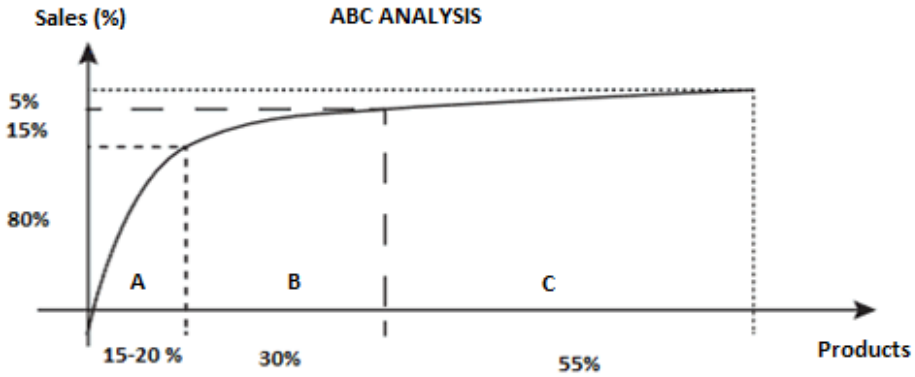


Figure 10 ABC analysis (Olivos Aarón & Penagos Vargas, 2013)

4.2. POLYNOMIALS AND ERRORS

Table ***2***, ***3,*** and ***4*** shows some polynomial, errors, and forecast graphics. The products A and B are correct forecasts with low error and are shown in figures ***11*** and ***12*** (de la Fuente Fernández, 2016).

Product C is a correct forecast with a big error due to an unexpected drop in sales, figure ***13***. These cases must be studied with the purchasing department.

Table 2 Products with their forecasted model, error, polynomial for product A

CODE	MODEL	ERROR (%)	POLYNOMIAL
PRODUCT A	ARIMA(1,1,0)(1,1,0)[12]	3,82	$(1+0.6345*B)(1+0.6998*B^{12})(1-B)(1-B12)*X_t=at$

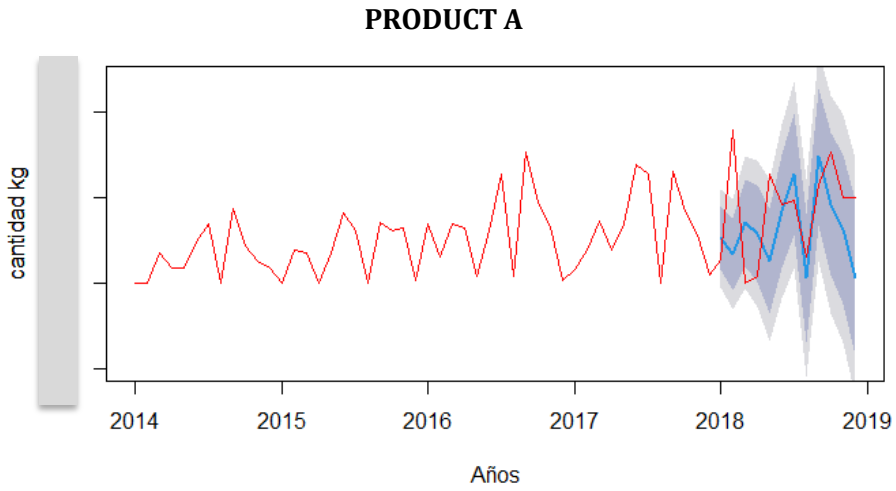


Figure 11 Product A forecast

This model is shown an autoregressive component of order one in the regular part of the model. This means that the future values will be determined by the previous value, (1 month ago).

The temporal series presents a trend, that is why differentiation is applied to convert the series into stationary.

The stationary part of the model represents an autoregressive component of order one. This means that the future values will be determined by the values of the previous period, (12 months ago).

Table 3 Products with their forecasted model, error, polynomial for product B

CODE	MODEL	ERROR (%)	POLYNOMIAL
PRODUCT B	ARIMA(2,1,0)(1,1,0)[12]	4,58	$(1+0.4364*B+0.8752*B^2)(1+0.4458*B^{12})$ $(1-B)(1-B^{12})*X_t=a_t$

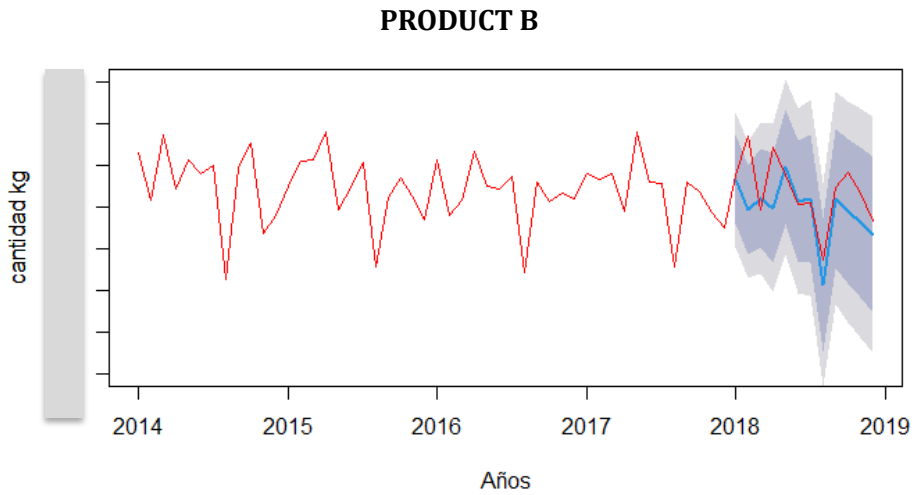


Figure 12 Product B forecast

In this model an autoregressive component of order two is displayed, therefore the future values will be determined by the two previous values (2 months ago).

A differentiation must be applied to the series because it is not stationary, and it must be transformed.

The stationary part of the model has an autoregressive order one component, so the future values will be determined by the value of the previous period (12 months ago).

Table 4 Products with their forecasted model, error, polynomial for product C

CODE	MODEL	ERROR (%)	POLYNOMIAL
PRODUCT C	ARIMA(1,1,0)(1,0,0)[12]	35,2	$(1+0.3523*B)(1-0.4261*B^{12})(1-B)*X_t=a_t$

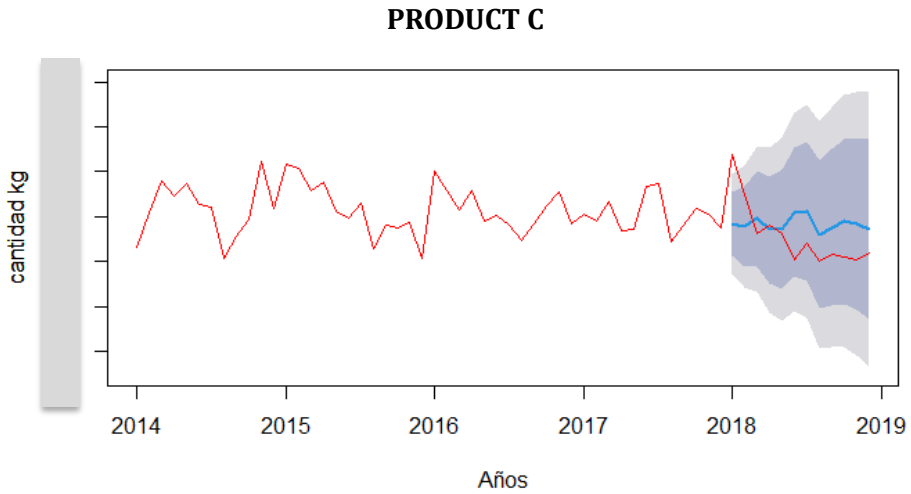


Figure 13 Product C forecast

This model represents an autoregressive component of order one, therefore the future values will be determined by the previous value (1 month ago).

A differentiation must be applied to the series due to it is not stationary and must be transformed.

The stationary part of the model has an autoregressive component of order one. The future values will be determined by the value of the previous period (12 months ago).

4.2.1. Polynomials

The polynomial formation is explained using the following equation as an example:

$$\left(1-\varphi_1 B-\varphi_2 B^2-\ldots-\varphi_p B^p\right)\left(1-\Omega_{12} B^{12}-\Omega_{24} B^{24}-\ldots-\Omega_{P*s} B^{P*s}\right)(1-B)^d(1-B^{12})^D X_t=\left(1-v_1 B-v_2 B^2-\ldots-v_q B^q\right)\left(1-\delta_{12} B^{12}-\delta_{24} B^{24}-\ldots-\delta_{Q*s} B^{Q*s}\right)a_t$$

Equation 18

B values are the time series past values, also called Xt-1, Xt-2, etc.

Xt is the forecasted value.

The at is the white noise or error.

The ϕ and Ω are the autoregressive and seasonal autoregressive values obtained from the model. When a value belongs to the seasonal part, the B is raised to the power of 12 (the seasonal order).

The v and δ are the moving averages and seasonal moving averages values obtained from the model. When a value belongs to the seasonal part, the B is raised to the power of 12 (the seasonal order).

The (1-B) is the differentiations applied to the time series. If the model had a seasonal differentiation in the polynomial is represented as $(1-B^{12})$ (de la Fuente Fernández, 2016).

5. ASSOCIATION RULES ALGORITHMS

Often, companies have very similar or repetitive orders where some products are always present. Some customers have more than one product in the same order, so an algorithm is used to detect these relationships between orders and apply them to the predictions.

These algorithms aim to find relationships within a set of transactions or attributes that tend to occur together.

The word transaction refers to each group of events that are associated in some way, for example, the supermarket shopping basket.

Each event or element involved in a transaction is known as an item and a combination of them is called itemset. A transaction can be made up of one or more items. When transactions have more than one item, each possible subset is a different set of items. For example: the transaction {Blue,Red,Green} is formed by three items and their possible itemsets are: {Blue,Red,Green}, {Blue, Red}, {Blue, Green}, {Red, Green}, {Blue}, {Red} and {Green} (Hernando Ávila-Toscano, 2018).

5.1. ECLAT

The equivalence class transformation algorithm (ECLAT) is used to find patterns or sets of elements that are frequent. This algorithm analyses transactions in vertical format, where each row contains an item, the transactions that this item takes part in, and the minimum support.

The minimum support is the minimum value from which the itemsets will be displayed. Usually, to consider a frequent itemset the value must be higher than the minimum support value, when it is lower than the minimum support, that itemset will be discarded. The minimum support value can be modified depending on the input (Amat Rodrigo, 2019).

To know more about this algorithm, an example is explained below:

Table *5* contains transactions' information in a horizontal format. The minimum support value in this case is 20%.

Table 5 Items per transaction (Amat Rodrigo, 2019)

TRANSACTION	ITEM
1	Orange, Black
2	Blue, Green, Orange, Black
3	Blue, Green, Black
4	Green, Orange
5	Green, Orange
6	Orange, Black
7	Red, Green, Black
8	Red, Black
9	Green, Orange, Black

The algorithm transforms the table into a vertical format, identifies the elements that appear in all the transactions, and calculates the minimum support.

To calculate the minimum support, the transactions in which each item appears are divided by the total number of transactions.

Table *6* are shown the results for itemset = 1:

Table 6 Results for each itemset with 1 item (Amat Rodrigo, 2019)

ITEMSET	TRANSACTION	MINIMUM SUPPORT
BLUE	2, 3	2/9 = 0.22
RED	7, 8	2/9 = 0.22
GREEN	2, 3, 4, 5, 7, 9	6/9 = 0.66
ORANGE	1, 2, 4, 5, 6, 9	6/9 = 0.66
BLACK	1, 2, 3, 6, 7, 8, 9	7/9 = 0.77

All minimum supports are upper than 20%, therefore no itemset are discarded.

The results for itemset = 2 are in Table ***7***:

Table 7 Results for each itemset with 2 items (Amat Rodrigo, 2019)

ITEMSET	TRANSACTION	MINIMUM SUPPORT
BLUE, GREEN	2, 3	2/9 = 0.22
BLUE, ORANGE	2	1/9 = 0.11
BLUE, BLACK	2, 3	2/9 = 0.22
RED, GREEN	7	1/9 = 0.11
RED, BLACK	7, 8	2/9 = 0.22
GREEN, ORANGE	2, 4, 5, 9	4/9 = 0.44
GREEN, BLACK	2, 3, 7, 9	4/9 = 0.44
ORANGE, BLACK	1, 2, 6, 9	4/9 = 0.44

Minimum supports below 20% are discarded and the higher itemsets are the result.

Finally, in Table ***8*** are the results for itemset =3:

Table 8 Results for each itemset with 3 items (Amat Rodrigo, 2019)

ITEMSET	TRANSACTION	MINIMUM SUPPORT
BLUE, GREEN, BLACK	2, 3	2/9 = 0.22
GREEN, ORANGE, BLACK	2, 9	2/9 = 0.22

The algorithm finishes when there aren't more frequent itemsets. And the results for the example are in Table ***9***:

Table 9 Algorithm results (Amat Rodrigo, 2019)

ITEMSET	TRANSACTION	MINIMUM SUPPORT
BLUE, GREEN	2, 3	2/9 = 0.22

BLUE, BLACK	2, 3	2/9 = 0.22
RED, BLACK	7, 8	2/9 = 0.22
GREEN, ORANGE	2, 4, 5, 9	4/9 = 0.44
GREEN, BLACK	2, 3, 7, 9	4/9 = 0.44
ORANGE, BLACK	1, 2, 6, 9	4/9 = 0.44
BLUE, GREEN, BLACK	2, 3	2/9 = 0.22
GREEN, ORANGE, BLACK	2, 9	2/9 = 0.22

5.2. PROGRAMMING ECLAT IN R

The following package must be installed to work with R data (*ML Using R Section 25 Eclat Association Rule Learning*, n.d.):

- Arules package: package that provides the functions to represent, manipulate, and analyse patterns and transaction data.

Once the package is installed, the data must be transformed into a transaction format.

The program can be found in Appendix 3.

The output shows the products which are sold together with the most frequently among the studied data. The support matches the relative frequency for each product.

Figure ***14*** are the results for Algorithm Eclat in R Studio.

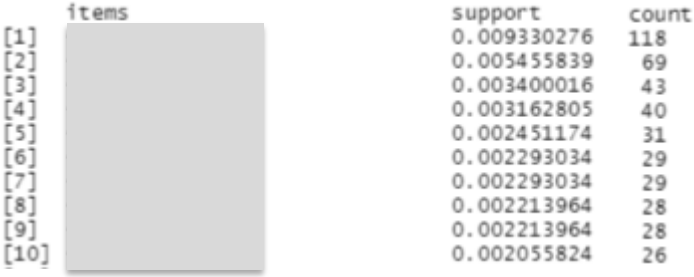


Figure 14 Algorithm Eclat results in R Studio

6. MARKET SEGMENTATION

The company's customers and abnormal orders can also be studied. There are two different types of segmentation.

6.1. CUSTOMER SEGMENTATION

This method is created to make classifications among the company's customers and apply a specific marketing method to each group (J. A. Berry & S. Linoff, 1997).

The technique RFM is used to analyse the customers using three parameters specific for each of them (Hebbali, 2019).

These parameters are:

- Recency: number of days since the last order.
- Frequency: number of times customers have placed orders in a period.
- Monetary: total amount spent by customers on their orders.

Once these parameters are calculated, they can be classified from 1 to 4 according to quartiles (25%, 50%, 75%) and classify customers in (Hebbali, 2019):

- Recommended: they buy frequently and recently and who spend the most money.
- Loyal customers: they spend a lot of money and respond to promotions.
- Loyal potentials: they buy recently, spend a lot of money, and more than once.
- Recent customers: buy recently but infrequently.
- Promise: recent buyers but spend not much money.
- Customers who need attention: buy above average on all three parameters.
- Sleepy: below average on all three parameters.
- At risk: spend money and buy frequently but haven't ordered in a long time.
- Can't lose them: spend a lot of money and buy very frequently but haven't ordered for a long time.

- Hibernating: the last order was a long time ago, with low monetary value and low frequency.
- Lost: low recency, frequency, and monetary value.

After this classification, the purchasing department could make a specific marketing strategy for each customer.

The program can be found in Appendix 4 (Wicaksono, 2019).

The result obtained from the segmentation is the following, where each customer is classified in a different group, filtering by country, among other ways.

Figure ***15*** are the customer segmentation results in Power BI.

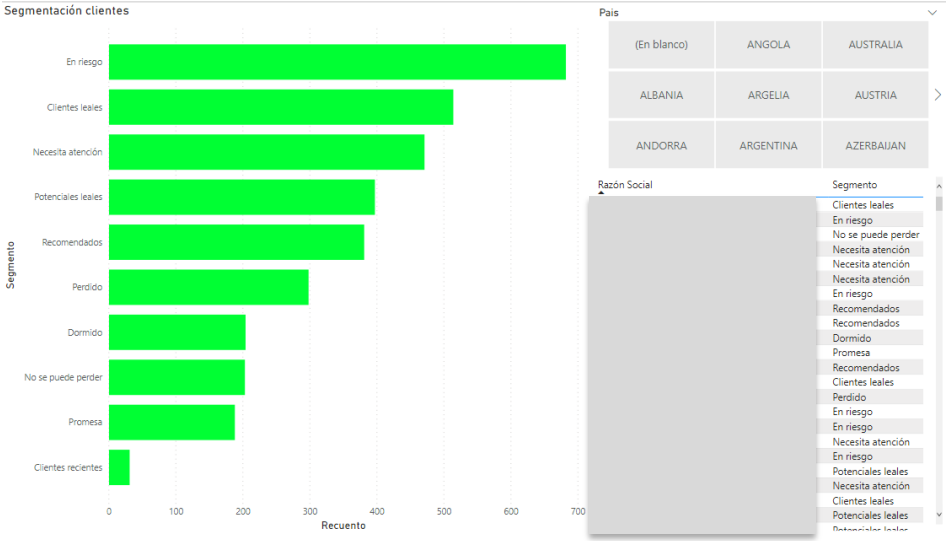


Figure 15 Customers segmentation results in Power BI

6.2. QUANTITY SEGMENTATION

Order quantities are profiled to detect abnormal order quantities. This is done by analysing the orders' quantities placed in a period.

The program can be found in Appendix 5 (Banchero et al., n.d.).

The results show which orders are abnormal. For each order, Figure ***16*** shows their date, customer, purchase number, product code, product name, and quantity.

The boxplot to observe the outliers is shown in Figure ***17***.

	Fecha	Cliente	NoAlb	Codigo	Descripcion	Cantidad
1	2020-03-18		2006402			22000
2	2019-10-31		1924015			22000
3	2020-09-25		2015547			22000
4	2020-02-18		2003890			11000
5	2019-07-29		1917444			11000
6	2019-05-29		1912223			11000
7	2019-04-12		1909162			11000
8	2019-02-12		1903712			11000
9	2020-01-24		2001833			7000
10	2019-09-13		1919612			9600

Figure 16 Quantity segmentation results in R Studio

An outlier value is considered when a value is upper or lower than 1.5 multiplied by interquartile rank.

The interquartile rank calculates the difference between the third quartile and the first quartile where each quartile is calculated from each of the orders' quantity.

The points are the orders' quantity which are outliers, the orders that are not outliers are inside of the grey box.

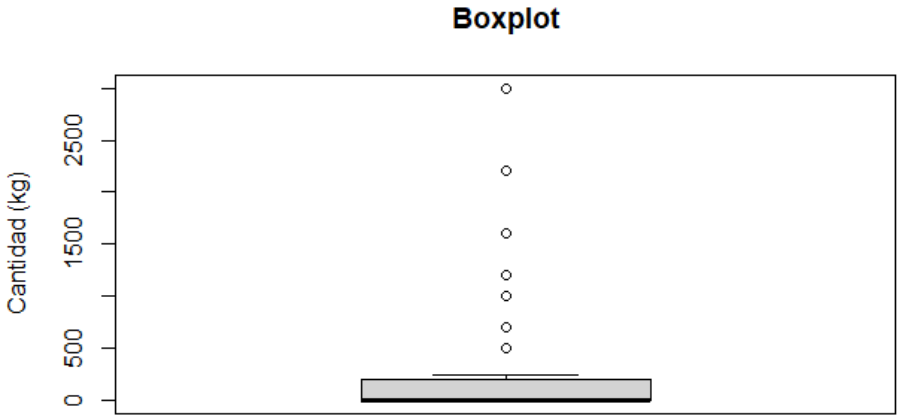


Figure 17 Boxplot for quantity segmentation results

7. CLUSTERING METHOD

This method is a multivariate statistical technique whose objective is to group elements or variables trying to reach the maximum homogeneity in each group and the maximum difference among groups (J. A. Berry & S. Linoff, 1997).

There are two different clustering methods (Villardón, 2011):

- Non-hierarchical method: the cluster number is determined beforehand. Each observation is assigned to a group and maximizing the homogenization inside the groups.
- Hierarchical method: they form groups with tree structure; therefore, the lower level clusters are included in other higher-level clusters. The cluster number is determined after applying the method, using the tree structure.

In this case, the k-means, a hierarchical method, is used in order to form groups by the similarity of the values of recency, frequency, and monetary (Villagómez & Landa, 2010).

The program can be found in Appendix 6 (Delgado, 2018).

In Figure ***18*** are the results obtained in Power BI, where the three parameters are related. In the graphics, each cluster is represented by one colour.

These results in Power BI allow to filter by country and group of segmentation discussed above:

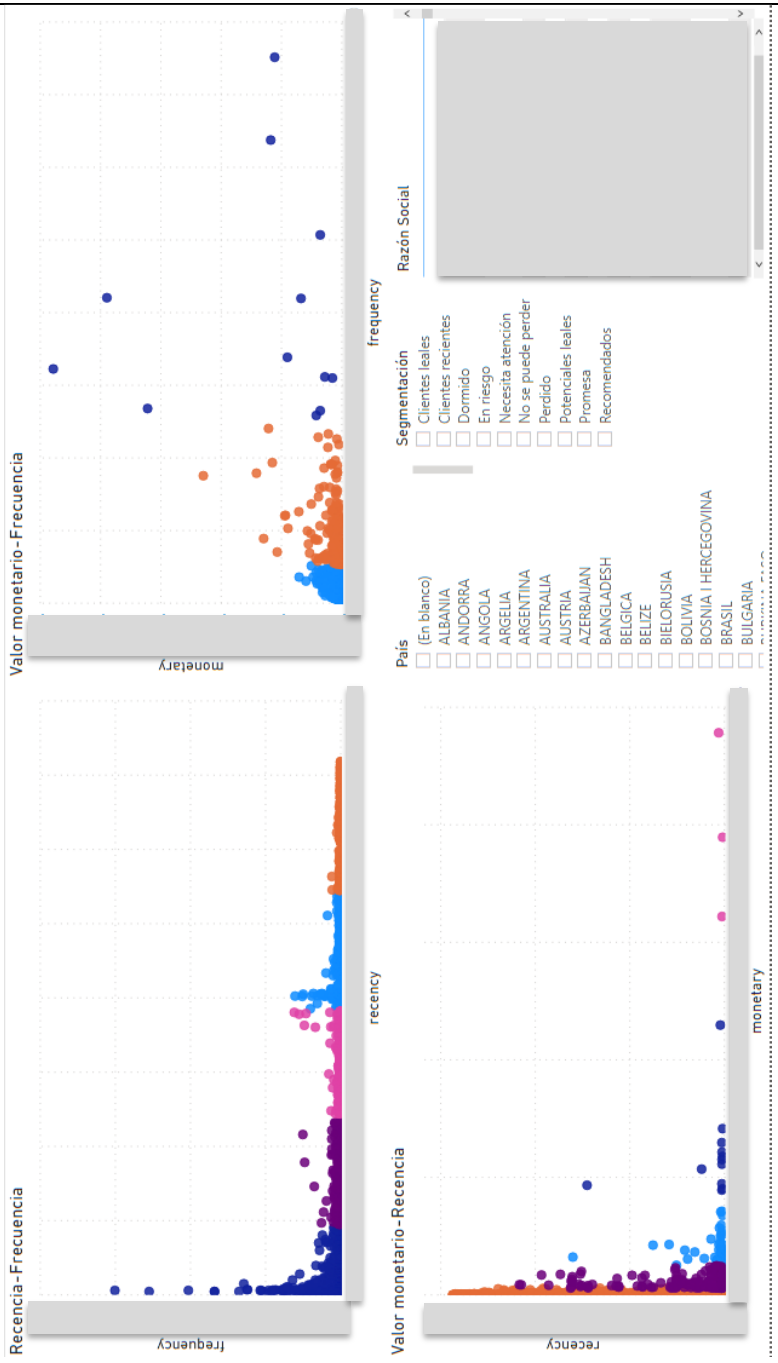


Figure 18 Clustering method results

8. MONTE CARLO SIMULATION

The Monte Carlo Simulation is the next step in forecasting demand. Once the monthly quantities are calculated with the ARIMA method, the Monte Carlo simulation is done to determine which quantity will be obtained and which container will use every product.

The simulation's program has three different stages:

1. The first stage is the frequencies calculation of each product and container. The simulation is done 5 times, each time, it is simulated as many times as the number of orders obtained in the demand forecasting (ARIMA method).

   The simulation's result is 5 columns (one per simulation) with different products, as many as the input.

2. The second stage consists of doing the recount of each simulation and joining them in a table where each product is related to the number of times that appear in the simulation.

3. Finally, the last stage consists of obtaining the minimum and maximum tons, of the 5 simulations, for each product.

The program can be found in Appendix 7.

In Table ***10*** are the results obtained for the Monte Carlo simulation.

Table 10 Monte Carlo simulation results

PRODUCT	**SIMULATED TONS**	
	Minimum	Maximum
PRODUCT A	173	198
PRODUCT B	155	211
PRODUCT C	66	85
PRODUCT D	85	98
PRODUCT E	100	129

9. RESULTS

While the project is being made, it's necessary to confirm if the program gives results according to reality and the forecasting errors are low.

To do this, three different scenarios are planned to calculate the forecasting errors and reduce the pandemic effect.

These scenarios are:

- Scenario 1: forecasts including 2020
- Scenario 2: forecasts non-including 2020
- Scenario 3: forecasts changing April, May, June, and July 2020 to 2019

A forecast was made for September, October, November, and December and the forecast error was calculated at the end of the month and are in Table ***11***:

Table 11 ARIMA errors

	Scenario 1	*Scenario 2*	*Scenario 3*
SEPTEMBER	4.14%	29.90%	15.56%
OCTOBER	29.66%	5.13%	5.98%
NOVEMBER	0.39%	28.17%	21.49%
DECEMBER	10.36%	13.86%	0.77%

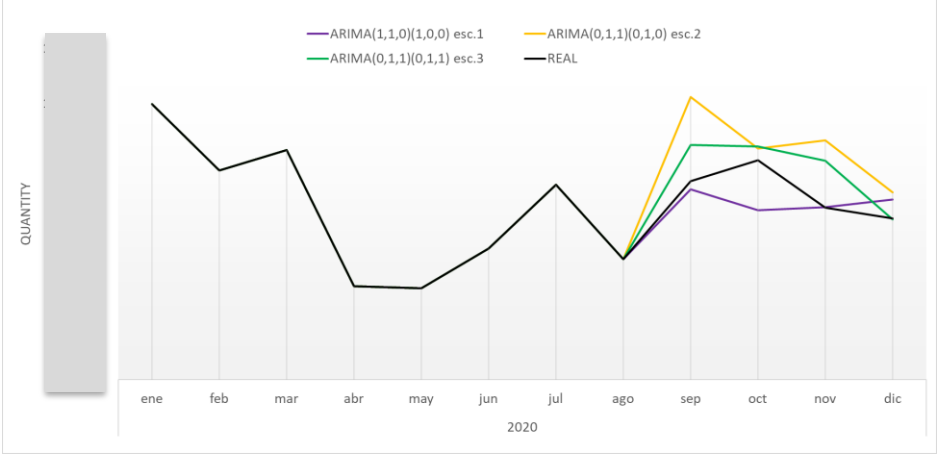


Figure 19 Graphic prediction results for 2020

The Monte Carlo simulation's results are displayed in Power BI along with the ARIMA's method results.

Power BI is a program that allows doing interactive presentations with an intuitive and simple interface for the user.

The output obtained by Power BI depicts the simulated time series from 2017 to 2021 along with the times series' comparison for each year from 2017 to forecast 2021.

Besides, it presents the maximum and minimum tons obtained in the Monte Carlo simulation and the simulation's result can be filtered by section.

Each scenario has its results in different dashboards.

The obtained dashboard for one scenario is in Figure ***20***:

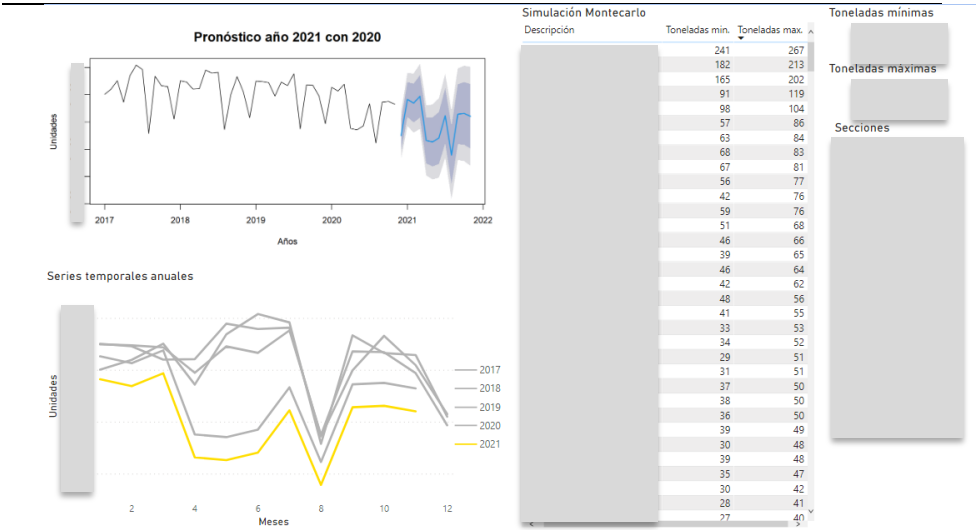


Figure 20 Power BI results

To validate the Monte Carlo method, the simulation is applied to know how many tonnes of each product will be ordered in the future based on the tonnes forecasted for a month by the ARIMA method.

Then, the results are compared with the real data and the MAPE error is calculated.

In Table ***12*** , the results are the average of all errors for each product:

Table 12 Monte Carlo simulation errors

	SEPTEMBER	***OCTOBER***	***NOVEMBER***	***DECEMBER***
SCENARIO 1 ERROR	21.75%	16.78%	11.78%	13.46%
SCENARIO 2 ERROR	19.19%	24.68%	14.21%	12.61%
SCENARIO 3 ERROR	27.73%	17.30%	12.13%	9.45%

10. CONCLUSIONS

The demand forecasting results are good obtaining errors lower than 10% in some of the suggested scenarios That value can be considered as the maximum error to say that the prediction is correct.

The algorithm Eclat can be very useful for the company to find relations between different orders. Then it can be applied to the prediction because it links products that use to be sold in the same order. For example, if prediction gives Product X as a result and the Eclat algorithm results say that Product X and Product Y are usually ordered together, it is probably that Product Y will also be demanded so it makes the prediction stronger.

Customers segmentation can be used to group customers according to their characteristics against the company's sales. With this method, decisions can be made to adapt the marketing strategies depending on which customers' group will be oriented.

The quantity segmentation helps to find orders with unusual quantities. It can also be applied to the predictions to determine which results differ a lot from the normal quantities demanded and predicted.

The clustering method is very useful to obtain relations between the methods explained before (customers and quantity segmentation). This method can also be applied to the forecasting results to group items by country, orders, etc. according to the similarity between the values.

The Monte Carlo simulation has larger errors than the demand forecasting because it is influenced by the previous part errors. In order to reduce these errors, neural networks (artificial intelligence) can be useful to continue the project as a second part and join all the developed methods.

REFERENCES AND NOTES

Amat Rodrigo, J. (2019). *Reglas de asociación y algoritmo Apriori con R*. https://www.cienciadedatos.net/documentos/43_reglas_de_asociacion%0Ahttps://rpubs.com/Joaquin_AR/397172

Banchero, S., Leo, L., & Fernandez, J. M. (n.d.). *RPubs - Detección de Outliers (LAB03)*.

Carrión García, A. (2017). *Análisis de series temporales, técnicas de previsión*. *August 2001*.

Chambers, J. C., Mullick, S. K., & Smith, D. D. (1971). How to choose the right forecasting technique. In *Harvard Business Review* (Vol. 49, Issue 4, pp. 45–70). http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:How+to+choose+the+right+forecasting+technique#0%5Cnhttp://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:HOW+TO+CHOOSE+RIGHT+FORECASTING+TECHNIQUE#0

Corres, G. A., Esteban, A., García, J. C., & Zárate, Cl. (2009). Análisis De Series Temporales Time Series Analysis. *Revista Ingeniería Industrial*, *1*, 21–34.

Cromogenia. (n.d.).

de Arce, R., & Mahía, R. (2001). Introducción a los modelos ARIMA. *Media*, 31. http://www.uam.es/personal_pdi/economicas/eva/ecoii.html

de la Fuente Fernández, S. (n.d.). *ALISADO SERIES TEMPORALES*. *Métodos de alisado*, 1–40.

de la Fuente Fernández, S. (2016). Series Temporales: Modelo Arima. *Universidad Autónoma de Madrid*, 1–14. http://www.estadistica.net/ECONOMETRIA/SERIES-TEMPORALES/modelo-arima.pdf

Delgado, R. (2018). *RPubs - Introducción a los Modelos de Agrupamiento en R*. https://rpubs.com/rdelgado/399475

Hebbali, A. (2019). *RFM - Customer Level Data*. https://cran.r-project.org/web/packages/rfm/vignettes/rfm-customer-level-data.html

Hernando Ávila-Toscano, J. (2018). Minería de datos aplicada al análisis bibliométrico. Descripción y usos de reglas de asociación y modelos de regresión basados en árboles. *Cienciometría y Bibliometría. El Estudio de La Producción Científica Métodos, Enfoques y Aplicaciones En El Estudio de Las Ciencias Sociales.*, 195–220. https://dialnet.unirioja.es/servlet/libro?codigo=722508

Hernando, E. S. (2015). Ciclo De Vida De Producto. Modelos Y Utilidad Para El Marketing. *Anuario Del Centro de La Universidad Nacional de Educación a Distancia En Calatayud*, *21*, 207–227. http://www.calatayud.uned.es/web/actividades/revista-anales/21/03-10-EduardoSanchezHernando.pdf

Hyndman, R. J. (2010). *The ARIMAX model muddle | Rob J Hyndman*. https://robjhyndman.com/hyndsight/arimax/

Ihaka, R. (2009). The R Project: A Brief History and Thoughts About the Future. *Stat.Auckland.Ac.Nz*, 34. https://www.stat.auckland.ac.nz/~ihaka/downloads/Massey.pdf

J. A. Berry, M., & S. Linoff, G. (1997). *Data Mining Techniques* (2nd Editio).

Kourentzes, N., & Petropoulos, F. (2016). *Forecasting with R*. *June*.

Mar Hernández, N. N., Cano-Olivos, P., Sánchez-Partida, D., Martínez-Flores, J.-L., & Caballero-Morales, S. O. (2020). *Demand Forecasting*. 1–21. https://doi.org/10.4018/978-1-7998-1831-1.ch001

Método Análisis inventarios ABC. (n.d.).

ML Using R Section 25 Eclat Association Rule learning. (n.d.).

Monte Carlo , el origen del método. (n.d.). 1947–1948.

Olivos Aarón, S., & Penagos Vargas, J. W. (2013). Modelo de Gestión de Inventarios: Conteo Cíclico por Análisis ABC. *Ingeniare*, *14*, 107. https://doi.org/10.18041/1909-2458/ingeniare.14.617

Otero, J. M. (1993). *José María Otero Moreno - Econometría_ Series temporales y predicción-Alfa Centauro (1993).pdf*.

Pavolini, G. (2019). *Estimando Errores de Pronóstico - Giovanni Pavolini's Random Ideas*.

Peña Sánchez, D., & De Rivera. (2005). *Análisis de series temporales*.

Rey Graña, C., & Ramil Diaz, M. (2011). Series temporales. *Introduccion a La Estadistica Descriptiva. Segunda Edicion*, 85–105. https://doi.org/10.4272/978-84-9745-167-3.ch4

Shumway, R., & Stoffer, D. (2013). Time Series Analysis and its applications with R. In *Journal of Chemical Information and Modeling* (Vol. 53, Issue 9). https://doi.org/10.1017/CBO9781107415324.004

Terejanu, G. a. (2013). Tutorial on Monte Carlo Techniques. *Tutorials*, 1–15. http://www.cse.sc.edu/~terejanu/files/tutorialMC.pdf

Universidad nacional medellin. (2019). 8.1.1. Modelo Multiplicativo SARIMA. *Notas de Clase*, 127–142.

Villagómez, D. C., & Landa, M. del P. (2010). *Aplicacion del metodo cluster en R*. 1–26.

Villardón, J. L. V. (2011). *Introduccion Al Analisis De Cluster*. 22. http://benjamindespensa.tripod.com/spss/AC.pdf

Wicaksono, T. I. (2019). *Customer Segmentation using RFM Analysis (R) | Kaggle*. https://www.kaggle.com/hendraherviawan/customer-segmentation-using-rfm-analysis-r

ACRONYMS

ACF: Autocorrelation function

AR: Autoregressive model

ARIMA: Autoregressive Integrated Moving Average

ARIMAX: Autoregressive Integrated Moving Average with exogenous variable

ARMA: Autoregressive Moving average model

BI: Business Intelligence

ECLAT: Equivalence Class Transformation Algorithm

ETS: Simple exponential smoothing

I: Integrating model

MA: Moving averages model

MAE: Mean Absolute Error

MAPE: Mean Absolute Percentage Error

PACF: Partial autocorrelation function

RFM: Recency, Frequency, and Monetary technique

RMSE: Root Mean Square Error

SARIMA: Seasonal Autoregressive Integrated Moving average model

TBATS: Trigonometric Exponential Smoothing Method

APPENDICES

APPENDIX 1: SCRIPT R STUDIO ARIMA METHOD

```
library(readxl)
library(forecast)
library(tseries)

#carga de los datos
datos <- read_excel("DATOS-50-PROD.xlsx",sheet = "2014-2017")
futuro <- read_excel("DATOS-50-PROD.xlsx",sheet = "2018")

actual <- as.numeric(futuro$`2020`)
product <- "Pedidos 2020"
pedidos <- as.numeric(datos$`2019`)

# convertimos los valores a series temporales
ped <- ts(pedidos,start=c(2014,1),frequency=12)
plot(ped,xlab = "Años",ylab = "Cantidad",main = "Serie temporal")

# evaluacion del modelo
plot(stl(ped,s.window = "periodic"),main="Componentes")

# ajuste modelo
acf(ped,main = "Autocorrelación")
pacf(ped, main = "Autocorrelación parcial")

adf <- adf.test(ped) #si p mayor que 0.05 la serie no es estacionaria,
nueva diff
diff.ped <- diff(ped)
plot(diff.ped,xlab= "Años",main="Serie diferenciada",ylab="diferenciación")
adf.1 <- adf.test(diff.ped) #si p mayor que 0.05 la serie no es
estacionaria, nueva diff

acf(diff.ped,main = "Autocorrelación diferenciación")
pacf(diff.ped, main = "Autocorrelación parcial diferenciación")

# comprobacion diferencias modelo
if(adf$p.value > 0.05){
  d = 1
}else{
  d = 0
  }

if(d == 1){
  if(adf.1$p.value > 0.05){
    d = 2
  }else{
    d=1
    }
}else{
  d=0
}

D <- nsdiffs(ped) # diferenciacion estacional

# ajuste modelo
modelo <- auto.arima(ped,d,D,stepwise=T,approximation = T,trace=T)

# comprobacion residuos para ruido blanco y validez del modelo
tsdisplay(modelo$residuals,main = "Residuos del modelo")
box.test <- Box.test(modelo$residuals, type="Ljung-Box")
```

```
# comprobacion ruido blanco
# debe ser mayor a 0.05 para datos distribuidos independientemente
if(box.test$p.value < 0.05){
    break("modificar modelo, datos con distribucion NO independiente")
 }

# pronostico una vez el modelo está validado
pronostico <- forecast(modelo,h=12)
plot(pronostico,xlab = "Meses",ylab = "Unidades",main = "Pronóstico
2020",sub=modelo)
pronostico

pron <- as.data.frame(pronostico)
pron1 <- pron$`Point Forecast`

mape <- function(actual,pred){
  mape <- (1/12)*((mean(abs(actual - pred)/actual)))*100
  return (mape)
}
mape(actual,pron1)

# calculo cuando mape se indetermina

mae.media <- function(actual,pred){
  mae.media <- mean((1/12)*((abs(actual-pred))/mean(actual+pred)))*100
  return(mae.media)
}
mae.media(actual,pron1)

error.cuadratico <- function(actual,pred){
  error.cuadratico <- mean((abs(actual-pred)^2)/(actual)^2)*100
  return(error.cuadratico)
}
error.cuadratico(actual,pron1)

# graficas pronostico

plot(pronostico,
     ylim=c(-250,13000),
     ylab="cantidad kg",
     col="blue",
     type="l")

par(new=T)

plot(c(pedidos,actual),
     ylim=c(-250,13000),
     xaxt="n",
     xlab= "Años",
     ylab="cantidad kg",
     col="red",
     sub= product,
     type="l")


write.xlsx(datos,"result.xlsx")
write.xlsx(pronostico,"resultado.xlsx")
```

APPENDIX 2: ARIMA PROCEDURE RESULTS

The ARIMA method procedure and results are presented in this appendix where 2019 is forecasted using the type A company products.

The input is transformed into a time series and is represented in Figure ***21***:

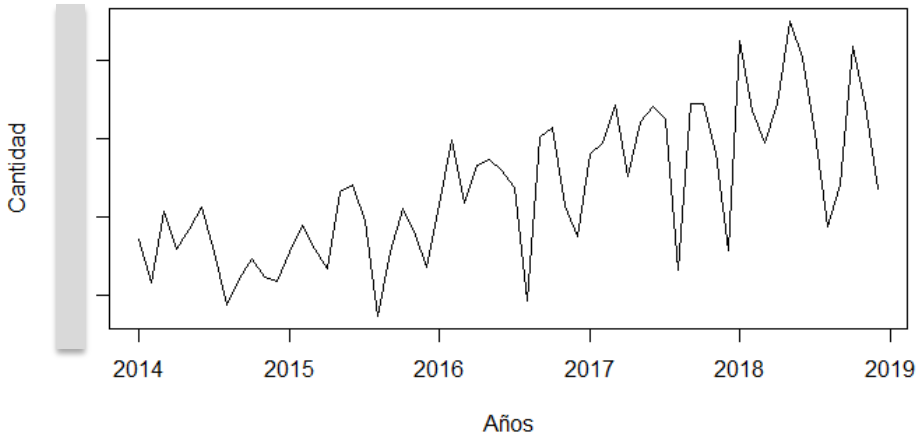


Figure 21 Time series

The time series components; trend, seasonality, and cyclical component are depicted in Figure ***22***:

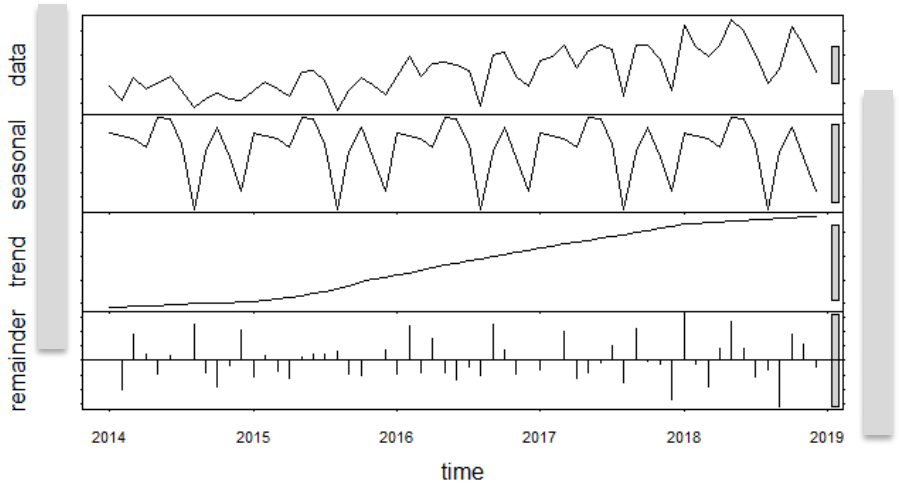


Figure 22 Time series components

Once the time series components are known, the ARIMA parameters must be determined using the autocorrelation and autocorrelation partial correlograms.

On the one hand, the autocorrelation plot is represented in Figure ***23***:

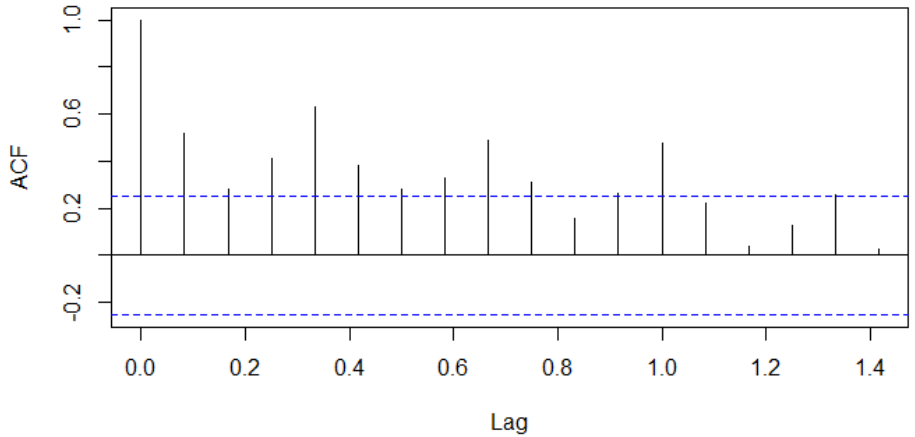


Figure 23 Autocorrelation plot

On the other hand, the autocorrelation partial plot is represented in Figure ***24***:

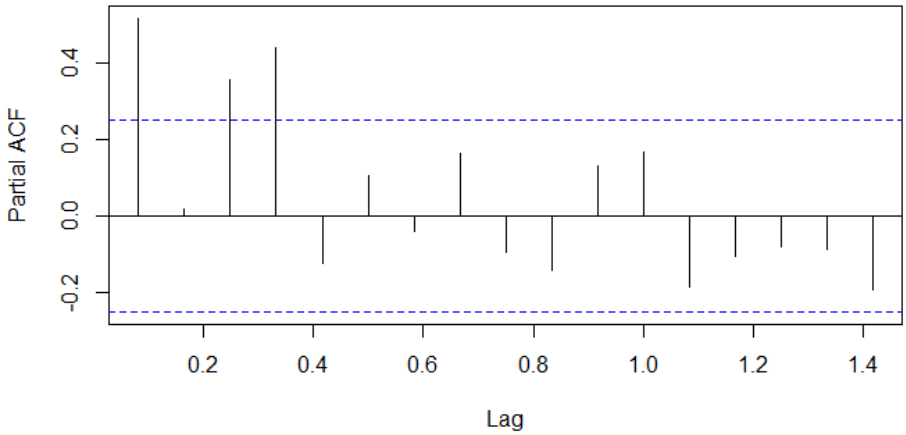


Figure 24 Partial autocorrelation plot

Then, the Dickey-Fuller test is applied to know if the time series is stationary or non-stationary. To do this, hypothesis 0 is checked and if it is rejected, differentiation will be applied to the time series. In Figure ***25*** are the hypothesis 0 result:

```
        Augmented Dickey-Fuller Test

data:  ped
Dickey-Fuller = -1.4476, Lag order = 3, p-value = 0.7925
alternative hypothesis: stationary
```

Figure 25 Hypothesis 0 result

In this example, hypothesis 0 is rejected so differentiation is applied to the time series. The differentiation applied to the time series is showed in Figure ***26***:

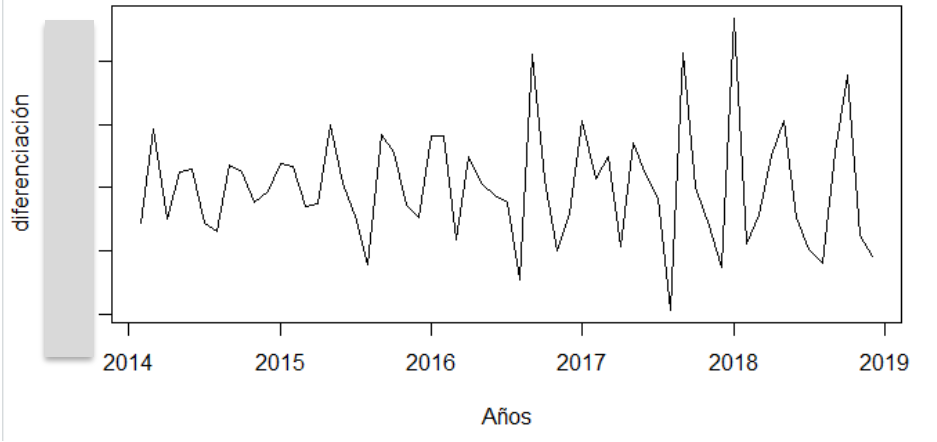


Figure 26 Serie with differentiation

Once the differentiation is applied, the Dickey-Fuller test is applied again, in this case, the hypothesis 0 is not rejected and the time series is already stationary. In Figure ***27*** are the hypothesis 0 result for the series with differentiation:

```
        Augmented Dickey-Fuller Test

data:  diff.ped
Dickey-Fuller = -4.2976, Lag order = 3, p-value = 0.01
alternative hypothesis: stationary
```

Figure 27 Hypothesis 0 result

When the differentiation parameter (d) is determined, an automatic function in R Studio can be applied to obtain the autoregressive (p) and moving averages (q) parameters. It gives the best model's parameters in Figure ***28***.

```
Best model: ARIMA(2,1,1)(0,1,1)[12]
```

Figure 28 Best model

The Box-Jenkins test is applied to the remainders to validate the model. With hypothesis 0 the model's validation will be determined. The model's remainders are represented in Figure ***29*** and ***30***:

```
        Box-Ljung test

data:  modelo$residuals
X-squared = 0.033449, df = 1, p-value = 0.8549
```

Figure 29 Hypothesis 0 result

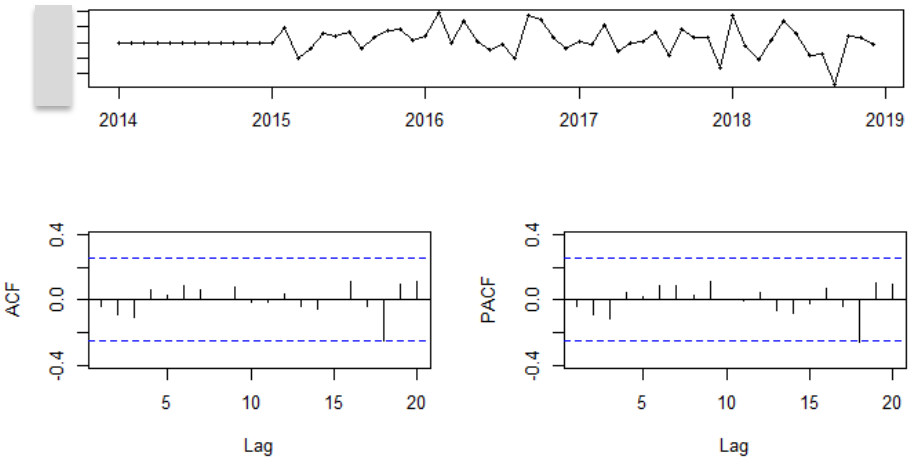


Figure 30 Model's remainders

Once the model is validated, the forecasting is done for 12 months of 2019. The results are shown in Figure ***31***:

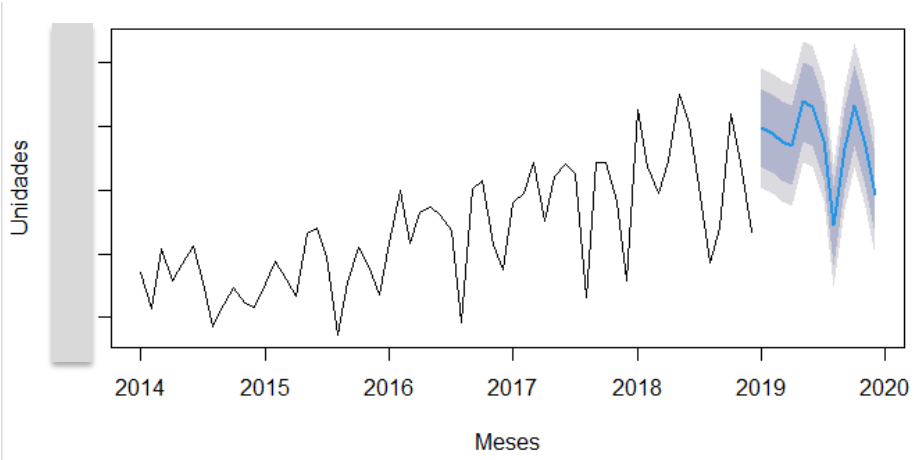


Figure 31 Demand forecasting results for 2019

And the forecasting error is calculated and plotted, Figure ***32*** represents the comparison between real data and forecasted data:

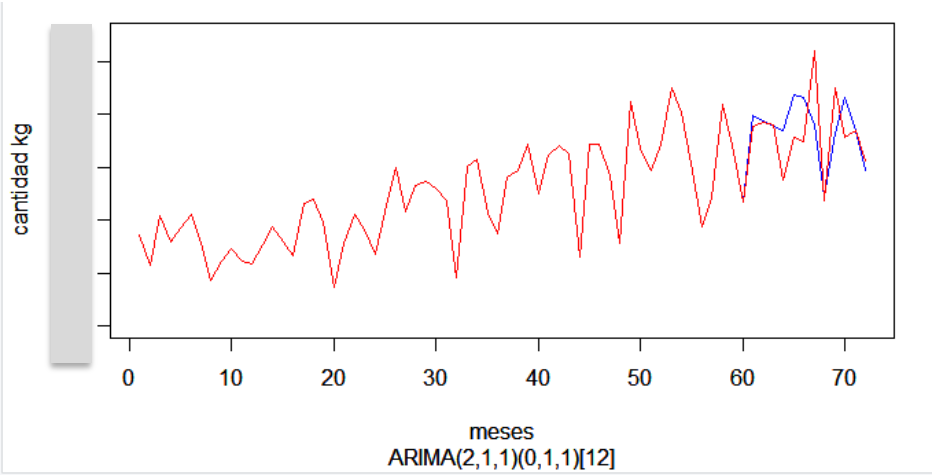


Figure 32 Comparison real and forecasted data

The forecasting error is calculated in Table ***13***:

Table 13 Forecasting error

METHOD	ERROR (%)
MAPE	5,67

APPENDIX 3: SCRIPT R STUDIO ECLAT ALGORITHM

```
library(arules)
library(methods)

#carga de los datos

datos <- read.csv("Libro4.csv",sep=(";"),header = T)
datos <- subset(datos, !is.na(datos$seccion.gfh.Seccion))

trans <- split(datos$seccion.gfh.Seccion,datos$N°Albar)

transacciones <- as(trans,"transactions")

#transformamos datos a transacciones

itemFrequencyPlot(transacciones,topN = 25,ylim = c(0,0.5),ylab =
"Frecuencia item")

#realizamos el algoritmo; soporte minimo = 0.25 y minimo 2 productos

itemsets <- eclat(data=transacciones,parameter=list(support=0.01,minlen=2))

#muestra los resultados ordenados por soporte decreciente

resultado <- as.data.frame(inspect(sort(itemsets,by='support')))
```

APPENDIX 4: SCRIPT R STUDIO CUSTOMERS SEGMENTATION

```
library(readxl)
library(ggplot2)
library(scales)
library(plyr)
library(rfm)
library(dplyr)
library(DT)
library(lubridate)
library(plotly)
library(ggthemes)
library(rpart)
library(rpart.plot)
library(factoextra)

data <- read_excel("Ventas.xlsx")

# redefiniendo los tipos de datos ---------------------------

# los que son factores, los dejamos como factores
data$Fecha <- as.factor(as.character(data$Fecha))
data$Codart <- as.factor(as.character(data$Codart))
data$Descripción <- as.factor(as.character(data$Descripción))
data$`Razón Social` <- as.factor(as.character(data$`Razón Social`))

# revisión y ajuste de Razón Social ---------------------------

# cuántos valores únicos hay de Razón Social?
length(unique(data$`Razón Social`))

# cuántos CustomerID con NA?
sum(is.na(data$`Razón Social`))

# eliminamos los CustomerID con NAs
data <- subset(data, !is.na(data$`Razón Social`))

# cuántos registros quedan?
nrow(data)

# no está en formato fecha, por lo que realizamos la creación de un nuevo
atributo
data$FechaFF<-as.Date(substr(as.character(data$Fecha),1,10))

# qué rango de fechas hay?
range(data$FechaFF)

# eliminamos el atributo anterior de fecha
data$Fecha <-NULL

###SELECCION DEL PERIODO DE TIEMPO A
UTILIZAR-----------------------------------------------

# respaldamos antes de realizar la operación de selección de un año de
datos
dataold2<- data

# seleccionaremos los datos de un año, por lo que tomaremos el caso:
InvoiceDate >= "2019-10-27"
data <- data[which(data$FechaFF >= as.Date("2017-01-02","%Y-%m-%d")),]
```

```
# cuántos registros quedan?
nrow(data)

# entonces el rango de fechas quedo como sigue
range(data$FechaFF)

### TENEMOS DATOS PREPARADOS PARA TRABAJAR
### CALCULAMOS VARIABLES DEL MODELO RFM: RECENCY, FRECUENCY, MONETARY

# identificar los retornos --------------------------

data$item.return <- grepl("C", as.character(data$N°Albar), fixed=TRUE)
data$factura <- ifelse(data$item.return=="TRUE", 0, 1)

# creación de variables RFM --------------------------

# primero crearemos un dataset de clientes que completaremos después
clientes <- as.data.frame(unique(data$`Razón Social`))
# dejamos la columna con el nombre de CustomerID
names(clientes) <- "Razón Social"

### CALCULAMOS EL
RECENCY-----------------------------------------------------------------

data$recency <- Sys.Date() - data$FechaFF

# removemos los retornos, sonsiderando sólo las compras: *purchase*, y lo
dejamos en una tabla temporal
temp <- subset(data, factura == 1)

# obtenemos el número de días desde la compra más reciente y lo dejamos en
otra tabla temporal
recency <- aggregate(recency ~ `Razón Social`, data=temp, FUN=min,
na.rm=TRUE)
# eliminamos la tabla temporal temp
remove(temp)

# agregamos el recency a la tabla de clientes que creamos
clientes <- merge(clientes, recency, by="Razón Social", all=TRUE,
sort=TRUE)
# eliminamos la tabla temporal recency
remove(recency)
# dejamos como numérico el recency en la tabla de clientes
clientes$recency <- as.numeric(clientes$recency)

### CALCULAMOS EL
FRECUENCY---------------------------------------------------------------

# seleccionamos algunos atributos
factura.cliente <- subset(data, select = c("Razón Social","N°Albar",
"factura"))
# eliminamos los duplicados
factura.cliente <- factura.cliente[!duplicated(factura.cliente), ]
# ordenamos con CustomerID
factura.cliente <- factura.cliente[order(factura.cliente$`Razón Social`),]
row.names(factura.cliente) <- NULL
# obtenemos el número de facturas por año solo para las compras
```

```
factura.anual <- aggregate(factura ~ `Razón Social`, data=factura.cliente,
FUN=sum, na.rm=TRUE)
#cambiamos el nombre de la columna que agregó los datos a frecuency
names(factura.anual)[names(factura.anual)=="factura"] <- "frequency"
# lo agregamos a los datos del cliente
clientes <- merge(clientes, factura.anual, by="Razón Social", all=TRUE,
sort=TRUE)
# eliminamos los datasets que ya no necesitamos
remove(factura.cliente, factura.anual)
# veamos cómo quedó frecuency
range(clientes$frequency)
table(clientes$frequency)

# removamos los clientes que no tienen ninguna compra en el año pasado
clientes <- subset(clientes, frequency > 0)
# veamos cuántos clientes nos quedan
nrow(clientes)

### CALCULAMOS EL
MONETARY-----------------------------------------------------------------

data$Cantidad <- data$Unidades * data$Precio
# agreguemos el total de ventas por cliente
ventas.anuales <- aggregate(Cantidad ~ `Razón Social`, data=data, FUN=sum,
na.rm=TRUE)
# cambiemos el nombre de la columna a monetary
names(ventas.anuales)[names(ventas.anuales)=="Cantidad"] <- "monetary"
# agreguemos la columna monetary a nuestro dataset de clientes
clientes <- merge(clientes, ventas.anuales, by="Razón Social", all.x=TRUE,
sort=TRUE)
# eliminemos el dataset temporal
remove(ventas.anuales)

hist(clientes$monetary)

### REGLA DEL 80/20-------------------------------------------------

# ordenamos los clientes de mayor a menor monetary
clientes <- clientes[order(-clientes$monetary),]
clientes$monetary[is.na(clientes$monetary)] <- 0
# calculamos el corte del 80%
pareto.cutoff <- 0.8 * sum(clientes$monetary)
# de acuerdo a este corte clasificamos a los clientes
clientes$pareto <- ifelse(cumsum(clientes$monetary) <= pareto.cutoff, "Top
20%", "Bottom 80%")
# dejamos el atributo como factor
clientes$pareto <- factor(clientes$pareto, levels=c("Top 20%", "Bottom
80%"), ordered=TRUE)
# vemos los niveles
levels(clientes$pareto)
# y vemos la proporción que nos quedaron
round(prop.table(table(clientes$pareto)), 2)

# eliminamos la variable que creamos para el corte
remove(pareto.cutoff)
# dejamos los datos ordenados por CustomerID
clientes <- clientes[order(clientes$`Razón Social`),]
```

```
# visualicemos los datos con las variables frecuency y monetary
scatter.1 <- ggplot(clientes, aes(x = frequency, y = monetary))
scatter.1 <- scatter.1 + geom_point(aes(colour = recency, shape = pareto))
scatter.1 <- scatter.1 + scale_shape_manual(name = "Designación 80/20",
values=c(17, 16))
scatter.1 <- scatter.1 + scale_colour_gradient(name="Recency\n(Días desde
último pedido))")
scatter.1 <- scatter.1 + xlab("Frequency (Número de pedidos)")
scatter.1 <- scatter.1 + ylab("Monetary clientes (Ventas Anuales,Euros)")
scatter.1

remove(scatter.1)

###
PARAMETRIZACION-------------------------------------------------------------

# Donde hacer los cortes
punt.rec <- as.data.frame(quantile(clientes$recency, probs = c(0, 0.25,
0.5, 0.75, 1)))
punt.frec <- as.data.frame(quantile(clientes$frequency, probs = c(0, 0.25,
0.5, 0.75, 1)))
punt.mon <- as.data.frame(quantile(clientes$monetary, probs = c(0, 0.25,
0.5, 0.75, 1)))

# puntaje para recency
# R_score
clientes$R_Score[clientes$recency>punt.rec$`quantile(clientes$recency,
probs = c(0, 0.25, 0.5, 0.75, 1))`[4]]<-1
clientes$R_Score[clientes$recency>punt.rec$`quantile(clientes$recency,
probs = c(0, 0.25, 0.5, 0.75, 1))`[3] &
clientes$recency<=punt.rec$`quantile(clientes$recency, probs = c(0, 0.25,
0.5, 0.75, 1))`[4]]<-2
clientes$R_Score[clientes$recency>punt.rec$`quantile(clientes$recency,
probs = c(0, 0.25, 0.5, 0.75, 1))`[2] &
clientes$recency<=punt.rec$`quantile(clientes$recency, probs = c(0, 0.25,
0.5, 0.75, 1))`[3]]<-3
clientes$R_Score[clientes$recency<=punt.rec$`quantile(clientes$recency,
probs = c(0, 0.25, 0.5, 0.75, 1))`[2]]<-4

# puntaje para frecuency
# F_score
clientes$F_Score[clientes$frequency>punt.frec$`quantile(clientes$frequency,
probs = c(0, 0.25, 0.5, 0.75, 1))`[4]]<-4
clientes$F_Score[clientes$frequency>punt.frec$`quantile(clientes$frequency,
probs = c(0, 0.25, 0.5, 0.75, 1))`[3] &
clientes$frequency<=punt.frec$`quantile(clientes$frequency, probs = c(0,
0.25, 0.5, 0.75, 1))`[4]]<-3
clientes$F_Score[clientes$frequency>punt.frec$`quantile(clientes$frequency,
probs = c(0, 0.25, 0.5, 0.75, 1))`[2] &
clientes$frequency<=punt.frec$`quantile(clientes$frequency, probs = c(0,
0.25, 0.5, 0.75, 1))`[3]]<-2
clientes$F_Score[clientes$frequency<=punt.frec$`quantile(clientes$frequency,
probs = c(0, 0.25, 0.5, 0.75, 1))`[2]]<-1

# puntaje para monetary
# M_score
clientes$M_Score[clientes$monetary>punt.mon$`quantile(clientes$monetary,
probs = c(0, 0.25, 0.5, 0.75, 1))`[4]]<-4
```

```
clientes$M_Score[clientes$monetary>punt.mon$`quantile(clientes$monetary,
probs = c(0, 0.25, 0.5, 0.75, 1))`[3] &
clientes$monetary<=punt.mon$`quantile(clientes$monetary, probs = c(0, 0.25,
0.5, 0.75, 1))`[4]]<-3
clientes$M_Score[clientes$monetary>punt.mon$`quantile(clientes$monetary,
probs = c(0, 0.25, 0.5, 0.75, 1))`[2] &
clientes$monetary<=punt.mon$`quantile(clientes$monetary, probs = c(0, 0.25,
0.5, 0.75, 1))`[3]]<-2
clientes$M_Score[clientes$monetary<=punt.mon$`quantile(clientes$monetary,
probs = c(0, 0.25, 0.5, 0.75, 1))`[2]]<-1

# RFM_score
clientes<- clientes %>% mutate(RFM_Score = 100*R_Score +
10*F_Score+M_Score)

recomendados <- c(444)
clientes_leales <- c(334, 342, 343, 344, 433, 434, 443)
potenciales_leales <-
c(332,333,341,412,413,414,431,432,441,442,421,422,423,424)
clientes_recientes <- c(411)
promesa <- c(311, 312, 313, 331)
necesita_atencion <- c(212,213,214,231,232,233,241,314,321,322,323,324)
dormido <- c(211)
en_riesgo <- c(112,113,114,131,132,133,142,124,123,122,121,224,223,222,221)
no_perder <- c(134,143,144,234,242,243,244)
hibernando <- c(141)
perdido <- c(111)

clientes$segmentRFM <- NA
clientes$segmentRFM[which(clientes$RFM_Score %in% recomendados)] =
"Recomendados"
clientes$segmentRFM[which(clientes$RFM_Score %in% clientes_leales)] =
"Clientes leales"
clientes$segmentRFM[which(clientes$RFM_Score %in% potenciales_leales)] =
"Potenciales leales"
clientes$segmentRFM[which(clientes$RFM_Score %in% clientes_recientes)] =
"Clientes recientes"
clientes$segmentRFM[which(clientes$RFM_Score %in% promesa)] = "Promesa"
clientes$segmentRFM[which(clientes$RFM_Score %in% necesita_atencion)] =
"Necesita atención"
clientes$segmentRFM[which(clientes$RFM_Score %in% dormido)] = "Dormido"
clientes$segmentRFM[which(clientes$RFM_Score %in% en_riesgo)] = "En riesgo"
clientes$segmentRFM[which(clientes$RFM_Score %in% no_perder)] = "No se
puede perder"
clientes$segmentRFM[which(clientes$RFM_Score %in% hibernando)] =
"Hibernando"
clientes$segmentRFM[which(clientes$RFM_Score %in% perdido)] = "Perdido"

resultado <- as.data.frame(clientes)
```

APPENDIX 5: SCRIPT R STUDIO QUANTITY SEGMENTATION

```
# paquetes necesarios
library(readxl)
library(ggplot2)
library(dplyr)

datos <- read_excel("consulta.xlsx",sheet = "datos")

# deteccion outliers mediante IRQ*1,5
data.riq <- IQR(datos$Unidades)

# calcular Q1 y Q3
cuantiles <- quantile(datos$Unidades, c(0.25, 0.5, 0.75), type = 7)

# determinacion barrera menor
outliers_min <- as.numeric(cuantiles[1])-1.5*data.riq

# determinacion barrera mayor
outliers_max <- as.numeric(cuantiles[3])+1.5*data.riq

# boxplot con outliers
bp = boxplot(datos$Unidades,ylab = "Cantidad (kg)",main = "Boxplot")

# calculo extremos boxplot
out_inf = bp$stats[1]
out_sup = bp$stats[5]
cat("Extremo inferior", out_inf)
cat("Extremo superior", out_sup)

# boxplot sin outliers
boxplot(datos$Unidades[datos$Unidades>outliers_min &
datos$Unidades<outliers_max],
        decreasing = FALSE,ylab = "Cantidad (kg)",main = "Boxplot")

#resultados de pedidos buenos
pedidos <- subset(datos,Unidades<out_sup & Unidades>out_inf )

#resultados de pedidos anomalos
outliers.sup <- as.data.frame(subset(datos,Unidades>out_sup))
outliers.inf <- as.data.frame(subset(datos,Unidades<out_inf))

outliers <- rbind(outliers.sup,outliers.inf)

ped.anomalos <- as.data.frame(cbind(as.data.frame(pedidos$FechaFF),
                                    pedidos$`Razón Social`,
                                    pedidos$N°Albar,
                                    pedidos$Codigo,
                                    pedidos$Descripción,
                                    pedidos$Unidades))

ped.outlier <- as.data.frame(cbind(as.data.frame(outliers$FechaFF),
                                   outliers$`Razón Social`,
                                   outliers$N°Albar,
                                   outliers$Codigo,
                                   outliers$Descripción,
                                   outliers$Unidades))

colnames(ped.anomalos) <-
c("Fecha","Cliente","NoAlb","Codigo","Descripcion","Cantidad (kg)")


colnames(ped.outlier) <-
c("Fecha","Cliente","NoAlb","Codigo","Descripcion","Cantidad (kg)")
```

APPENDIX 6: SCRIPT R STUDIO CLUSTERING METHOD

The algorithm k-means is used to determine which cluster each customer will be for each parameter.

Then, the results are plotted in Power BI as is showed in point 7.

```
library(cluster)
library(factoextra)
library(purrr)
library(tidyverse)

# Carga de datos inicial

datos <- read_excel("result.xlsx")

prod <- datos$`Razón Social`

# CLUSTER RECENCY----------------------------------------

cluster <- kmeans(datos[,2],centers = 5)

datos$Cluster_Recency <- cluster$cluster

#CLUSTER FRECUENCY---------------------------------------

cluster <- kmeans(datos[,3],centers = 3)

datos$Cluster_Frecuency <- cluster$cluster

#CLUSTER MONETARY----------------------------------------

cluster <- kmeans(datos[,4],centers = 5)

datos$Cluster_Monetary <- cluster$cluster
```

APPENDIX 7: SCRIPT R STUDIO MONTE CARLO SIMULATION

```
library(readxl)
library(forecast)
library(dplyr)
library(openxlsx)

# carga datos
datos <- read_excel("PRODUCTOS TIPO A pol-solv.xlsx",sheet = "montecarlo
codigo 2020")

n <- 5
n.ped <- 930529

# calculo frecuencias
frec.producto <- prop.table(datos$Unidades)
frec.acm.prod <- (c(rep(0,1),cumsum(frec.producto)))

# frecuencias productos pedidos
Codigo <- datos$Codigo

productos <- vector(mode="character", length=n.ped)
resultados <- matrix(ncol = n, nrow = n.ped)

# simulacion montecarlo producto envase
for (j in 1:n){
  for (i in 1:n.ped){
    x <- runif(1,min=0,max=1)
    interval.prod <- findInterval(x,frec.acm.prod)
    prod <- Codigo[interval.prod]
    productos[i] <- prod
  }

  resultados[,j] <- productos
}

result <- as.data.frame(resultados)

# creamos nuevo data frame donde juntaremos recuento de resultados
simulacion <- as.data.frame(Codigo)

# creamos data frame por cada simulacion
result1 <- as.data.frame(table(result[,1]))
colnames(result1) <- c("Codigo","Recuento")

result2 <- as.data.frame(table(result[,2]))
colnames(result2) <- c("Codigo","Recuento")

result3 <- as.data.frame(table(result[,3]))
colnames(result3) <- c("Codigo","Recuento")

result4 <- as.data.frame(table(result[,4]))
colnames(result4) <- c("Codigo","Recuento")

result5 <- as.data.frame(table(result[,5]))
colnames(result5) <- c("Codigo","Recuento")

# unimos cada data frame por Codigo, dejando los vacíos como NA
simulacion <- merge(result1,simulacion,by = "Codigo",all.x = T,all.y = T)
simulacion <- merge(result2,simulacion,by = "Codigo",all.x = T,all.y = T)
```

```
simulacion <- merge(result3,simulacion,by = "Codigo",all.x = T,all.y = T)
colnames(simulacion) <- c("Codigo","sim3","sim2","sim1")
simulacion <- merge(result4,simulacion,by = "Codigo",all.x = T,all.y = T)
simulacion <- merge(result5,simulacion,by = "Codigo",all.x = T,all.y = T)
colnames(simulacion) <- c("Codigo","sim5","sim4","sim3","sim2","sim1")

# reemplazo de NA por 0
simulacion[is.na(simulacion)] <- 0

# eliminamos la columna Codigo y la ponemos como nombre de filas
simulacion <- simulacion[with(simulacion, order(simulacion$Codigo)), ]
rownames(simulacion) <- simulacion$Codigo
simulacion <- simulacion[2:6]

# creacion nuevos vectores donde ira almacenado el maximo y minimo
unidades <- vector()
Codigo <- rownames(simulacion)
maxmin <- matrix()

for(i in 1:nrow(simulacion)){
  unidades[i] <- sum(simulacion[i,])/length(simulacion)
  maxmin <- as.data.frame(cbind(Codigo,unidades))
}

rm(result,simulacion,result1,result2,result3,result4,result5,resultados)
maxmin <- as.data.frame(maxmin)
```