University College Dublin

Michael Smurfit Graduate Business School

MSc Digital Innovation

MIS41230 Machine Learning for Business

Summer Term 2020/2021

Team Project

Beverage Demand

Written by:

Jéssica Egoávil Arbañil, 20203636, jessica.egoavil@ucdconnect.ie

Anastasia Kučerovska, 13350606, anastasia.kucerovska@ucdconnect.ie

Jack Ridgway, 20202568, jack.ridgway@ucdconnect.ie

Table of Contents

I. Executive Summary

Demand and sales forecasting are two of the most critical processes of manufacturers, distributors, and retailers. Keeping demand and supply in balance not only reduces excess and shortage of inventories but also improves profitability. When the producer aims to fulfil the overestimated demand, the excess production results in extra stock keeping. On the contrary, underestimated demand causes unfulfilled orders, lost sales, foregone opportunities and reduces service levels. Both scenarios lead to an inefficient supply chain.

In consequence, the accurate demand forecast is a real challenge for a participant in the supply chain. The ability to forecast the future based on past data is crucial to support individual and organizational decision-making. Even when there are several machine learning approaches to analyse from historical data, in this paper, there is a novel attempt to integrate the three different forecasting models that include: Linear Regression, Decision Tree and Random Forest for demand forecasting.

The most accurate models were chosen based on the context in which they were applied. Linear performed best overall, however a decision tree was ideal in a particular situation. Accurate predictions for order quantity and sales predictions were created.

This paper has also examined the limitations of the data set available and the narrow focus of the models used to predict the demand. Furthermore, it was noted that given the opportunity, a larger and more integrated data set would have been requested to make more accurate predictions, as well as the use of other interesting models that could have given us different answers.

II. Introduction And Literature Review

The amount of data collected is ever-increasing, summarizing what people, systems, and organizations do and capturing their behaviour in detail. This is collected due to the perceived value, regardless of the knowledge of its use. Even when it was used by organisations to report summaries, using business intelligence approaches, this has changed in the last decade. Organisations have discovered that predictive analytics can change the way that those organisations make key decisions regarding their businesses and processes.

Since then, businesses have evolved in the way they leverage information, from traditional "descriptive" analytic methods, to increasing the use of "predictive" analytic methods. Understanding that descriptive analytic methods allow people to get a retrospective view on the business, answering questions like ‘What has happened?’, ‘How big was the sale?’, ‘When will it be done?’ On the contrary, predictive analytics allows us to get a perspective of the business, answering questions like: ‘What is the amount I will need to sell if this trend continues?’ While traditional database management systems and relational frameworks have provided the engine for enabling "descriptive" analytics that permits users to query and get rich reports from data, it is expected that Machine Learning will fuel the use of "Predictive Analytics" from data. (Center, 2019)

Machine learning techniques enable us to accurately forecast multiple aspects of supply chain management such as demand, sale, revenue, production, and backorder. Some researchers applied a representative set of machine learning and traditional forecasting methods to a dataset to compare the precision of those methods. Their findings were that the average performances of the machine learning method did not outperform the traditional methods. However, when a Support Vector Machine was trained on several demand series, it produced the most precise predictions that greatly benefitted the organisation. The same researchers extended their work, using Support Vector Machines and Neural Networks. They concluded that the techniques of applying machine learning models provided noticeable improvements over the traditional models (Islam & Amin, 2020).

According to Carbonneau, et al. (2008), the use of machine learning techniques and Multiple Linear Regression (MLR) for forecasting distorted demand signals in the extended supply chain provide more accurate forecasts than simpler forecasting techniques (including naïve, trend, and moving average). However, the same author did not find that machine learning techniques perform significantly better than linear regression. Thus, the marginal gain in accuracy of the Recurrent Neural Networks and Support Vector Machine models should be weighed in practice against the conceptual and computational simplicity of the linear regression model.

In the context of the fast-moving consumer goods (FMCG) industry, Tarallo, et al. (2019) mentions that machine learning results in better demand predictability than the use of traditional models. One of the benefits of this better sales forecast accuracy is an improvement in inventory management due to better decisions on production and distribution. The decrease in the occurrence of stockouts, which cause a shortage of items in stores, is also mentioned, as well as the greater availability of products at the points of sale, thus increasing revenue and customer satisfaction.

The main benefits observed were reducing human bias due to the greater automation of the forecasting process, a higher degree of forecast precision, and the flexibility to change variables. The drawbacks are, the possible unavailability of detailed historical information, and the large number of learning algorithms available, which could make choosing the proper one difficult. (Fujimaki, et al., 2016)

III. Data Set and Business Case

This section explains in detail the dataset used for this project on beverage demand. It also gives background about its selection and what real business scenario was chosen to be solved in this paper.

A. Product Demand Forecasting

The dataset used, named “Beverage demand”, was obtained from one of our team members’ own professional experiences. It contains historical product sales data for a company that imports different kinds of beverages from different suppliers around the world. It has roughly a million and a half observations in the form of sale orders in units and value. It is categorized in 723 unique products categorized in 63 brands and 13 categories. Each row represents a sales transaction specifying the product, its categorization, the customer, sales representative that did the sale and more information specified in Appendix A.

The dataset was chosen for the following reasons. Firstly, it is large enough compared to public datasets found on the web that were shorter, having over eighty thousand rows and twenty different variables. Secondly, even when it was acquired in a professional environment, the dataset is not attached to privacy rights because it was freely shared by a person that joined from another company voluntarily and freely. It is important to highlight that the dataset is five years old, which means that although it is not recently updated, it is relevant and may be used to conduct analysis on. Finally, because the dataset derives from a real company and it is very rich in information, it was used for different business scenarios. Its most common usages were related to supply chain, marketing, and sales. Since the company is located in South America, the information contains Spanish terminology; however, the headers were translated into English for this paper and there is a brief vocabulary summary in Appendix A that explains some of the terms' meaning, when needed. For this essay purposes, the business case chosen will be focused on supply chain business challenges, specifically the prediction of future sales and demand.

B. Related Business Case

The dataset used might be classified as part of the Fast-Moving Consumption Growth industry, specifically for beverages. Even when the database comes from a multinational beverage company, the dataset is from the Peruvian branch only. This branch imports all products from other locations so it is relevant for top managers to plan their imports and sales in advance to ensure they will be able to achieve their business objectives. Due to the geographically distributed places where products are imported from and the size and priority of the Peruvian market, the import timings vary from 3 to 6 months on average. Considering this context, top leaders usually have monthly and quarterly meetings to internally agree on the purchase orders they are going to request for the following months. At these business meetings, every top leader brings their own data analysis to support their import suggestion and justify how this is in alignment with the business target that the company has asked for. Then, this paper aims to give them a practical business tool that helps them to make future predictions in both units and sales (revenue) that enable them to have these conversations efficiently and effectively.

IV. Data Analysis

In order to be able to apply machine learning processes to our business case, the data set was first prepared and explored prior to in depth analysis. Described below are the steps that were taken to dissect the data set, gain a better understanding of beverage imports, apply to our selected models, and be able to predict the demand for beverages in order to assist the managers of the organisation in the future.

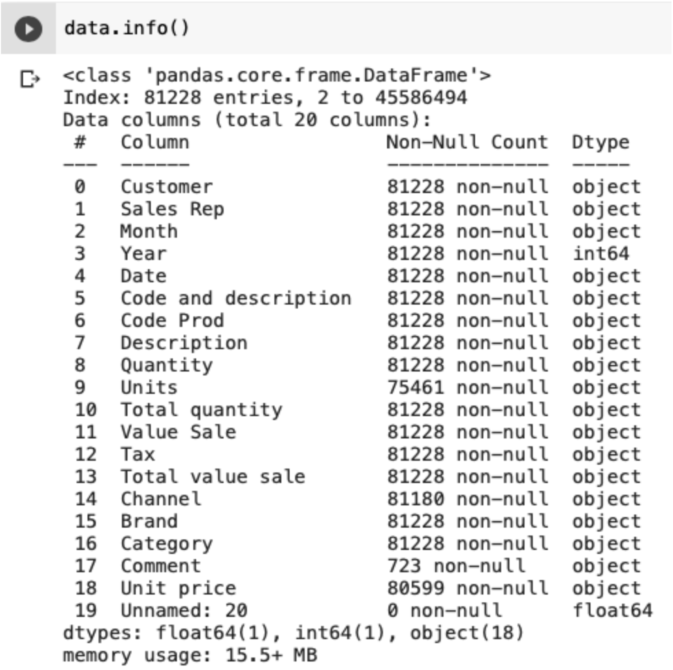
A. Data Preparation

To start off, we wanted to explore the data set, which was visualised, to create a basic understanding of what was within our CSV file and learn what preparation would have to be made on the data set. Géron (2019) makes it clear that "if your training data is full of errors, outliers, and noise [ ] , it will be harder for the system to detect the underlying patterns, so your system is less likely to perform well." Any data points that could cause issues further down the line needed to be identified and dealt with promptly.

B. Data Exploration

The dataset was visualised in two ways; using the .info() method (see Figure 1), and a for loop (see Figure 2).

The data.info() as seen in Figure 1 displayed basic information about the dataset, including the index and column information. The index states that there are 81228 entries meaning that there are 81228 rows of information. The table displayed the column number, beginning at 0, the corresponding column name, the non-null count, i.e., the number cells containing data, and finally the Dtype, i.e. the type of data that is stored within the column.



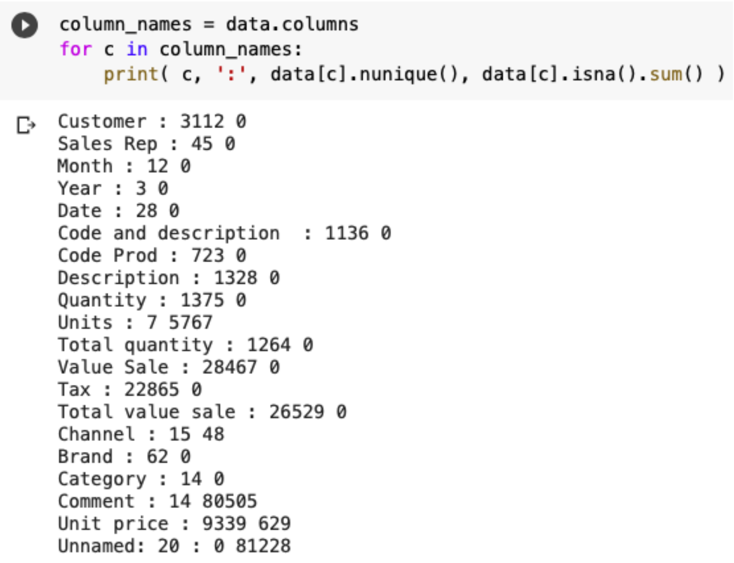
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 81228 entries, 2 to 45586494
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Customer              81228 non-null  object
 1   Sales Rep             81228 non-null  object
 2   Month                 81228 non-null  object
 3   Year                  81228 non-null  int64
 4   Date                  81228 non-null  object
 5   Code and description  81228 non-null  object
 6   Code Prod             81228 non-null  object
 7   Description           81228 non-null  object
 8   Quantity              81228 non-null  object
 9   Units                 75461 non-null  object
 10  Total quantity        81228 non-null  object
 11  Value Sale            81228 non-null  object
 12  Tax                   81228 non-null  object
 13  Total value sale      81228 non-null  object
 14  Channel               81180 non-null  object
 15  Brand                 81228 non-null  object
 16  Category              81228 non-null  object
 17  Comment               723 non-null    object
 18  Unit price            80599 non-null  object
 19  Unnamed: 20           0 non-null      float64
dtypes: float64(1), int64(1), object(18)
memory usage: 15.5+ MB
```

Figure 1: data.info() result

The untouched dataset contains 20 columns in total with 1 column of floating-point numbers, 1 column of integers, and 18 columns of objects. Such a high number of object data type columns was highlighted here as the csv file seemingly contained a higher number of numeric columns. The columns: Units, Channel, Comment, Unit Price, and Unnamed: 20 contained rows with missing values, as seen in their Non-Null Count being lower than the number of total entries, 81228, stated at the top of the table. Unnamed: 20 in particular stood out because it had a non-null count of 0, meaning that the entire column had nothing in it. The Comment column was also nearly empty, containing only 723 non-null cells.

To further investigate the raw dataset, a for loop (see Figure 2) displayed the column names, the number of unique numbers within the column, and the number of cells with missing values. The customer column contained 3112 unique values out of a total of 81228 with 0 empty cells. There are 45 different sales reps that make up the full column. Total Quantity has 1375 unique values and Total value sale has 26529. Other noteworthy columns are Units, with 5767 empty cells, Channel with 48, Comment with 80505, Unit Price with 629, and Unnamed: 20 with 81228 empty cells.



```
column_names = data.columns
for c in column_names:
    print( c, ':', data[c].nunique(), data[c].isna().sum() )
```

```
Customer : 3112 0
Sales Rep : 45 0
Month : 12 0
Year : 3 0
Date : 28 0
Code and description  : 1136 0
Code Prod : 723 0
Description : 1328 0
Quantity : 1375 0
Units : 7 5767
Total quantity : 1264 0
Value Sale : 28467 0
Tax : 22865 0
Total value sale : 26529 0
Channel : 15 48
Brand : 62 0
Category : 14 0
Comment : 14 80505
Unit price : 9339 629
Unnamed: 20 : 0 81228
```

Figure 2: the column names, unique numbers, and empty cells for loop

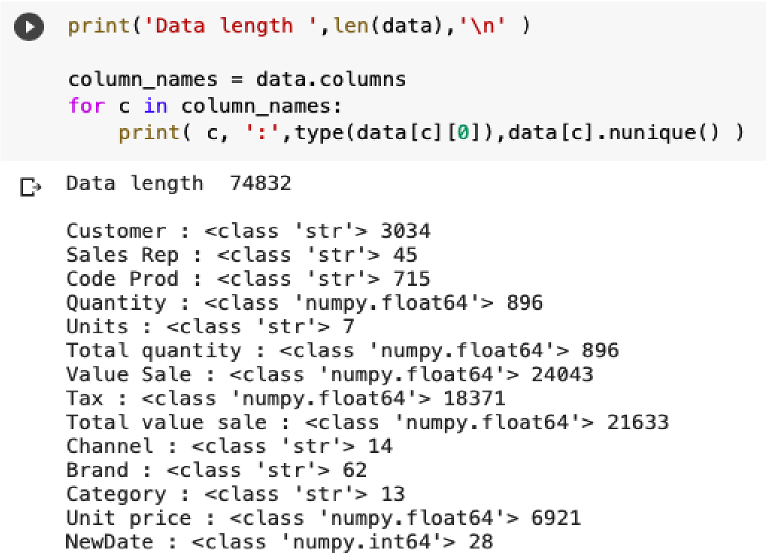
While unclear from the two previous figures, the Total value sale column contained cells with numbers surrounded by parentheses, and several other columns contained numbers that were separated by commas. It is worth highlighting this punctuation here as it is part of the raw data, how it and other cleaning of the data is explained in the following section.

C. Data Cleaning

Significant data cleaning was required to make the dataset workable within the contexts required for modelling.

The first step in cleaning was simply removing both the Unnamed: 20 and the Comment columns using the data.drop method. Unnamed: 20 was dropped because it contained nothing. Comment was dropped because it was mostly empty and the cells that did have data in them contained nothing useful.

As previously stated, Total value sale contained cells with parentheses. To remove them, the .replace method was used, replacing either (or) with “” empty quotation marks. This method removed the parentheses without putting anything in their place. Doing this also meant that what was previously an object column could be turned into a numeric one.



```
print('Data length ',len(data),'\n' )

column_names = data.columns
for c in column_names:
    print( c, ':',type(data[c][0]),data[c].nunique() )
```

```
Data length  74832

Customer : <class 'str'> 3034
Sales Rep : <class 'str'> 45
Code Prod : <class 'str'> 715
Quantity : <class 'numpy.float64'> 896
Units : <class 'str'> 7
Total quantity : <class 'numpy.float64'> 896
Value Sale : <class 'numpy.float64'> 24043
Tax : <class 'numpy.float64'> 18371
Total value sale : <class 'numpy.float64'> 21633
Channel : <class 'str'> 14
Brand : <class 'str'> 62
Category : <class 'str'> 13
Unit price : <class 'numpy.float64'> 6921
NewDate : <class 'numpy.int64'> 28
```

Figure 3: New data length, class, and unique values for loop

Numeric columns were defined in this section. They included the columns Quantity, Total Quantity, Value Sale, Tax, Total value sale, and Unit price. These were put into a list named numeric_colums. In this step, we further enhanced the data by modifying the Month, Date and Year columns that were problematic and written in Spanish, to create a NewDate column.

To ensure no empty cells remained in the dataset, the method .dropna() was implemented to remove any rows that contained an empty cell. While this method may seem like quite an extreme measure, it ensures that no empty cells are left to interfere with the modelling process. The risk associated with using this method is that it would remove too many rows and make the dataset too small to be useful, however, as seen in Figure 3, the updated data length after cleaning is 74832 which is still considered a large enough dataset to be useful.

A for loop that went through the previously defined numeric columns was the final step in the data cleaning process. The for loop first used the .replace method to remove commas from the numbers within the numeric columns. This means the only punctuation that could appear within these columns was a decimal point. This step was necessary because without it the following method would not work. Most of the numeric columns were classed as an object data type which does not work very well in many modelling scenarios. The for loop then converted the numeric columns to numbers using the pd.to_numeric method.

A final check of the data (see Figure 3) revealed a data length of 74832 and with the defined numeric columns being numeric data types, specifically floating-point numbers.

D. Pre-processing Data

This analysis of the data set used both Total quantity and Total value sale as the basis for their test-train split to investigate two aspects of the business problem. First, the data was split into 80% training and 20% testing, meaning the training data size was (59865, 17). Separation of the training data into lists of category columns and number columns allowed for the creation of corresponding pipelines. These pipelines were then brought together to create the data_prepared variable which has a size of (59865, 3752) with a label_train of (59865,). It was understood that the test-train split has been successful because the value of 59865 is the same across the training data size, data_prepared, and the label_train. The dataset is now ready to be processed.

V. Model Selection and Results

Fig.4 shows the machine learning models that were used in different supply chain areas. Our paper is focused on planning, which is a more popular algorithm used is Neural Network; however, we chose different models to investigate our data with: Linear Regression, Decision Tree, Random Forest. Support Vector Machines and Neural Networks were excluded from this investigation for the following reasons.

Even when the Support Vector Machine is a supervised learning method that can be used for regression problems, it is mostly suitable for handling high-dimensional, non-linear classification problems. (Bousqaoui, Achchab and Tikito, 2017)

Similarly, Neural Networks can be a powerful alternative tool and a complement to statistical techniques when data are multivariate with a high degree of interdependence between factors, when the data is noisy or incomplete, or when many hypotheses are to be pursued and high computational rates are required. (Hakimpoor et al., 2011)

Regarding this organisation’s particular scenario, their planning process in the organisation is manual and involves excel formulas only. So, using a sophisticated model seems to be the next step after maximizing the current data without overspending in models with high degree of computational rates required. Moreover, the data history is limited in the amount of data points (<100,000) and features (<700). Even when there is no hard-and-fast rule in determining the sample size for training neural networks, those numbers are the ones mentioned in class as reference and our database is below them.

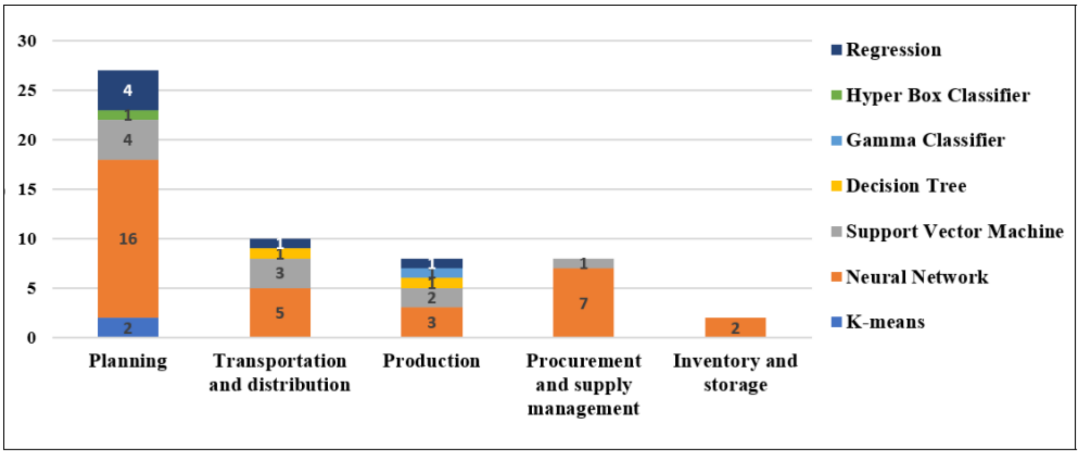


Fig. 4 The machine learning algorithms applied in the supply chain (Bousqaoui, Achchab and Tikito, 2017)

A. Results

Total Quantity Results		
Model	RMSE Score	R^2 Score
Linear Regression Train	0.000	1.000
Linear Regression Test	0.0006	1.000
Decision Tree Train	19.6554	0.999
Decision Tree Test	2.7071	0.999
Random Forest Train	24.7744	0.981
Random Forest Test	2.9258	0.999

Total Value Sale Results		
Model	RMSE Score	R^2 Score
Linear Regression Train	0.000	1.000
Linear Regression Test	0.1877	1.000
Decision Tree Train	68.5569	0.998
Decision Tree Test	42.9018	0.999
Random Forest Train	372.5264	0.985
Random Forest Test	18.9364	1.000

Table 1: RMSE Score and R^2 with a random 80/20 split

While Table 1 is interesting because it shows that the models have an apparent near perfect prediction rate when the data is randomly split 80% training and 20% testing, these results have limited use, especially in relation to the business case. Therefore, more analysis was needed so a manager could gain actionable information from the modelling. The following section explains how this was achieved.

B. Model evaluation

To apply the models in a meaningful way, the data was organised by the month in which the sale was made (see Appendix A). There was a total number of 28 months included in the data set. Four separate tests were performed for both Total Quantity and Total Value Sale to attempt to test how the models would cope under different circumstances. Month 28 was incomplete and contained significantly fewer variables when compared to the other 27 months. In the first two tests it was included as the final month of the series, while in the second two tests it was left out so only complete months would be assessed. The last 3 or 6 months were used as the test set and the remaining months were used at the training set. The error was determined by comparing the actual results from the chosen time frame to the predicted results.

Total Quantity				
Test Months	Total Test Transactions	Model	Number of Errors	Mean Errors Per Month
Last 3 Months #s 26, 27, 28	6023	Linear	348	116
		Decision Tree	40	13.33
		Random Forest	991	330.33
Last 6 Months #s 23, 24, 25, 26, 27, 28	16183	Linear	586	97.67
		Decision Tree	157	26.17
		Random Forest	1607	267.83
Last 3 Full Months #s 25, 26, 27	8571	Linear	5	1.67
		Decision Tree	74	24.67
		Random Forest	1465	488.33
Last 6 Full Months #s 22, 23, 24, 25, 26, 27	19118	Linear	56	9.33
		Decision Tree	182	30.33
		Random Forest	2194	365.67

Table 2: Total Quality Model Assessment Per Defined Months

Total Value Sale				
Test Months	Total Test Transactions	Model	Number of Errors > 1$	Mean Errors > 1$ Per Month
Last 3 Months #s 26, 27, 28	6023	Linear	32	10.67
		Decision Tree	123	41
		Random Forest	2284	761.33
Last 6 Months #s 23, 24, 25, 26, 27, 28	16183	Linear	70	11.67
		Decision Tree	383	63.83
		Random Forest	6435	1072.5
Last 3 Full Months #s 25, 26, 27	8571	Linear	0	0
		Decision Tree	167	55.67
		Random Forest	3560	1186.67
Last 6 Full Months #s 22, 23, 24, 25, 26, 27	19118	Linear	377	62.83
		Decision Tree	429	71.5
		Random Forest	6922	1165.33

Table 3: Total Value Sale Model Assessment Per Defined Months

Total Quantity was assessed on the total number of errors within the given time period and the mean errors per month (see Table 2). Total Value Sale was assessed by the number of errors that were greater than 1 dollar and the mean errors greater than 1 dollar per month (see Table 3). The value of greater than one dollar (>1$) was chosen because of the nature of the data itself; making the value equal to 0 skewed the data too much as the model counted almost every prediction as an error. The >1$ value ensures any significant difference between actual and predicted values is included, while excluding insignificant values with multiple decimal places.

The models considered to be the most accurate predictors have the lowest number of errors and lowest mean errors, these are highlighted in grey in both Table 2 and 3.

VI. Business Implications

A. Strategic Decisions

Thanks to our analysis, several strategic decisions were identified that would benefit management in making organisational decisions. The most accurate models were Linear and ideally the models should exclude incomplete months, this gives a manager the best predictions. If incomplete months are included there are limited effects in Total Value Sale, however, in Total Quantity incomplete months make the Decision Tree the best model to use. Management should be clear when choosing the range of the data as it can have a significant impact on the reliability of the predictions. Which models to choose and how they are applied needs to depend on context, otherwise mistakes can be made, especially in relation to forecasting expected orders or profits. Using this ideal model means that inventory can be ordered in advance with a reasonable degree of certainty that it will be ordered by customers. However, the most important contribution would be building a model that ideally involves collaborative forecasting. This means getting and using additional information from down the supply chain, not only the focal company but suppliers and customers.

Choosing the right model and applying it in the correct context can lead to better forecasting, however, a manager must not forget that it is only a forecast and that it may never be perfect. When making purchasing decisions, the models should be considered, however, other factors, such as special events, should also be considered.

Managers from different teams within the company, logistics and sales for example, should come together, analyse, and scrutinise the predictions to make the best decision. There is little point in having sales pre-emptively ordering inventory without consulting with logistics to ensure supply chains and warehousing are not overwhelmed.

Modelling, historical knowledge, and collaboration should be used in tandem to help ensure good strategic decision making.

VII. Conclusion

A. Summary of results

Machine learning is an inherently human involved process, if the right data and models are applied correctly then there are few limits to the scope of prediction. If a mistake is made, however, all you would have is meaningless numbers. Context matters; due to the underdevelopment of the company's demand prediction processes, it makes sense to recommend systematizing the process as a first step even by enabling models that focus on two different labels for two different business managers. Having created models contributes to the efficiency of the current planning process, automatizing the way both managers - sales and logistics - calculate and take their business decisions. However, it does not automatize the complete process. The recommended next step might be then to build a model - a more complex one such as a Support Vector Machine or Neural network - that makes a unique future prediction minimizing the time invested in meetings to determine this forecast.

B. Limitations and further research

This project was not without its limitations. The dataset, which the team was dependent on to make the decisions and conduct the analysis on was quite limited, as it only had slightly over two years' worth of data, which was not recent and in a way does not reflect the world that we live in today, following on from the coronavirus pandemic. If given the opportunity, having a larger data set that was gathered over a longer period of time would have helped give a more accurate prediction. It would also have been an interesting opportunity to compare the data before and during the coronavirus pandemic, and how the organisation predictions have dealt with the unexpected forks in the road.

Further research could be conducted on investigating multiple other issues, such as uncertainty, randomness, seasonality, and the bullwhip effect, to give a more in-depth analysis to the available data and give managers the opportunity to make predictions with unique circumstances. This again was hindered by the lack of data as, for example, the data only had two Decembers, so predicting the Christmas period would have been largely inaccurate with such a lack of data.

This data would also have benefitted from a wider use of different models, as they could have not only yielded different results, but have given clearer directions in predicting the demand. If done again, further models would have been used to improve the accuracy of the forecasts.

Lastly, although our team was lucky in obtaining internal data, it is somewhat limited, as there could have been a much richer analysis conducted on the data if there was a supply chain integration between the organisation, their providers, and customers, in real time. This shared data would have been much more accurate in making demand predictions.

Overall, there are limitations with the data presented and the models used, there are many opportunities to further improve and research this dataset with other unique models that can give us a clearer picture of this organisation's demand and make more accurate predictions for supply levels.

VIII. Bibliography

Bousqaoui, H, Achchab, S. and Tikito, K., (2017) Machine Learning applications in supply chains: An emphasis on neural network applications. *2017 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech)*, [online] Available at: <http://10.1109/CloudTech.2017.8284722>

Carbonneau, R., Laframboise, K. & Vahidov, R., (2008) Application of machine learning techniques for supply chain demand forecasting. *European Journal*

Center, I. T. J. W. R., (2019) *The Role of Machine Learning in Business Optimization,* New York: Yorktown Heights.

Fujimaki, R., Muraoka, Y., Ito, S. & Yabe, A., (2016) From prediction to decision making - Predictive optimization technology. *NEC Technical Journal.*

Géron, A. (2019) *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. 2nd edn. O'Reilly Media, Inc. *l of Operational Research,* 184(3), pp. 1140-1154.

Hakimpoor, H., et al. (2011) Artificial Neural Networks' Applications in Management. *World Applied Sciences Journal*, [online] 14(7), pp.1008-1019. Available at: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.389.6419&rep=rep1&type=pdf>

Islam, S. & Amin, S. H., (2020) Prediction of probable backorder scenarios in the supply chain using Distributed Random Forest and Gradient Boosting Machine learning techniques. *Journal of Big Data,* Issue 7.

Tarallo, E. et al., (2019) Machine Learning in prediction demand for Fast-Moving Consumer Goods: An exploratory research. *ScienceDirect,* 52(13), pp. 737-742.

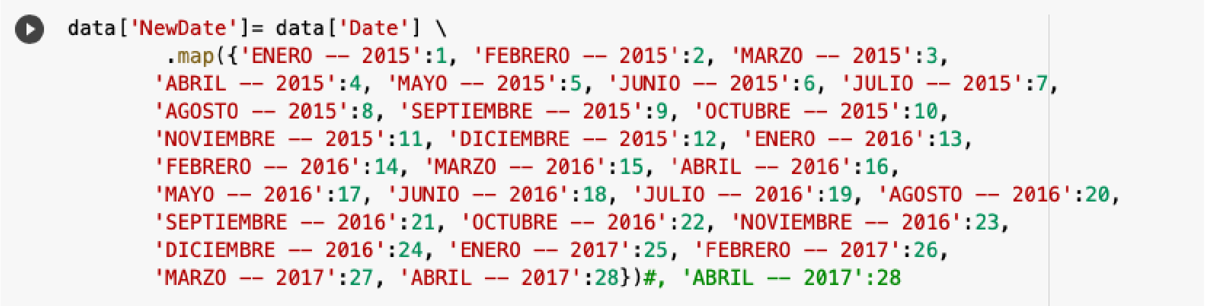
IX. Appendices

Appendix A: Overview of Features

N°	Feature	Data Type	Description
1	**Customer code**	Categorical	The customer’s name encoded
2	**Customer**	Categorical	The customer’s name
3	**Sales Rep**	Categorical	The sales representative who did the sale
4	**Month**	Categorical	The month when the sales was done
5	**Year**	Numeric	The year when the sales was done
6	**Date**	Categorical	The month and year when the sale was done
7	**Code and description**	Categorical	The detail of product code and its description
8	**Code Prod**	Categorical	Product code
9	**Description**	Categorical	Description of the product
10	**Quantity**	Numeric	Single order quantity
11	**Units**	Numeric	The logistic unit or product presentation/format, like bottle or box
12	**Total quantity**	Numeric	The total amount of single product sold
13	**Value Sale**	Numeric	The cost without taxes

14	**Tax**	Numeric	The amount of money attributed to taxes
15	**Total value sale**	Numeric	The total amount of money received
16	**Channel**	Categorical	The customer type the product was sold to
17	**Brand**	Categorical	The brand of the product sold
18	**Category**	Categorical	The kind of drink sold
19	**Comment**	Categorical	Any comment in relation to the sale
20	**Unit price**	Numeric	The unit price of the sale, this is usually tracked to understand price increases over time

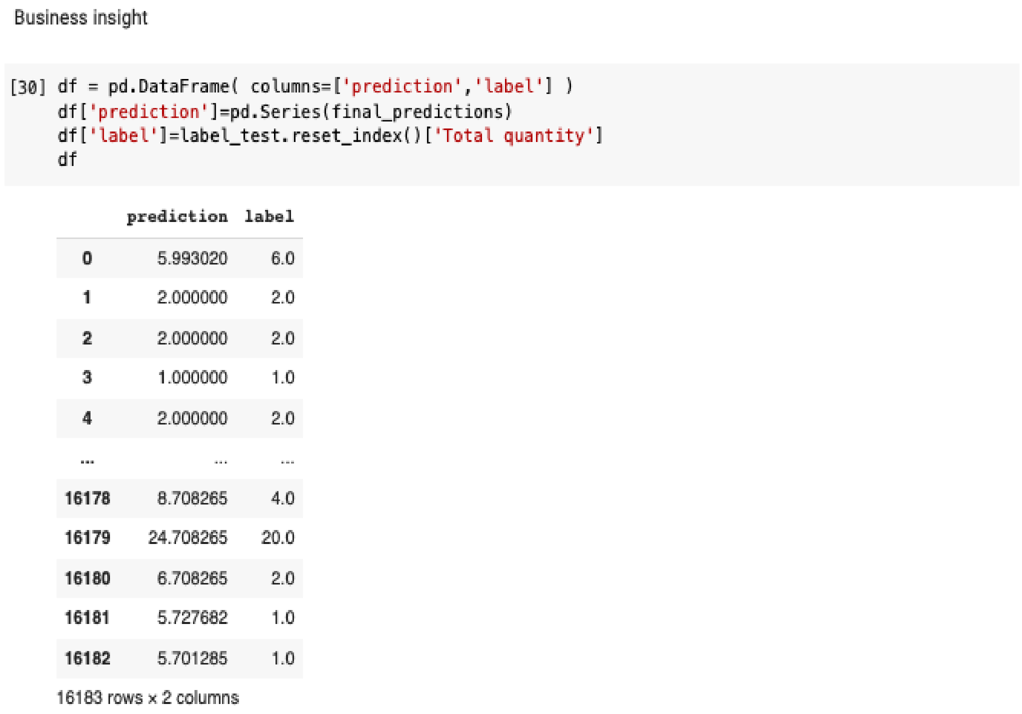
Appendix B: Organised By Month



```
data['NewDate']= data['Date'] \
        .map({'ENERO -- 2015':1, 'FEBRERO -- 2015':2, 'MARZO -- 2015':3,
       'ABRIL -- 2015':4, 'MAYO -- 2015':5, 'JUNIO -- 2015':6, 'JULIO -- 2015':7,
       'AGOSTO -- 2015':8, 'SEPTIEMBRE -- 2015':9, 'OCTUBRE -- 2015':10,
       'NOVIEMBRE -- 2015':11, 'DICIEMBRE -- 2015':12, 'ENERO -- 2016':13,
       'FEBRERO -- 2016':14, 'MARZO -- 2016':15, 'ABRIL -- 2016':16,
       'MAYO -- 2016':17, 'JUNIO -- 2016':18, 'JULIO -- 2016':19, 'AGOSTO -- 2016':20,
       'SEPTIEMBRE -- 2016':21, 'OCTUBRE -- 2016':22, 'NOVIEMBRE -- 2016':23,
       'DICIEMBRE -- 2016':24, 'ENERO -- 2017':25, 'FEBRERO -- 2017':26,
       'MARZO -- 2017':27, 'ABRIL -- 2017':28})#, 'ABRIL -- 2017':28
```

Appendix C: Linear Regression

I. Total Quantity

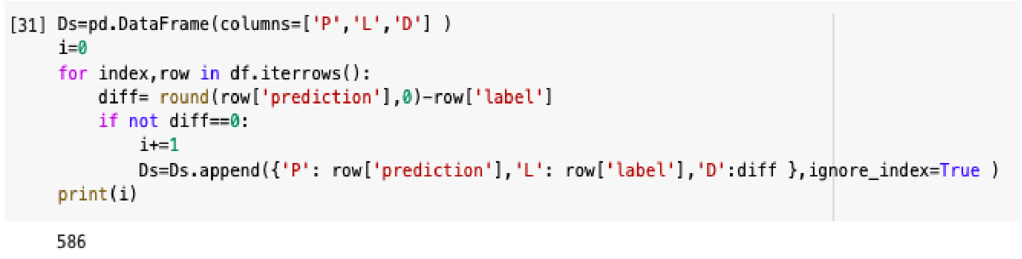


Business insight

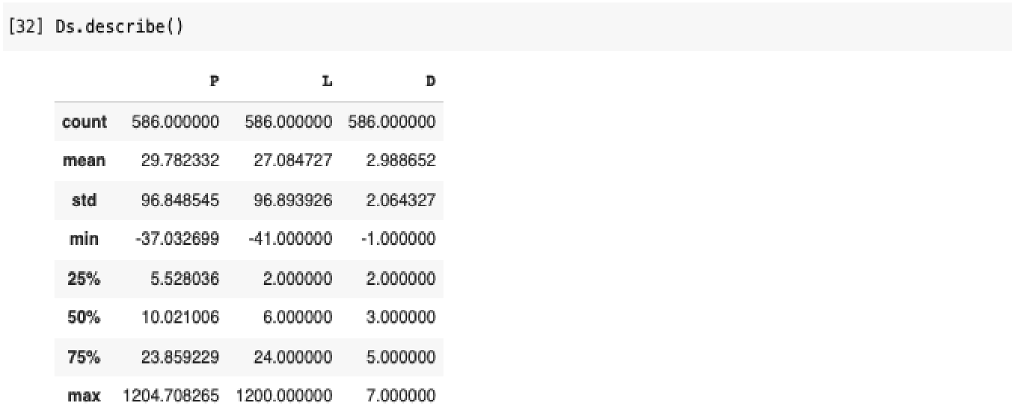
```
[30] df = pd.DataFrame( columns=['prediction','label'] )
     df['prediction']=pd.Series(final_predictions)
     df['label']=label_test.reset_index()['Total quantity']
     df
```

	prediction	label
0	5.993020	6.0
1	2.000000	2.0
2	2.000000	2.0
3	1.000000	1.0
4	2.000000	2.0
...	...	...
16178	8.708265	4.0
16179	24.708265	20.0
16180	6.708265	2.0
16181	5.727682	1.0
16182	5.701285	1.0

16183 rows × 2 columns

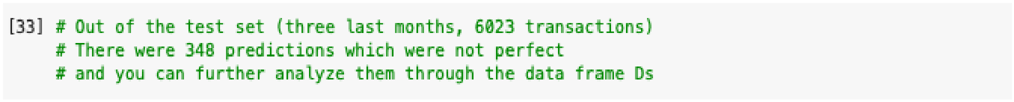


```
[31] Ds=pd.DataFrame(columns=['P','L','D'] )
     i=0
     for index,row in df.iterrows():
         diff= round(row['prediction'],0)-row['label']
         if not diff==0:
             i+=1
             Ds=Ds.append({'P': row['prediction'],'L': row['label'],'D':diff },ignore_index=True )
     print(i)

     586
```



```
[32] Ds.describe()
```

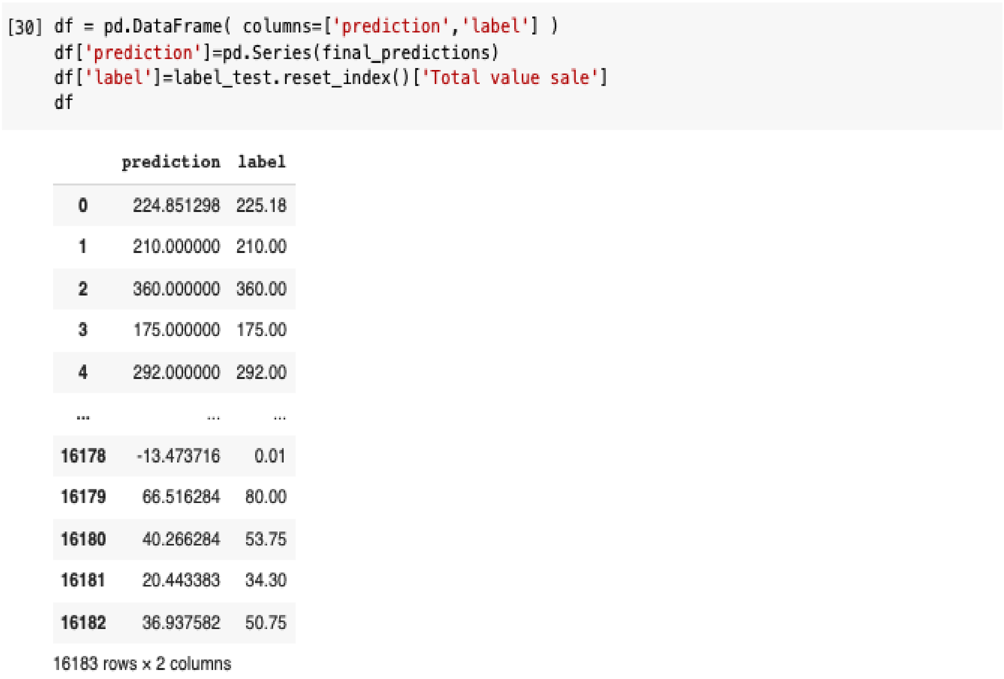
	P	L	D
count	586.000000	586.000000	586.000000
mean	29.782332	27.084727	2.988652
std	96.848545	96.893926	2.064327
min	-37.032699	-41.000000	-1.000000
25%	5.528036	2.000000	2.000000
50%	10.021006	6.000000	3.000000
75%	23.859229	24.000000	5.000000
max	1204.708265	1200.000000	7.000000



```
[33] # Out of the test set (three last months, 6023 transactions)
     # There were 348 predictions which were not perfect
     # and you can further analyze them through the data frame Ds
```

II. Total Value Sale

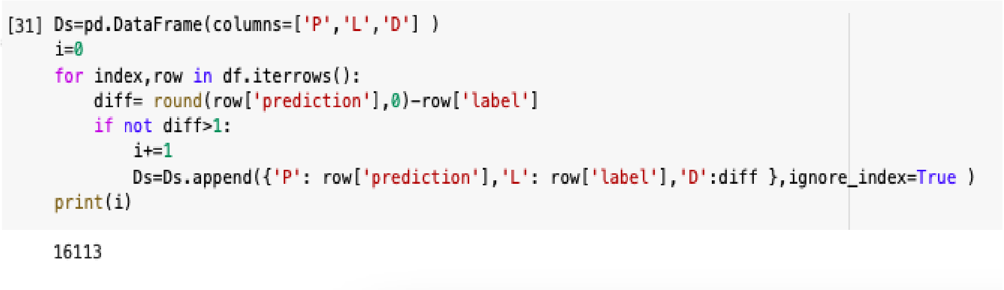
Business insight



```
[30] df = pd.DataFrame( columns=['prediction','label'] )
     df['prediction']=pd.Series(final_predictions)
     df['label']=label_test.reset_index()['Total value sale']
     df
```

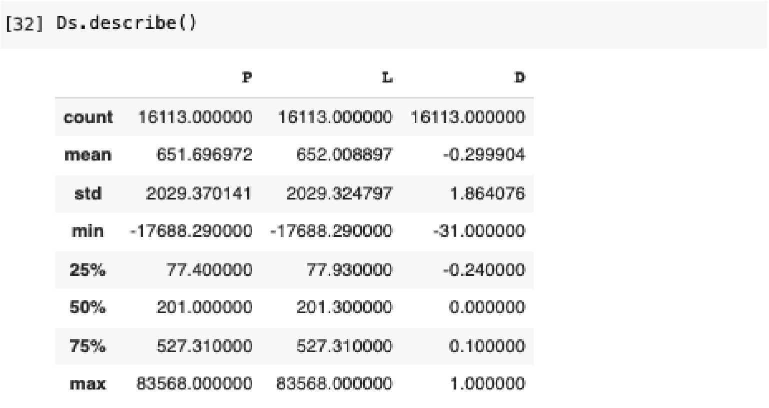
	prediction	label
0	224.851298	225.18
1	210.000000	210.00
2	360.000000	360.00
3	175.000000	175.00
4	292.000000	292.00
...	...	...
16178	-13.473716	0.01
16179	66.516284	80.00
16180	40.266284	53.75
16181	20.443383	34.30
16182	36.937582	50.75

16183 rows × 2 columns



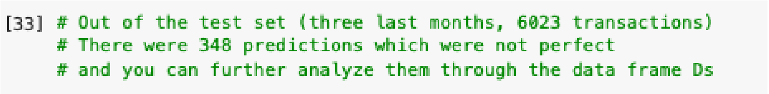
```
[31] Ds=pd.DataFrame(columns=['P','L','D'] )
     i=0
     for index,row in df.iterrows():
         diff= round(row['prediction'],0)-row['label']
         if not diff>1:
             i+=1
             Ds=Ds.append({'P': row['prediction'],'L': row['label'],'D':diff },ignore_index=True )
     print(i)
```

```
16113
```



```
[32] Ds.describe()
```

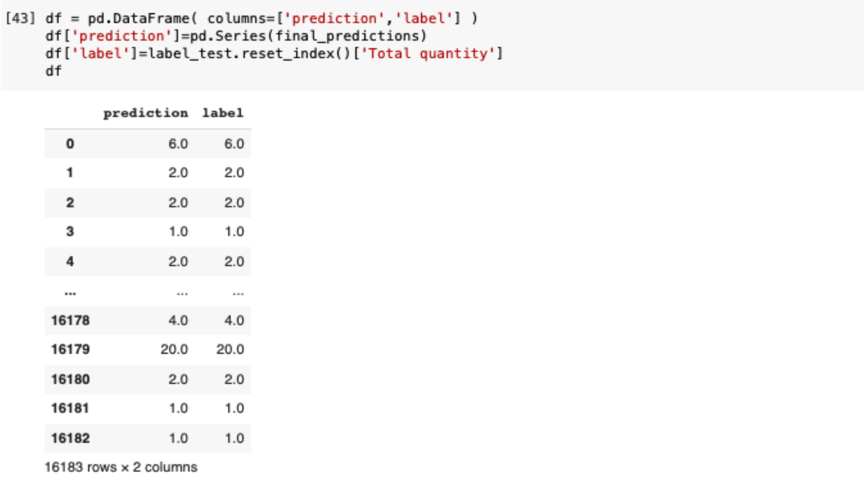
	P	L	D
count	16113.000000	16113.000000	16113.000000
mean	651.696972	652.008897	-0.299904
std	2029.370141	2029.324797	1.864076
min	-17688.290000	-17688.290000	-31.000000
25%	77.400000	77.930000	-0.240000
50%	201.000000	201.300000	0.000000
75%	527.310000	527.310000	0.100000
max	83568.000000	83568.000000	1.000000



```
[33] # Out of the test set (three last months, 6023 transactions)
     # There were 348 predictions which were not perfect
     # and you can further analyze them through the data frame Ds
```

Appendix D: Decision Tree

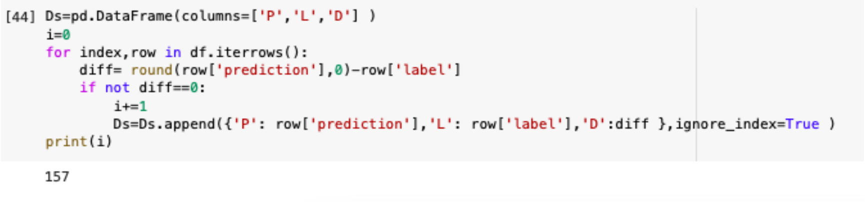
I. Total Quantity



```
[43] df = pd.DataFrame( columns=['prediction','label'] )
     df['prediction']=pd.Series(final_predictions)
     df['label']=label_test.reset_index()['Total quantity']
     df
```

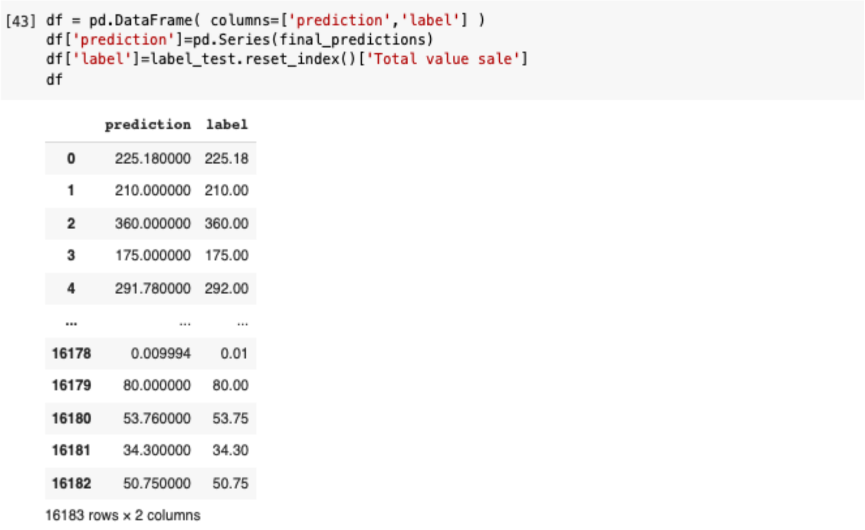
	prediction	label
0	6.0	6.0
1	2.0	2.0
2	2.0	2.0
3	1.0	1.0
4	2.0	2.0
...	...	...
16178	4.0	4.0
16179	20.0	20.0
16180	2.0	2.0
16181	1.0	1.0
16182	1.0	1.0

16183 rows × 2 columns



```
[44] Ds=pd.DataFrame(columns=['P','L','D'] )
     i=0
     for index,row in df.iterrows():
         diff= round(row['prediction'],0)-row['label']
         if not diff==0:
             i+=1
             Ds=Ds.append({'P': row['prediction'],'L': row['label'],'D':diff },ignore_index=True )
     print(i)

     157
```

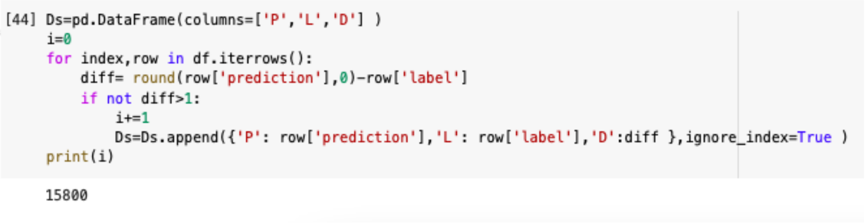
II. Total Value Sale



```
[43] df = pd.DataFrame( columns=['prediction','label'] )
     df['prediction']=pd.Series(final_predictions)
     df['label']=label_test.reset_index()['Total value sale']
     df
```

	prediction	label
0	225.180000	225.18
1	210.000000	210.00
2	360.000000	360.00
3	175.000000	175.00
4	291.780000	292.00
...	...	...
16178	0.009994	0.01
16179	80.000000	80.00
16180	53.760000	53.75
16181	34.300000	34.30
16182	50.750000	50.75

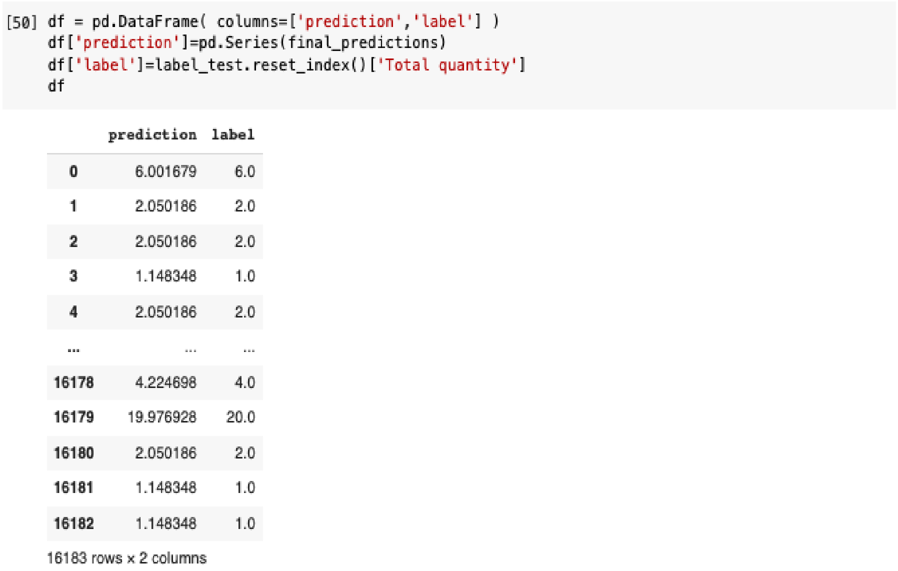
16183 rows × 2 columns



```
[44] Ds=pd.DataFrame(columns=['P','L','D'] )
     i=0
     for index,row in df.iterrows():
         diff= round(row['prediction'],0)-row['label']
         if not diff>1:
             i+=1
             Ds=Ds.append({'P': row['prediction'],'L': row['label'],'D':diff },ignore_index=True )
     print(i)

     15800
```

Appendix E: Random Forest

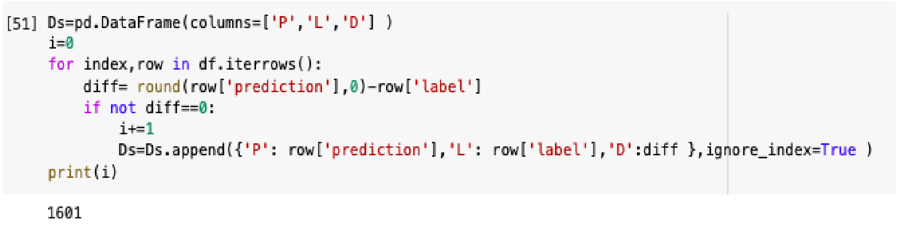
I. Total Quantity



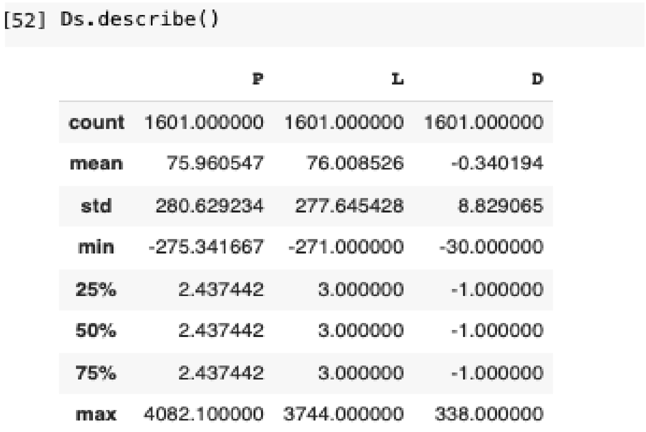
```
[50] df = pd.DataFrame( columns=['prediction','label'] )
     df['prediction']=pd.Series(final_predictions)
     df['label']=label_test.reset_index()['Total quantity']
     df
```

	prediction	label
0	6.001679	6.0
1	2.050186	2.0
2	2.050186	2.0
3	1.148348	1.0
4	2.050186	2.0
...	...	...
16178	4.224698	4.0
16179	19.976928	20.0
16180	2.050186	2.0
16181	1.148348	1.0
16182	1.148348	1.0

16183 rows × 2 columns



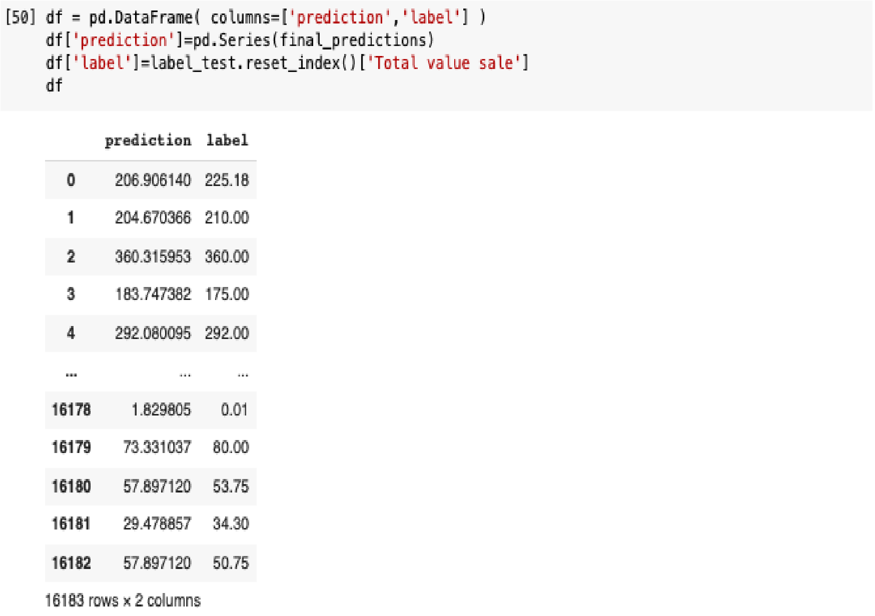
```
[51] Ds=pd.DataFrame(columns=['P','L','D'] )
     i=0
     for index,row in df.iterrows():
         diff= round(row['prediction'],0)-row['label']
         if not diff==0:
             i+=1
             Ds=Ds.append({'P': row['prediction'],'L': row['label'],'D':diff },ignore_index=True )
     print(i)

     1601
```



```
[52] Ds.describe()
```

	P	L	D
count	1601.000000	1601.000000	1601.000000
mean	75.960547	76.008526	-0.340194
std	280.629234	277.645428	8.829065
min	-275.341667	-271.000000	-30.000000
25%	2.437442	3.000000	-1.000000
50%	2.437442	3.000000	-1.000000
75%	2.437442	3.000000	-1.000000
max	4082.100000	3744.000000	338.000000

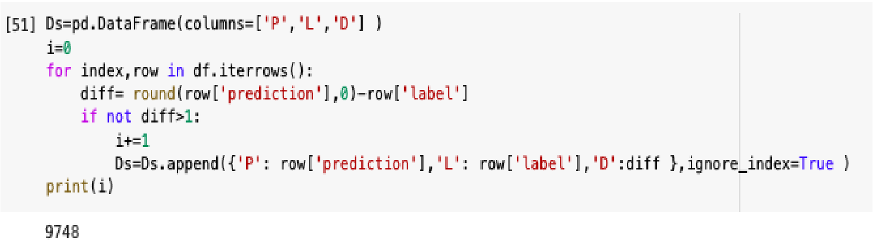
II. Total Value Sale



```
[50] df = pd.DataFrame( columns=['prediction','label'] )
     df['prediction']=pd.Series(final_predictions)
     df['label']=label_test.reset_index()['Total value sale']
     df
```

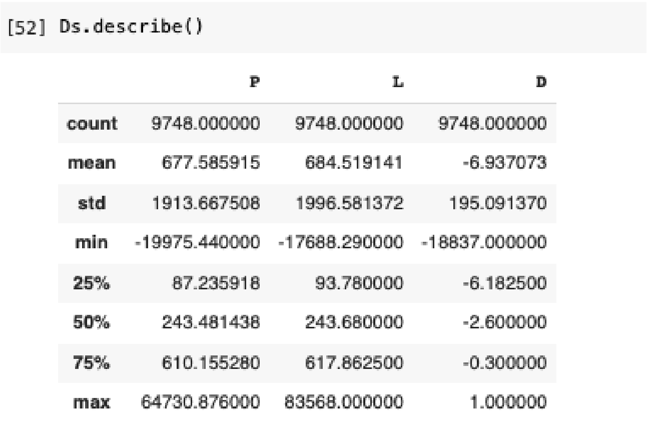
	prediction	label
0	206.906140	225.18
1	204.670366	210.00
2	360.315953	360.00
3	183.747382	175.00
4	292.080095	292.00
...	...	...
16178	1.829805	0.01
16179	73.331037	80.00
16180	57.897120	53.75
16181	29.478857	34.30
16182	57.897120	50.75

16183 rows × 2 columns



```
[51] Ds=pd.DataFrame(columns=['P','L','D'] )
     i=0
     for index,row in df.iterrows():
         diff= round(row['prediction'],0)-row['label']
         if not diff>1:
             i+=1
             Ds=Ds.append({'P': row['prediction'],'L': row['label'],'D':diff },ignore_index=True )
     print(i)
```

```
9748
```



```
[52] Ds.describe()
```

	P	L	D
count	9748.000000	9748.000000	9748.000000
mean	677.585915	684.519141	-6.937073
std	1913.667508	1996.581372	195.091370
min	-19975.440000	-17688.290000	-18837.000000
25%	87.235918	93.780000	-6.182500
50%	243.481438	243.680000	-2.600000
75%	610.155280	617.862500	-0.300000
max	64730.876000	83568.000000	1.000000