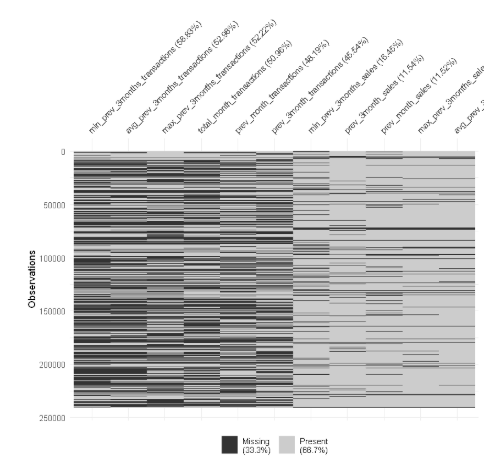
**Introduction**

In my Data Science project I want to forecast the number of unit sales per month by product name and by store number. In order to explore the ultimate model of prediction I used a dataset of Favorita stores located in Ecuador. There is a big motivation from groceries to have a model which forecasts the sales accurately. It comes from the big need of not being stuck with overstocked, perishable goods and products which finished their self-life and need to be destroyed. More forecasting systems are based on a statistical learning method like moving average, and regression which are reasonably effective in most cases but require heavy and delicate feature engineering. In the past, in order to predict unit sales Alexia Wenxin Xu, a researcher from Stanford proposed to use a residual neural network to output a predicted unit sales. She recognized that both neural and residual networks improved the model by decrease the dev loss but there was a big noise in the dataset (1). Another group tried to solve the challenge of forecasting the product sales by applying the convolutional neural network method in the context of time series data. (2) They also used the Favorita stores Dataset and found out that convolutional neural networks worked good at handling historical data and catching seasonality, trends, cycles, and irregular components. Using a CNN WaveNet, a sequence to sequence architecture in order to do sales forecasting was a very effective method of solving time series predicting problems. Another group tried to combine between the deep learning method and the machine learning method by using LSTM and combine it with lightGBM, the first one used in order to dig in the data and get information from while the second used for being with strong interpretability (3). The combination between these two models used to forecast the supply chain sales by comparing the implementation of other different models and discuss the advantages of the combined model as well. Similar to first two projects which are discussed above they also took the Favorita Grocery’s supply chain sales data set for running the combined model on. Their insight was that the combined model predicts the supply chain sales accurately and the interpretability of the predicted results was significant to improve many aspects such as production mode, price management and precise marketing of supply chain. In my project the outcome feature is defined by the total item number sales per month, by store number.

**Data**

The data that I used, [Corporación Favorita](http://www.corporacionfavorita.com/) Grocery, was taken from kaggle. The data is based on time series, originally on a time frames of one day, no external data was added. Every day at the train data represent total unit sales per day for a specific product in a specific store number, every store number has her own city. In order to reduce the amount of data which contained a train data of about 200 million rows I decided to calculate for each item number the total unit sales per month for each store number and by that I had an initial data which contained about 2 million rows, I decided to choose a threshold for minimum 50000 time appearance per item number in order to continue to the final prediction. My outcome variable was the total unit sales per month, per item number in a specific store number. Some of the variables contained only 1 possible result and as a conclusion had no any effect on the outcome variable, therefore I chose to remove them from the data. In order to add more variables and enrich the data I used some techniques. The most prevalent technique was using a one hat encoding in order to create a Boolean features, also an aggregate functions were used in order to get data from past months, to calculate the mean of the past 3 months sales, and to calculate the sum of unit sales, holidays and transaction per month, most of the aggregate functions were used in sql at the flat file generation phase, finally I created a table which contains 84 columns, some of them represents features which came from the original data, and the rest I created by using sql aggregate functions. My data exploration phase started by look at the behavior of each feature, numeric and categorical. For the numeric features I checked the distribution by using a density plot. For the numeric features I checked the possibility of doing a log transformation. Regarding the outcome feature, and features contain sales data from the past months, doing a log transformation caused them to have a normal distributions which contribute to improve the quality of the models and enables to use a linear regression model and using a Pearson correlation matrix. After that I created a descriptive statistic tables that show us the minimum, maximum, and the standard deviation parameters for each ratio scale features. Next step was creating a correlation matrix using Pearson correlation method, creating graph which shows us how strong is the correlations between the different variable and between each variable and the outcome feature. After looking at the features behavior I proceeded to examine the outliers and the missing values. Regarding the outliers I decided to convert outliers to NA in features which removing them didn’t have a large effect on the feature distribution. Regarding the missing values I decided to remove all the rows with missing values. First I ran a code which gave me a cluster of all the rows and columns contained missing values inside. At the heatmap cluster we can see a relatively big percent of cells colored in black (figure 1) so that we can conclude that there are many rows that at list half of them is missing value at the relevant variables and in addition to the fact that the percentage of the missing values is very low and the fact that all our missing values came appeared after removing the outliers from the data, I infer that remove this rows from the data wouldn’t make a big effect on the features behavior and distributions.

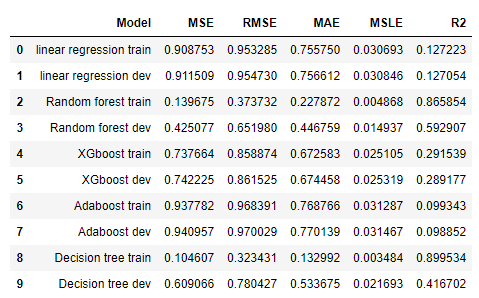
Figure 1- clustering the missing value



**Models**

Before starting running my models at the different metrics I divided the data into 3 pieces: train, dev and test. At first I divided the whole data into 2 pieces, temp and test with proportions of 90 and 10 percent accordingly. Secondly I took the temp data and divided it into another 2 pieces, dev and test with proportion of 90 and 10 percent accordingly. Finally after these 2 divisions the whole data we combined of 3 sub data: train, dev and test. In order to do a cluster analysis of my data I used an unsupervised learning techniques of kmeans clustering and principal component analysis (PCA). Because my outcome variable is a scale ratio variable I used a regression modes to evaluate the best prediction of the outcome. During running the models I checked couple of score parameters for each model. The models I checked were, Linear Repression, Random Forest regression tree, Ada Boost, XGboost and regular regression tree. The models fitting was checked on the train data and prediction made on both train and dev data. Finally I chose to continue with XGboost regression model because it was the most stable model with the best score. Regarding the regression models of regular decision tree and random forest they had better scores than XGBoost but there was a big difference between the train and dev when we look at the models scores by all metrics (table 1) .The parameters used to evaluate my model were: Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Squared Logarithmic Error (MSLE), R Square (R2) , because these metrics are very common used to evaluate regression models .The metric I selected to evaluate the model precision was Root Mean Square Error (RMSE) .The main reason I chose to use this method of measurement is that it’s a very common metric which used in regression models and this model gives us a good evaluation of how big is the prediction error when we talk about ratio scale features.

Table 1- regression models scores by all metrics.



**Results**

A final data of 1737726 rows was after cleaning into the model stage. This data wad divided to a 1407557 rows of train data, 156396 rows of dev and 173773 rows which assembled the test data. The initial data which has been created from the flatfile was assembled of 1975242 rows.

Outliers were found at some features and were converted to NA where removing them didn’t

change either the assumptions or the model (Table 2). Before treating the missing values their mechanism was checked and for all the features which had missing values the missing

mechanism was Missing Not At Random (MNAR). The missing value treated by removing all

rows had missing values after checking a visual cluster of the NA presence in the data (Table 3). A couple of techniques were used to either transform some of the existing variables or create a

new variables. The techniques which were used are: minmax transformation, logarithmic

transformation, binary transformation, one hat encoding and basic aggregation functions in order to calculate sum, average, count the number of similar value appearance. The partition of the

data were made by using the train test function takes from the package mechkar. After the data

partition has been made I checked that the distribution of each features are similar to each other

between all the 3 pieces. At the fine tuning phase the selected model (XGboost) was trained with a combination of hyperparamters and a cross validation of 3 folds and 20 iterations. After

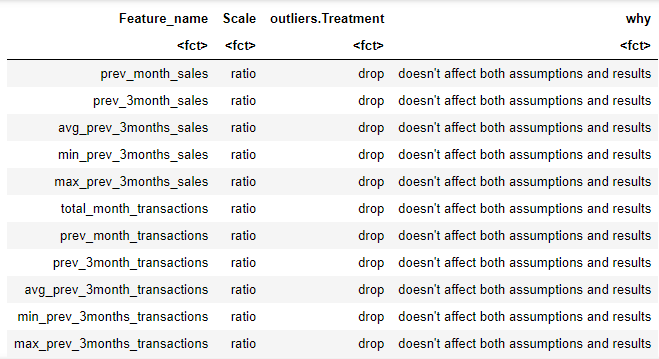
training the combination of hyperparamters and cross validation an accuracy rate of 88.58% has been measured, an improvement of 9.14% compare to the base model accuracy which was

81.16% (figure 3). While running the final selected model with the selected parameters, the

RMSE values for the train, dev, and test predictions were 0.568114, 0.583763 and 0.580874

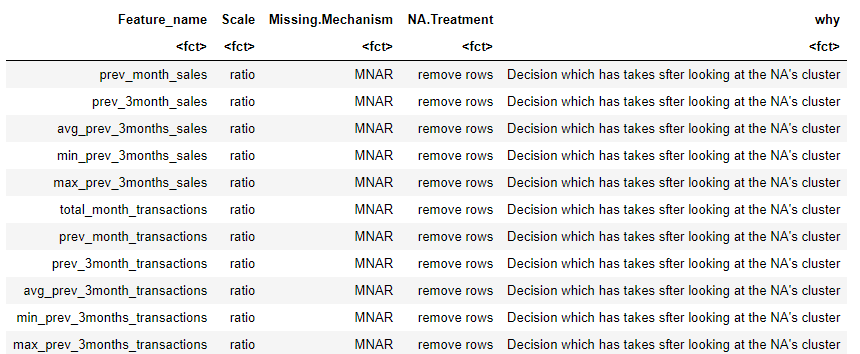
Accordingly (Table 4).

**Table 2- outliers treatment**

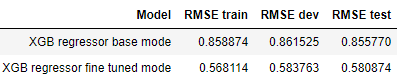


|  |
| --- |
|  |

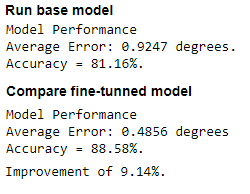
**Table 3 missing value treatment**



**Table 4 – evaluation result of the selected model**



**Figure 3 – accuracy results before and model fine tuning**

****

**Conclusions**

The project began by downloading the [Corporación Favorita](http://www.corporacionfavorita.com/) Grocery data from kaggle. While beginning to create the flatfile one of the main challenges was how I reduce the amount of rows and by that get a data size which would enable me to run the project scripts fluently without make my computer stuck. For that I chose to examine the total unit sales per month instead of total unit sales per day. Another big challenge was how I create a good feature combination which would enable me to get a good results at the feature selection strategy phase and then proceed to the next phase. The size of the data limited me for using the cross validation method widely and to use more iterations. One of the main things I would do differently is to look more on the scatter plots made at the phase of exploratory data analysis, and by paying attention to a problematical regression plots remove problematic columns from the data and by that prevent myself loosing important rows and ease the feature selection stage.

**References**

1. (Xu, A Generalizable Sales Forecasting Pipeline with Embedding and Residual n.d.)
2. (Kechyn, Yu, et al, 2018, Sales forecasting using WaveNet within the framework of the Kaggle competition)
3. <https://pdfs.semanticscholar.org/1b5f/d7be136793a213b91769344bdb4c72b99fe9.pdf>
4. (Glib Kechyn n.d.)<https://www.emerald.com/insight/content/doi/10.1108/IMDS-03-2019-0170/full/html?casa_token=Qs4fCrj6seEAAAAA:ZbkIoCtbQiU727yeZv2ZQ00U87aRDNPNXeQuNTOCtaDVWAF69k-OWVSsB3IOkaBP4c3PsmH7K4Ul_1ag9zeQ3tOFnRtoNlSuTj21RcloBMUDGWa4ERJH>