Machine Learning Engineer Nanodegree

Capstone Project

Jesus Martinez

Aug 22th, 2018

I. Definition

Project Overview

“La Petite” Jewelry opened its doors on the summer of 1979 in the city of Monterrey, Mexico. The founders saw an opportunity as they had a close relationship with Mexico’s most important jewelry manufacturers and owned well located commercial premises. They currently offer a great variety of products (both in silver and gold) ranging from rings, bracelets, neck chains (most sold product according to owner), earrings, etc.

Keeping track in how over a 1,000 different products are being sold on a daily basis can be a daunting task, especially if technology based solution have not being widely adopted. As it can be clear, the problems derived from a deficient monitoring of products being sold can be countless therefore we will only focus in trying to answer a simple question. **What quantities should the owner order from an specific product for maximizing utility purposes?**

Problem Statement

The question previously defined is asked every month by the owner in which using personal intuition and advice from in-house sales representatives an amount of product to be ordered is determined. As expected this process may be quite practical however not so accurate, as in average it deviates between 80% to 90% of exceeding product at the end of the month. Generating a more formal and accurate solution without affecting practicality will be the main goal.

This objective is planned to be achieved by modeling the data (captured from previously installed Point of Sale terminal) related to the product sold per day and applying machine learning algorithms in an effort to unveil purchasing patterns. Consequently, predictions will be generated and compared to standard metrics as well as quantifying any improvement from the current strategy.

Metrics

Standard widely used metrics were considered for this project to simplify evaluation. The **Mean Absolute Error** (MAE), a common measure of forecast error in time series analysis[4] will measure the difference between our predictions versus the true values.

Let n be the total number of data points marked as $y_1,...,y_n$ known as y_i associated with a predicted $\hat{y}_i$ therefore:

$$MAE = \frac{1}{n}\sum_{i}^{n} |y_i - \hat{y}_i|$$

On the other hand we will require a way to compare the accuracy performance between models while also comparing it to a basic model, being the mean of all the values within the target variable. This is where R^2 or **coefficient of determination** comes into place.

Defined by:

$$R^2 = 1 - SS_{res}/SS_{tot}$$

Where

$$SS_{res} = \sum_{i}(y_i - \hat{y}_i)^2$$

$$SS_{tot} = \sum_{i}(y_i - \bar{y})^2$$

and $\bar{y}$ representing the mean of the target variable.

Due to the nature of a regression problem, it is expected to have a variability between a number of trials or attempts, thus the following metrics were developed to compensate for these aspects:

Probable non-overfitting cases: percentage of events where $\frac{MAEtraining}{MAEtesting}$ is higher than 95% and aiding in defining how much confidence a certain model pertains.

Product quantity winning cases percentage (above 0% and below 80%): this metric basically measures the difference between a true and predicted value in order to understand whether there was a shortage or surplus within a given test set or a period of time, not to be confused with the Mean Absolute Error as it does not tell us if the prediction was over or underestimated.The upper limit aligns with the current benchmark where there is a minimum 80% average of product surplus ordered each month. Our aim is to have a higher percentage than the two situations described below.

Product quantity lost cases percentage below 0%: Least desired scenario as a shortage of product may result in a reduction of recurrent customers in the long term.

Product quantity lost cases percentage above 80%: No improvement has been achieved if a high percentage is encountered.

II. Analysis

Data Exploration

Our dataset included on the file “product_per_day.csv” consists of only 50 data points representing a particular day (starting in April 10th to May 29th, 2018) and the amount sold in grams for a single product line without any features established. General statistics were calculated and presented below.

count	50.000000
mean	51.962000
std	44.452249
min	0.000000
25%	13.600000
50%	46.350000
75%	74.775000
max	186.400000

The only abnormalities that were detected at this point relate to no product sold (minimum price = 0.0 gr) and possibly a slight difference between the mean and the median, 51.96 and 46.35 respectively, that may indicate a non-normal distribution.

Exploratory Visualization

The bar chart in Figure 1.0 displays the two sets of data available and previously defined, amount of product sold per day. Minimum value (0 gr) is repeated in various instances while the maximum value (186.4 gr) only once. It is important to mention that the latter represents a day before Mexico's Mother's Day.

This type of chart was chosen over the typically line chart used in time series analysis[5] in order to have a clear visualization of these extreme formerly stated quantities and their frequency.

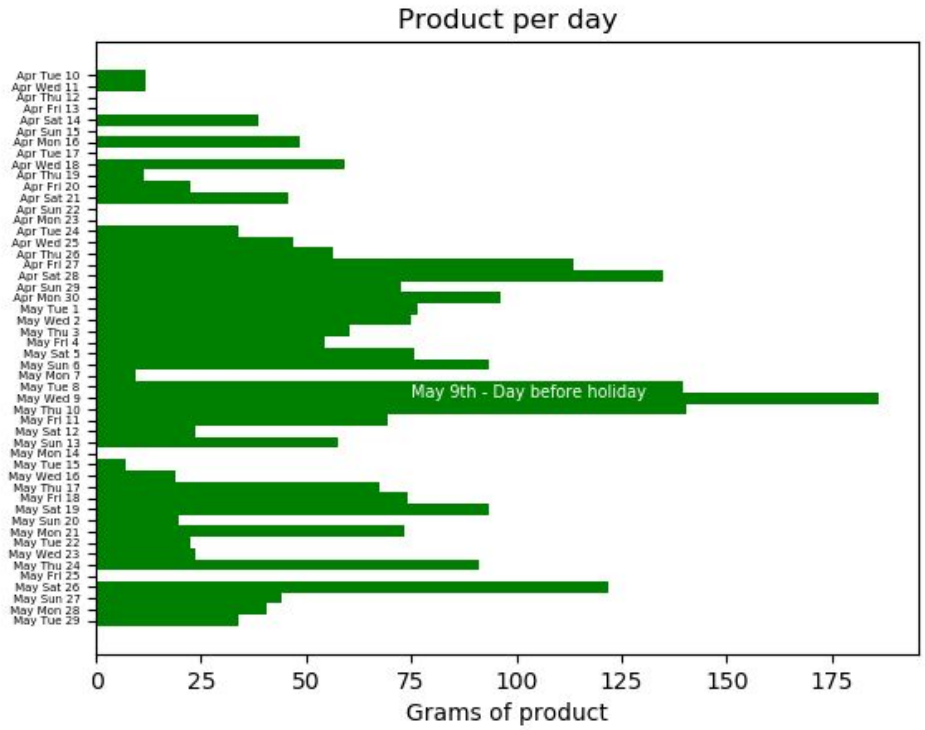


Fig 1.0 Grams sold per day

Algorithms and Techniques

As mentioned before, from the data set only a target variable and dates are available. For this reason, tasks related to feature engineering (creating new features) were considered as the "date" itself does not provide enough information for pattern revealing purposes. The features generated are listed below.

1. **'Days of Week'**: each day was assigned with a number ranging from 0 to 6 starting with Monday and ending in Sunday.
2. **'Holiday?**': appoints a '1' if a day falls within 2 days before a holiday, i.e. for May 10th (Mexican Mother's Day) May 8th , May 9th and May 10th will be equal to '1' and '0' if this statement is FALSE.
3. **'Weekend?'**: if a particular day falls on a weekend it will be set to '1' while the rest to '0'.

Moreover, the predictions were generated by training two regression machine learning algorithms (using as inputs the already discussed features an target variable) aiming to find an expected value based on certain characteristics.. The first one being Linear Regression, extensively studied and used in practical applications[6] hence its preference. The other, a so-called ensemble method named Gradient Boosting very alike to the widely employed XGBoost and favored by many winning teams in these types of competitions[2].

Benchmark

Existing methods within the business correlate with a conservative strategy avoiding in most situations any product shortage. No data is currently being evaluated and new product orders are placed based on experience, intuition and current inventory. By analysing inventory records and product orders, a surplus per month average was calculated being in the range of 80% to 90%.

III. Methodology

Data Preprocessing

During this phase two main modifications within the data were performed. The chart in Fig 3.0 below clearly portrays these changes.

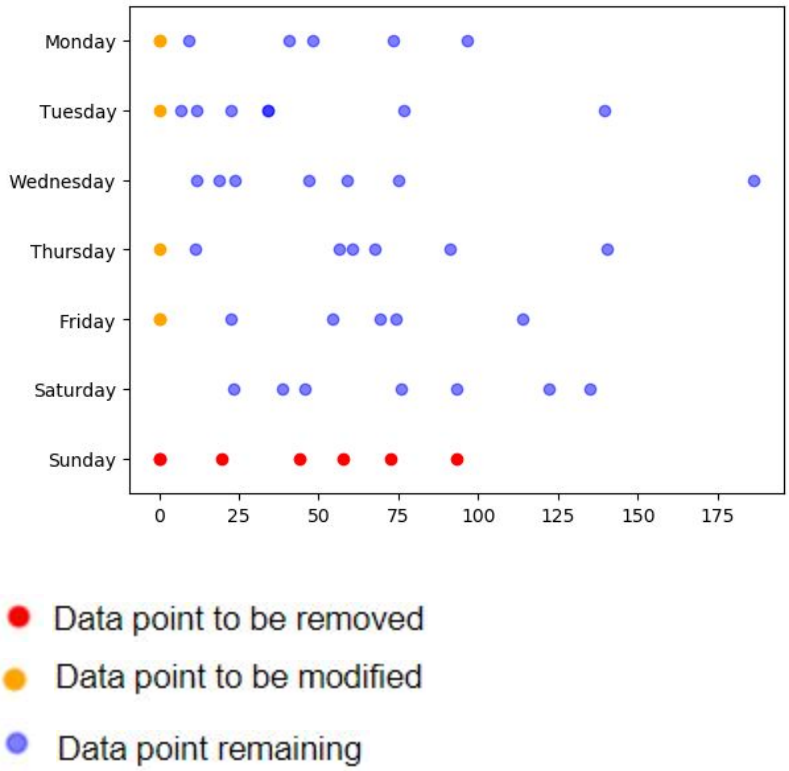


Fig 3.0 : Data point outlier modifications

The key factor considered for those data points removed lied in the fact that this particular store does not offer any service on Sundays however, for extraordinaire situations, it was operating during these days during this period. This is relatively evident as the overall weekend mean is diminished (by 11%) when considering data points on Sundays and possibly suggesting that the usual customer is not habituated in attending this place at this day.

For those data points that were altered, in yellow, all of these had a value equal to zero. It is well known and communicated by the owner that while adopting the new Point of Sale terminal several sales representatives struggled when making a transaction hence avoiding introducing

sales at all in some cases. Given this condition a modification was performed only in these situations and updated those with the median from a given day aligning to standard practices[1].

All final modifications are exhibited in Fig. 3.1 making sure that these new values are appropriate to be used in machine learning algorithms without losing the essence of the data itself.

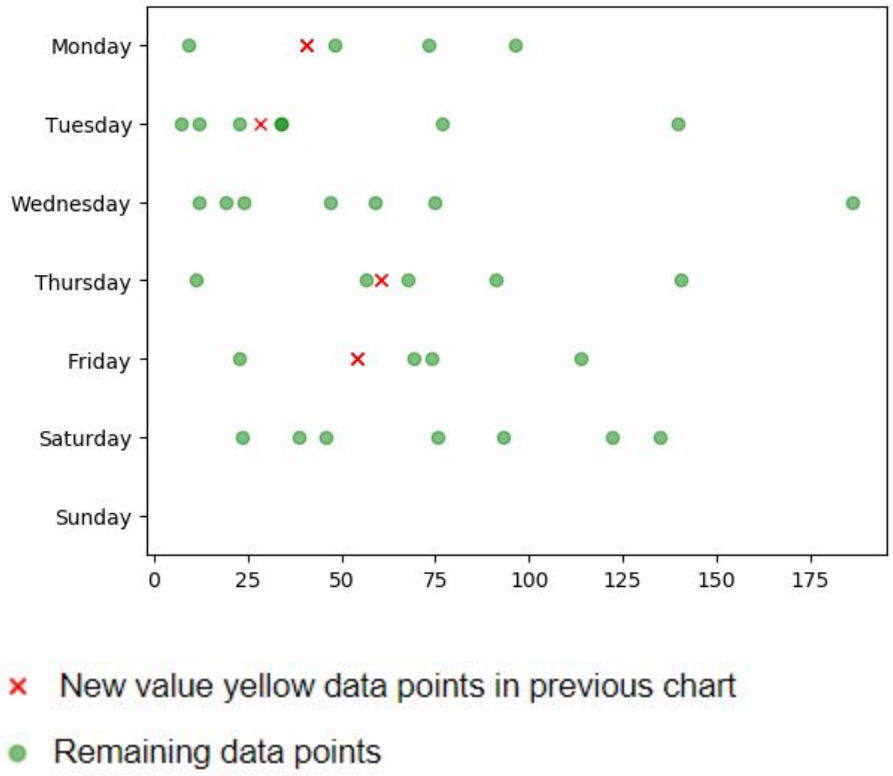


Fig. 3.1 : Final data point modifications

Implementation

Our main purpose during implementation was to develop an environment that enabled model experimentation and understand the inner dynamics. In order to obtain this, several specialized functions were created, the primary one being illustrated on Fig 3.2 which contemplates stochastic aspects while at the same time the ability to the test various features against a particular learner.

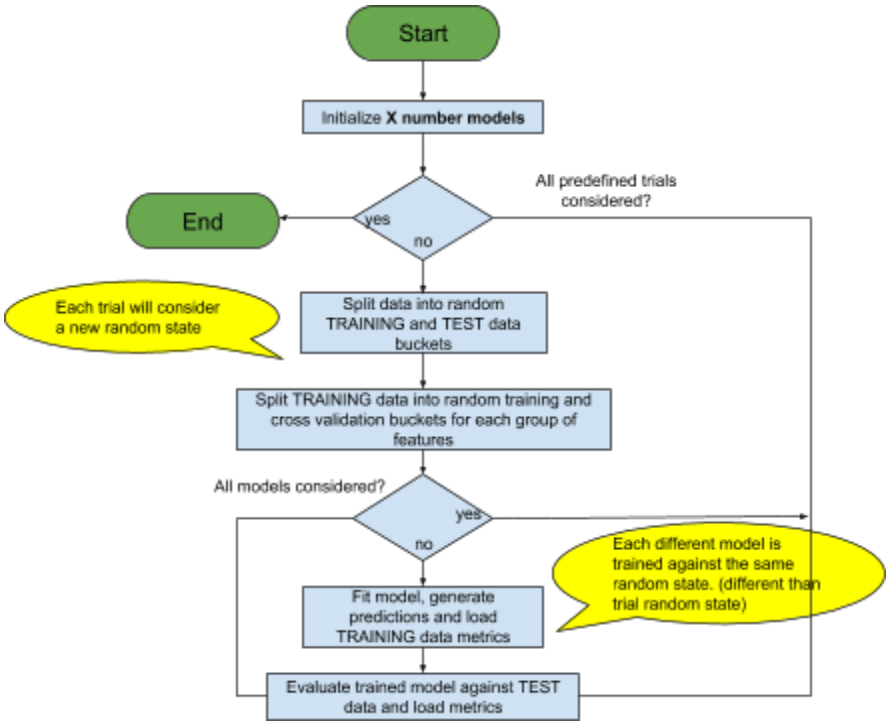


Fig. 3.2: Primary function program flowchart

The variables for this specific function are defined as,

The ***'X number models'*** : determined by the total groups of features and total machine learning algorithms, or techniques, at stake. For example in our first model comparison we considered two groups of features ('Day of the week' - 'Holiday?' and 'Weekend?' - 'Holiday') and two learners (Linear Regression and Gradient Boosting Regressor) making a total of four models.

trials: currently set to 3,000 as it was found heuristically a reduction in metrics variation.

bmark: actual benchmark set to 80.

cv_test_size: cross validation test size during TRAINING initially set to 0.20 or 10 days.

Once the models are properly trained the results for each are displayed along with a frequency chart (Fig 3.3) that allows to understand the confidence level (non-overfitting cases) and the number of favourable outcomes, above 0% and lower than the current benchmark of 80% as seen below.

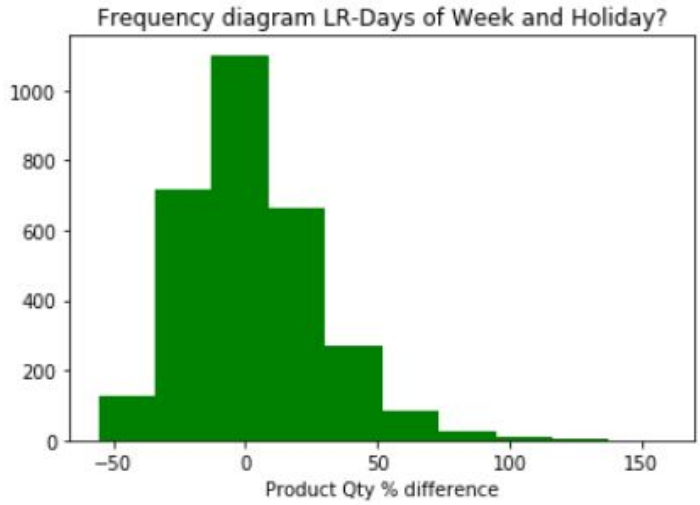
***LR**-Days of Week and Holiday?*
MAE : Mean = 84.12 1 minus STD = 41.38 1 plus STD = 127.85
R2 : Scores above 0.00 29.97 %

***G1**-Days of Week and Holiday?*
MAE : Mean = 86.92 1 minus STD = 42.10 1 plus STD = 132.70
R2 : Scores above 0.00 29.53 %

***LR**-Weekend? and Holiday?*
MAE : Mean = 84.80 1 minus STD = 43.05 1 plus STD = 127.17
R2 : Scores above 0.00 30.67 %

***G1**-Weekend? and Holiday?*
MAE : Mean = 84.80 1 minus STD = 43.05 1 plus STD = 127.17
R2 : Scores above 0.00 30.67 %

The first two letters symbolize the learner in concern. LR = Linear Regressor and G1 = Gradient Boosting Regressor, the latter has a variety of configurations hence the number included.




```
Probable non-overfitting cases 48.10%

Product qty percent of:
Winning cases (above 0% and lower than 80%) is 49.37%
Lost cases (below 0%) is 49.93%
Lost cases (above 80%) is 0.70%
```

Fig. 3.3: Frequency chart example for a specific model

Complications during implementation relate to the fact that only 50 data points were taken into account this created a limitation in the amount of predictions being substantially less than the requirement of 30 days. The solution formulated consisted in generating 'dummy' data that followed a similar pattern only to the most frequent feature 'Days of Week'. For example if there were five different values for a given day, i.e. Monday, these were copied to the next following Mondays (and for the rest of the days) until reaching 6 months approx or in this case 250 data points. These two disparities can be clearly observed within the Jupyter notebooks 'Forecasts_Jewelry_1.66_initial' and 'Forecasts_Jewelry_1.66_5xData' respectively under the Data Preprocessing section. This decision was firstly consulted with the owner and agreed that at least for the past 10 years sales from May to December have minor fluctuations.

Refinement

Once a particular model was identified to stand out from the rest, it was isolated and hyper parameter tuning was applied. With the assistance of a unique function called *GridSearchCV* from the Scikit Learn model selection library and guided by the web article "Complete Guide to Parameter Tuning in Gradient Boosting (GBM) in Python"[7] two different approaches were seeked.

First option consisted in adjusting parameters for the selected model which was basically a non-optimized Gradient Boosting Decision Tree with features 'Days of Week' and 'Holiday?' while maintaining the default learning rate of 0.1.

For the second option, the learning rate was modified in order to reduce the number of estimators (and expecting to reduce the amount of training time) since the previous choice resulted in a figure higher than 100. Following the article guidelines a value of 60 can be considered a good start.

As shown on the Jupyter Notebook file 'Forecasts_Jewelry_1.66_5xData' cases with R2 score higher than zero increased from **1.10% (G1)** to **1.37% (G2)** and finally to **1.77% (G3)** however non-overtting cases are highest on **G2** with a percentage of **47.03%** converting it into a slightly more trusted model than the rest**.** For time performance, only **G2** and **G3** were compared, since considering G1 didn't bring any value at this point, being the lowest on the latter with almost 100% reduction **(0.97 min vs 2.07 min)** from the former.

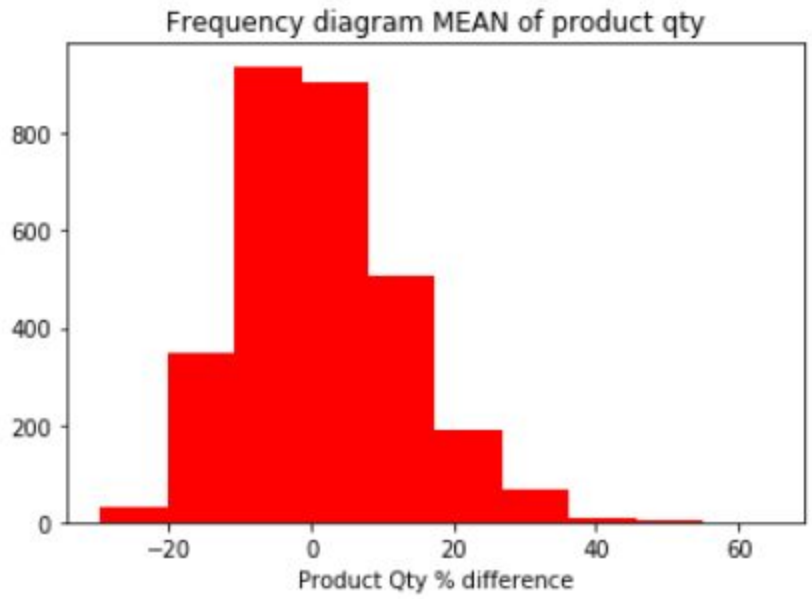
It is fair to mention that the main function that performs the training for each different trial was modified to ensure a real improvement while still keeping the stochasticity when splitting the data in training and testing sets.

IV. Results

Model Evaluation, Validation and Justification

Evendough a model was chosen to be optimized, expecting an ROI (return of investment) out of any decision based on this model is highly unlikely as we stumbled upon a more than 50% chance of overfitting cases. A positive aspect that is worth mentioning is its robustness, considering that there are a millions of different splits, small variations (less than 1%) were obtained considering only 3,000 trials.

At this point, it was proven that using the mean of the total product provides a more accurate prediction than any model previously trained under the features in concern. As presented on the following chart in Fig. 4.0, to a certain extent it can even improve the current strategy (or benchmark of 80% of product surplus) as the maximum negative most frequent impact lies at -20% of product shortage so it is only a matter of properly planning for these type of situations to achieve a supply and demand equilibrium. In other words this becomes more of a managing risk problem.




```
Winning cases (above 0% and lower than 80%) is 50.63%
Lost cases (below 0%) is 49.37%
Lost cases (above 80%) is 0.00%
```

Fig. 4.0 : Product mean relevant metrics

V. Conclusion

Free-Form Visualization

While making an adjustment from product grams to product quantities new patterns were revealed that could assist in create additional features. These patterns are visible and highlighted in red on Fig 5.0 as opposed to Fig. 5.1 where only the portion in green was identified, in this case representing the ‘Holiday?’ feature. The target variable was initially portrayed in grams as this is how the owner acquires the product each month however a different approach may seem a more feasible approach.

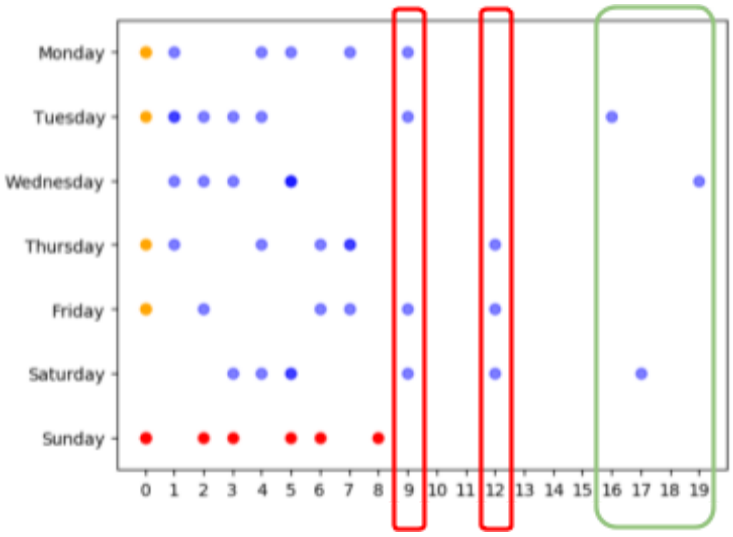


Fig. 5.0: Additional possible features

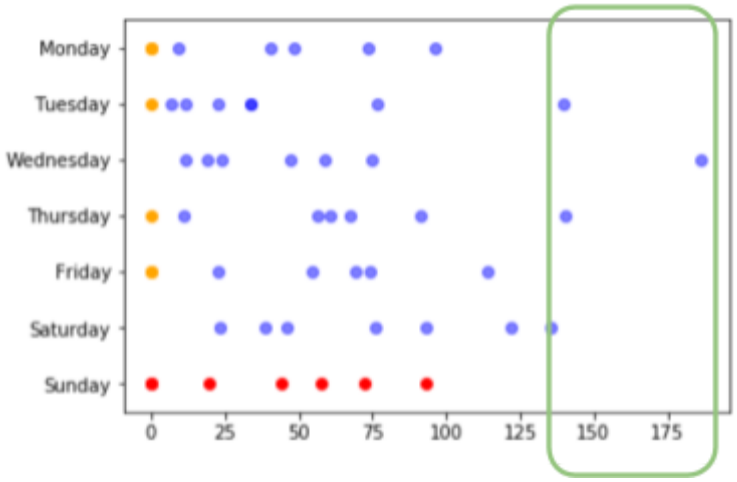


Fig 5.1: Current feature considered and used for training models.

Reflection

This project was initiated as a goal to provide the shop owner database infrastructure services however a business case had to be created to ensure the investment had tangible returns. Only a product line was taken into account as the cost to cover over 1,500 different products, financially and logistically, was too high. From creating the product codes, acquiring the Point of Sale terminal, providing training to sales representatives and even to the owner there was a considerable amount of effort focused on making it a holistic and non-intrusive solution to obtain the right data. During the implementation and evaluation phases several questions emerged that have not yet being answered for example. How come the R2 frequency scores above 0.00 decrease when additional data was included? Will a low or no variation within the different values for each feature (i.e. if the amount of product sold for each day is always the same) have an increase in prediction accuracy?

There is absolutely personal room for improvement in delimiting and even in finding the problem itself. In this situation capturing the data occurred first then understanding the issue (adding to the fact that according to the personnel it was non-existent) leading to circumstances such as spending a considerable amount of time in cleaning the data, generating and designing product codes that were not even considered providing no value and only cost at the end. This was absolutely a lesson learned, if there is no problem identified there is no point in capturing or even analyzing existing data.

Improvement

As described on the Free-Form Visualization section new features can be detected and can generate additional and more accurate models by simply displaying the data as product quantity

instead of grams. A feature of ‘Payday’ can be considered for example which has higher frequency than ‘Holiday?’.This will imply modifying some of the functions to account for a third feature however it should not take more than a day of developing and testing the code.

Works Cited

1. “Working with Missing Data in Machine Learning - Towards Data Science.” Towards Data Science, Towards Data Science, 10 Dec. 2017, https://towardsdatascience.com/working-with-missing-data-in-machine-learning-9c0a430df4ce
2. Dmlc. “Dmlc/Xgboost.” Github, 21 Mar. 2018, https://github.com/dmlc/xgboost/tree/master/demo#machine-learning-challenge-winning-solutions
3. Everitt, B.S, and A. Skrondal. *The Cambridge Dictionary of Statistics*. Cambridge University Press, 2011.
4. Hyndman, Rob J., and Anne B. Koehler. *Another Look at Measures of Forecast Accuracy.* 2005.
5. Salkind, Neil J. *Statistics for People Who (Think They) Hate Statistics: Using Excel 2016.* SAGE, 2017.
6. Yan, Xin, and Xiaogang Su. *Linear Regression Analysis: Theory and Computing.* World Scientific, 2009.

7. Jain, Aarshay. “Complete Guide to Parameter Tuning in Gradient Boosting (GBM) in Python.” Analytics Vidhya, 27 May 2016, https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/