Prediction procurement for Grocery Retailers

**Data Science Project Protocol**

Hereby submitted to: Dr. Tomas Karpati

By: Michael and Eyal

Date: February 24, 2019

Table of Contents

[Introduction 5](#_Toc1925139)

[Methodology (Project design) 7](#_Toc1925140)

[Data 7](#_Toc1925141)

[The data source: 8](#_Toc1925142)

[External data sources 8](#_Toc1925143)

[Training and test 8](#_Toc1925144)

[Time frames 8](#_Toc1925145)

[Subject definition: 8](#_Toc1925146)

[Outcomes 8](#_Toc1925147)

[Confounding 9](#_Toc1925148)

[Source of bias: 9](#_Toc1925149)

[Data exploration 9](#_Toc1925150)

[Enriching the data 9](#_Toc1925151)

[Outliers 10](#_Toc1925152)

[Missing values and corrupted data: 10](#_Toc1925153)

[Data retrieval protocol 10](#_Toc1925154)

[Models 11](#_Toc1925155)

[The data 11](#_Toc1925156)

[Train, validate and test 11](#_Toc1925157)

[Data balance 12](#_Toc1925158)

[Subsampling 12](#_Toc1925159)

[Technique 12](#_Toc1925160)

[Cross validation 12](#_Toc1925161)

[Measures to evaluate the model 12](#_Toc1925162)

[Assembling 13](#_Toc1925163)

[Deployment of your model 13](#_Toc1925164)

[QA 13](#_Toc1925165)

[Final user 13](#_Toc1925166)

[Presentation: 14](#_Toc1925167)

[Training and interpretation 14](#_Toc1925168)

[Platform 14](#_Toc1925169)

[Model updates 14](#_Toc1925170)

[Incomplete data 14](#_Toc1925171)

[Production models 15](#_Toc1925172)

[Results 15](#_Toc1925173)

[Amount of data 15](#_Toc1925174)

[Outliers 15](#_Toc1925175)

[Missing values 16](#_Toc1925176)

[Prediction Results: 16](#_Toc1925177)

[Conclusion 16](#_Toc1925178)

[Annexes 19](#_Toc1925179)

[Annex 1 – GitHub 19](#_Toc1925180)

[Annex 2 – Code description 20](#_Toc1925181)

[Step 1 – Data retrieval 21](#_Toc1925182)

[Step 2 – Pair plot 21](#_Toc1925183)

[Step 3 – Box and join plots 22](#_Toc1925184)

[Step 4 – Client ID, Size 23](#_Toc1925185)

[Step 5 - Mechkar lib 23](#_Toc1925186)

[Step 6 – Z score 24](#_Toc1925187)

[Step 7 – Single product test 24](#_Toc1925188)

[Step 8 – P Value and correlation 25](#_Toc1925189)

[Step 9 – Initial prediction calculation 25](#_Toc1925190)

[Step 10 – High correlation dataset 26](#_Toc1925191)

[Annex 3 – High level QA Protocol 27](#_Toc1925192)

[Annex 4 – Data models, results comparison 28](#_Toc1925193)

# Introduction

*“AI technologies could eliminate many levels of manual activities in areas such as promotions, assortments, and supply chain. AI will enable retailers to increase both the number of customers and the average amount they spend by creating personal and convenient shopping experiences.” — McKinsey Global Institute (2017)*

Seeing the number of big retail chains that are closing stores (think Sears and Payless) may support a surface-level impression that retail is dying out. But that could not be further from the truth. The industry is booming in the digital space. The revolution started by companies like Amazon and eBay has led to huge challenges for the traditional retail business model, but also massive potential for retailers and consumers alike.

This also means that there is an increasing shift towards optimization and efficiency, and a shift away from excess and waste. Retail is well-placed to benefit from the intersection of Artificial Intelligence, machine learning and big data. There is a need to manage and track a large number of items across various categories, track consumers’ shopping habits and above all, maintain a compelling brand that keeps consumers coming back. Today’s consumer wants to keep up with the latest trends, but also craves convenience; hence, the popularity of subscription boxes and online shopping. A recent survey of retailers worldwide identified cost savings, enhanced decision-making and process automation as some of the main areas that AI has the potential to impact meaningfully[[1]](#footnote-1). (Chandran, Jul 2018)

In the presented project we have done, we looked in to the sales out data of a grocery retail chain that supports more the 100 franchises.

The franchises placing orders using the retailer web portal, later those orders are being sent to the manufactures or the retailer warehouse.

As can expected, the demand is varying from franchise to franchise and from time to time. A lot of causes effecting the demand, among others are: price and discounts. time of the year, holidays, weather, product availability and more.

A note that should be taking in to account are the delivery days, in the project below we measures all the parameters on a week based, since we are not dealing with the sale out of the franchises but with the sale out of the retailer to the franchises, the days of deliverables are, in most cases, fixed i.e. manufacture X deliver its goods to franchise Y every Monday, there warehouse delivers to franchises Z every Tuesday and so on.

In this project we are focusing on the purchasing orders form a few perspectives:

We wish to identify patterns in the PO

We wish to exam how "external causes" are affecting the franchises purchasing.

We wish to build a forecasting model allowing the retailer (and its manufactures) to plan more accurately the availability of the goods

# Methodology (Project design)

## Data

The data we are using has been generated by the ERP system of the retailer and includes the sale out to its franchises in a period of 6 month.

The data includes the franchise ID, date of order, products ID (SKU), quantities and price.

Please note that the data is owned by the retailer and granted to be use in this exercise only.

In addition to it, we have used a public based information such as:

* Weather
* Weeks clustering (split the data to weeks)
* Israel Holidays date

The overall the data includes more than 600,000 order lines[[2]](#footnote-2) covering a period begin at March 2018 until August 2018.

As we refer to grocery retail, this period includes Peshach[[3]](#footnote-3) and the summer holidays[[4]](#footnote-4), two periods that requires a special attention.

The data included 3 tables (DB):

1. PO main data base
2. Weather
3. Holidays

The DB's ware joined in to a flat file based on the date of order[[5]](#footnote-5).

Below you will see that we have used two similar flat files that differ in the number of products each includes.

### The data source:

The data main source is the ERP system if the retailer, more data added to it from the public domain

### External data sources

We have added weather data and Holidays date.

### Training and test

Due to the size of the DB, we have used 90% of the data for turning and 10% as test

### Time frames

The project data is of 6 month wherein the training data (90%) and the testing data (10%) are randomly chosen

### Subject definition:

In our view, the flat file data resides under inclusion criteria

### Outcomes

The main purpose of the project was to predict the demand.

Since the manufacturers wishes (as much as possible) to be aligned with the demand, our outcomes assist to (1) define the quantities that should be manufactured to meet the demand (2) allow more accurate pre-orders by the retailer planners (in case that the products are delivered from its warehouse).

### Confounding

There are a few external confounders that can affect the results and reside in data that we don’t have. One example would be the franchise line of credit (LoC), the franchises purchases are limited by a lined of credit that the retailer granted each on of them (open credit). the franchises are not allowed to exceed this LoC, in most cases the LoC is managed properly, however it might affect the purchases in special periods such as *Pesach* where the pre-holidays purchases are increase dramatically.

### Source of bias:

we don’t see any source of bias in the data we have. (no one in the supply chain wishes to have high stock volumes in its possession)

### Data exploration

As mentioned above Initially we got Excel files with the data, for data exploration and data unification we used SQL and for data visualization we used Tableau.

Prior to the data cleansing that has been done on the SQL, we export a flat file to Python and continued the exploration in the notebook we used. In later stages we have used p-value and Pearson correlation to examine the relationship and the influence of the parameters on each other

### Enriching the data

We used data enrichment allowing the different models to examine more data, the following methods have been applied on each row in the DB

1. We have defined a franchise ID based on its quantity portion out of the franchisee overall quantities.
2. The quantity in week+1 (Y value)
3. Franchise size (Small or Big)
4. Weather of this week (Hi temp, Low temp and Rain Y/N)
5. Holidays, if a holiday happen at the week of the purchase (Y/N)

### Outliers

Generally, since we are dealing with real data outliers should be investigated carefully, outliers where handled based on the cause of each, errors in the data, miscalculation or any other cause.

A common case in the retail market is a skip delivery\week and therefore on the week before or after the deliverables are doubled, in such case, the detection is fairly simple, and we can split the order to two or remove the two outliers (the double and the zero).

In addition, we have identified a few outliers that although based on real data, they effect the outcomes (prediction and accuracy) and we manage to remove them from the dataset.

### Missing values and corrupted data:

We identified approximately 170 lines (out of 600K) with corrupted data or missing values, those lines are mostly in the boundaries of the data and therefore we decide to remove them from the dataset

### Data retrieval protocol

(see at github <https://github.com/eyalfarkash/TCDS-Final.git> under Documents, Data Retrieval Protocol V4.xlsx)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature name** | **Source** | **Value type** | **Scale** | **Data type** | **Unique count** | **Measure unit** | **Conversion factor** | **Min** | **Max** | **Outlier treatment** | **Null** | **Notes** |  |  |  |  |
| Product | PO | Numeric | Nominal | Varchar | Yes | N/A | N/A | 0 | 7.3E+12 | Leave | NULL | SKU number |  |  |  |  |
| Weenum | PO | Numeric | Nominal | Integer | No | N/A | N/A | 9 | 36 | Leave | NA | week number |  |  |  |  |
| Above\_Standart\_Price | PO | Numeric | Nominal | Boolean | No | N/A | N/A | 0 | 1 | Leave | NULL |  |  |  |  |  |
| Below\_ Standart\_Price | PO | Numeric | Nominal | Boolean | No | N/A | N/A | 0 | 1 | Leave | NULL |  |  |  |  |  |
| Sum\_Quantity | PO | Numeric | Nominal | Integer | No | Unit | N/A | 0 | 10000 | Null | NULL | total quantity in a week | |  |  |  |
| Standard\_Price | Product | Numeric | Nominal | Float | No | N/A | N/A | 0 | 1000 | Leave | NULL | Standard price |  |  |  |  |
| Holiday | Holiday | Categorical | Nominal | Boolean | No | N/A | N/A | 0 | 1 | Null | NULL |  |  |  |  |  |
| Temp\_Hi | Weather | Numeric | Nominal | Float | No | N/A | N/A | 10 | 50 | Leave | NULL |  |  |  |  |  |
| Temp\_Lo | Weather | Numeric | Nominal | Float | No | N/A | N/A | 0 | 25 | Leave | NULL |  |  |  |  |  |
| Temp\_Rain | Weather | Numeric | Nominal | Boolean | No | N/A | N/A | 0 | 1 | Leave | NULL |  |  |  |  |  |
| Avg\_Price\_Change | Internal | Numeric | Nominal | Float | No | N/A | N/A | -1000 | 1000 | Leave | NULL |  |  |  |  |  |
| Sum\_Quantity\_1\_Y | Internal | Numeric | Nominal | Float | No | N/A | N/A | 0 | 100000 | Leave | NULL | The sale quantity in Week+1 (it is our Y) | | |  |  |
| Avg\_Price\_Change\_1 | Internal | Numeric | Nominal | Float | No | N/A | N/A | -1000 | 1000 | Leave | NULL | avg price change in week+1 | |  |  |  |
| All\_week | Internal | Numeric | Nominal | Integer | No | N/A | N/A | 0 | 39 | Leave | NULL | Number of weeks that the frencise is operating | | |  |  |
| Count\_week\_PO | Internal | Numeric | Nominal | Integer | No | N/A | N/A | 0 | 39 | Leave | NULL | Number of weeks that the frencise bought this produck (SKU) | | | |  |
| Sum\_Quantity\_1\_minus\_Sum\_Quantity | Internal | Numeric | Nominal | Integer | No | N/A | N/A | 0 | 1000 | Leave | NULL | total quantity of next week minus this week (the change) | | | |  |
| Avg\_Quantity\_for\_Week | Internal | Numeric | Nominal | Float | No | N/A | N/A | 0 | 1000 | Leave | NULL |  |  |  |  |  |
| Avg\_Quantity\_for\_PO | Internal | Numeric | Nominal | Float | No | N/A | N/A | 0 | 100 | Leave | NULL |  |  |  |  |  |
| Total\_Quantity\_for\_Customer | Internal | Numeric | Nominal | Integer | No | N/A | N/A | 0 | 100000 | Leave | NULL |  |  |  |  |  |
| Total\_Quantity | Internal | Numeric | Nominal | Integer | No | N/A | N/A | 0 | 100000 | Leave | NULL |  |  |  |  |  |
| Frequency\_PO | Internal | Numeric | Nominal | Integer | No | N/A | N/A | 0 | 10 | Leave | NULL | how often a PO for the product is issued | | |  |  |
| Customer\_ID | Internal | Numeric | Nominal | Float | Yes | N/A | N/A | 0 | 100 | Leave | NULL | it is a relative number, the customer ID is its quantities portion out of all quantities | | | | |
| Customer\_Number | PO | Numeric | Nominal | Integer | Yes | N/A | N/A | 416000000 | 416000170 | Null | NULL |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

## Models

### The data

We have initiated two datasets and compared the results between them:

* 75-products database: the DB includes the app 19% of the orders performed by the franchises, overall approximately 110K lines
* 3-Products database: the DB is a subset of the 75 products DB, wherein only 3 products exist, comprise approximately 9K orders

### Train, validate and test

As mentioned above, we have analyzed the data on a weekly base, we have operated in the following models on the data:

Randomly chosen: 90% for train while the other 10% used for dev/test.

### Data balance

We didn’t saw the need to data balance in the grocery market since a tolerance is acceptable in both direction

### Subsampling

The data we use is divided to franchises, however a sample data of each does not represent its purchasing history and trends, therefore stratified sampling is not applicable for our data subsampling

### Technique

We have used a few techniques and compare between them

In the data side we had, a POs with 23 different parameters such as:

1. The number of franchises who pouched this SKU
2. The week number
3. Holiday (yes/no)
4. Average weather of this week
5. (The outcome is) the purchased quantity (the Y parameter)

### Cross validation

Since we have a limited amount of data, we have used 50 (K) fold cross validation on it.

It allowed us to provide more accurate results.

### Measures to evaluate the model

We are using root mean squared error or root mean squared deviation, using RMSD we measure the difference between the actual values and the estimated values.

Since we are calculating "large numbers" (i.e. the total quantity of a product that has been bought by all the franchises together) the estimation can have an error, however it is important to understand its magnitude and if it's positive or negative (over stock or out of stock).

We find this method as the most appropriate to use while dealing with finite numbers.

### Assembling

We have tested based on the well-known models that fit to structured data. And used the most accurate one.

We chose not to use assembling since we got a much higher result in one technique comparing to the others (see more in annex-4, Data model results comparison)

## Deployment of your model

### QA

The QA of the project shall be performed by a professional QA team with knowledge in the field and the data.

In our case, the procurement team is responsible for the orders, however it is important that the commercial/marketing department will give its inputs since special offers, promotions and product availability may affect the prediction

QA protocol see in annex-3

### Final user

We have two final users (1) the manufactures and (2) the retail chain planner;

* The manufactures – they are planning the production based on the demand, a better forecasting of the demand shall optimize the production and supply chain operation.
* The chain planner – the chain planned place orders on behalf of the warehouse to the manufactures, a warehouse procurement is different from the end user, a purchasing order should be of high volume and for a long duration, an accurate planning of the warehouse reduce the overstock or our of stock of the warehouse in addition it assist in stock management in terms of expiration dates and such.

### Presentation:

Using the model. A planner can enter the data it has, such as product number, store ID, planned week, price act. and get the forecast of this franchise / product order quantity.

### Training and interpretation

We don't think that it would be necessary, an expert in the field would understand the outcomes.

### Platform

At this stage of the project, as we are dealing with relatively small amount of data and for a limited number of users. a standard computer\server with the needed environment installed will do the job.

In the future, if necessary, it can be exported to a cloud services utilizing GPU's[[6]](#footnote-6).

### Model updates

It is difficult to predict since it related very much to the "real life" results it provides. As the amount of data increase and the model has more data to learn from, we believe that it would become more accurate. In case that it will not happen then a reevaluation of the model would be needed.

### Incomplete data

In case of incomplete data, it is most likely that the SW will error and notify the user, for the data that we are using there are no defaults values that we can use.

Defaulted values such as average will generate noise in the data and too much of those will affect the prediction.

### Production models

Each of the below algorithm have been run a few times on each dataset

* Linear regression
* SVM
* Ada Boost
* Bayes
* Logistic regression

Based on the results we got the best algorithm is Ada Boost, in order to maximize the results we used cross validation of cv-50, in it we riches a max value of 0.5442

# Results

## Amount of data

As I have mentioned we have used two datasets, (1) 75 products and (2) 3 products:

The 75 products dataset has 107,077 lines for train and 11,897 lines for test

The 3 Products dataset has 7,319 lines for train and 813 lines for test

## Outliers

We didn’t have so many outliers in the data due to its source, however we have removed a few outliers (less than 50) that we thought can impact the results.

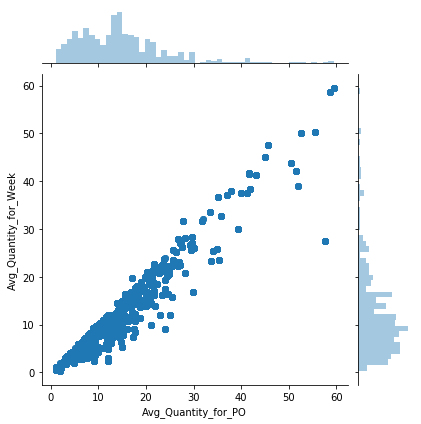
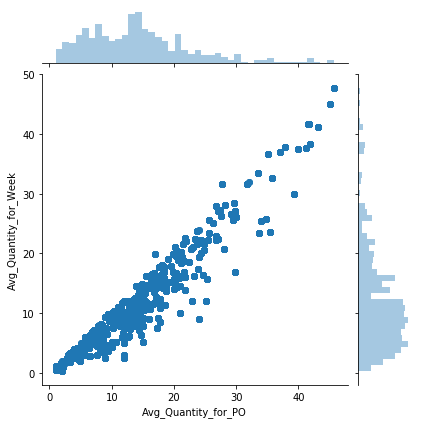
 

Chart 1: Outliers before and after

## Missing values

In the whole dataset we had app 170 lines with missing values, since the values ware mostly prices, we deleted those lines from the dataset.

## Prediction Results:

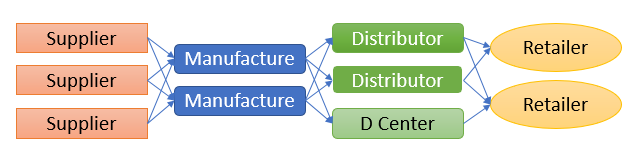
In the 75-product dataset we managed to get a max accuracy of 4081 out of 11897 lines (34.3%) with mean absolute error of 22.6

While in the 3 Products we managed to get an accuracy of 351 out of 813 lines (43.1%) with mean absolute error of 4.19

# Conclusion

In this project we have developed a proof of concept for *prediction procurement*.

The retail market and in particular the grocery market relays on product availability (the right product in the right time in the store), those products are delivered by distributors and manufactures. Having said that, the distributors are relay on the manufacturers, suppliers and so on, causing a long supply chain between the supplier and the retailer with lead times in between.



The immediate result of the above is: It is important to plan in advance and to place the current order in the right time.

When we initiated the project, after gathering the data, we understood that the data "as-is" is insufficient and we need to take care of it, as *"the quality of the results = the quality of the data"*.

From the retailer we got the data in excel sheets, trying to "play" with it in MS-Excel was intricate. We understood that a data environment is needed for data cleansing and therefore we transferred it to SQL DB that allowed us to manipulate it more efficiently, while using Tableau as a data visualization tool.

In addition, we saw the need (with your assistance) to add more variables to the dataset, external variables such as weather and Holidays have been added and internal such as average-price, average quantity and more. At the end a unified flat file for the use of the model was generated

The next phase was to build the python SW that supports the model, at this stage, again, we looked in to the data, exploring animalities or mistakes that might affect the results or the operation itself.

Upon finalization of the data, we have ren a few supervised models and compare the results, we looked in to the accuracy of the results and at the error rate. Since we are dealing with grocery a non-accurate value with a low error can be sufficient[[7]](#footnote-7)

We saw that the outcomes accuracies are relatively low. While reviewing the dataset we found that there is a low correlation between the different products and that each acting differently, therefore we have decided to check if a smaller dataset can result in more accurate outcomes. we generated a 3-products dataset for tests (as of that point we used a 75 products dataset).

As expected, the 3-products dataset, although it is a smaller one (less than 10% of the original) it generates more accurate results on both accuracy and error-rate.

At this stage we have initiate two operations: (1) remove low correlation parameters and a (2) cross validation operation. While the first didn’t proved itself dramatically in the second we got better results, up to 54.4% accuracy.

It is worth noting that the most accurate model we used was Ada boost[[8]](#footnote-8), that is used as a general approach for many statistical models utilizing stumps and weights.

Of course, as any other machine learning algorithm it has its own limitations, the main limitation is the data, outliers and noise, in the model that we have developed we were highly concern about the data quality and spent a large effort on its cleansing, in a production environment it is not always the case, in addition: (1) it might be that different products or at a different stage in the supply chain act differently and (2) the number of products, as it seems to us, the model is more suitable for low number of products and therefore we are not recommending using it with large number of products.

**To conclude**: In this project we have manage to show the advantages of *procurement prediction* based on machine learning, the results that we got are a good starting point in the journey towards a highly accurate errors free prediction model. We have presented the limitations we faced along the way and the methods we used. Yet, at this stage, it cannot be used as a "single source of trues" product to be used by retailers, but an assistance tool or a step in the direction towards *automatic orders* system

# Annexes

## Annex 1 – GitHub

All the files that have been using for this project reside in github and can be view by using the following link: <https://github.com/eyalfarkash/TCDS-Final.git>

The git includes the following directories:

* **Code**: in it you will find the code we develop for the project:
  + Final-project-CSV\_v4\_FullData-final.ipynb – was used on top of the 75 products dataset
  + Final-project-CSV-v5\_3Products-final.ipynb – was used on top of the 75 products dataset
  + SQLQueryFin.sql – includes the SQL code used for the generation of the flat files
  + All the other files are drafts and tests we done.
* **Data**: includes the data files we use
  + 3Products.CSV is the 3products dataset
  + Final\_csv1\_full\_v3 is the 75 products dataset
  + Hebrew holidays used to identify the Israeli holidays dates
  + Weather2018 used for day and night temperatures and rain
* **Documents**: includes some documentation on the project
  + The EDA directories are of the Mechkar HTML pages
    - [3Product-EDA-Mechkar-V4](https://github.com/eyalfarkash/TCDS-Final/tree/master/Documents/3Product-EDA-Mechkar-V4) is of the 3 Products
    - [Full-EDA-Mechkar-HTML-V3](https://github.com/eyalfarkash/TCDS-Final/tree/master/Documents/Full-EDA-Mechkar-HTML-V3) is of the 75 Products
  + [Data Retrieval Protocol V4.xlsx](https://github.com/eyalfarkash/TCDS-Final/blob/master/Documents/Data%20Retrieval%20Protocol%20V4.xlsx) is the retrieval protocol
  + TCDS Project Protocol V5 – is this document
* **Ref-Code**: is a draft directory we place in the git for our internal use

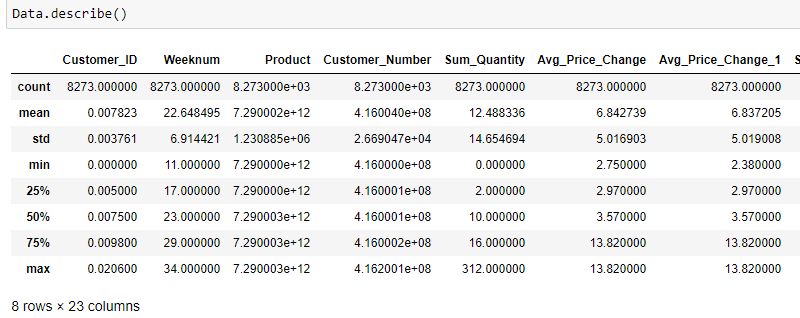
## Annex 2 – Code description

Below we will explain in high level the code we wrote for the project, although we used 2 datasets and at the end compared the results, both 75 products and 3 products code is mostly similar.

In this annex I will explain the 3Product code, available on git <https://github.com/eyalfarkash/TCDS-Final/blob/master/Code/final-project-CSV_v5_3Products.ipynb>

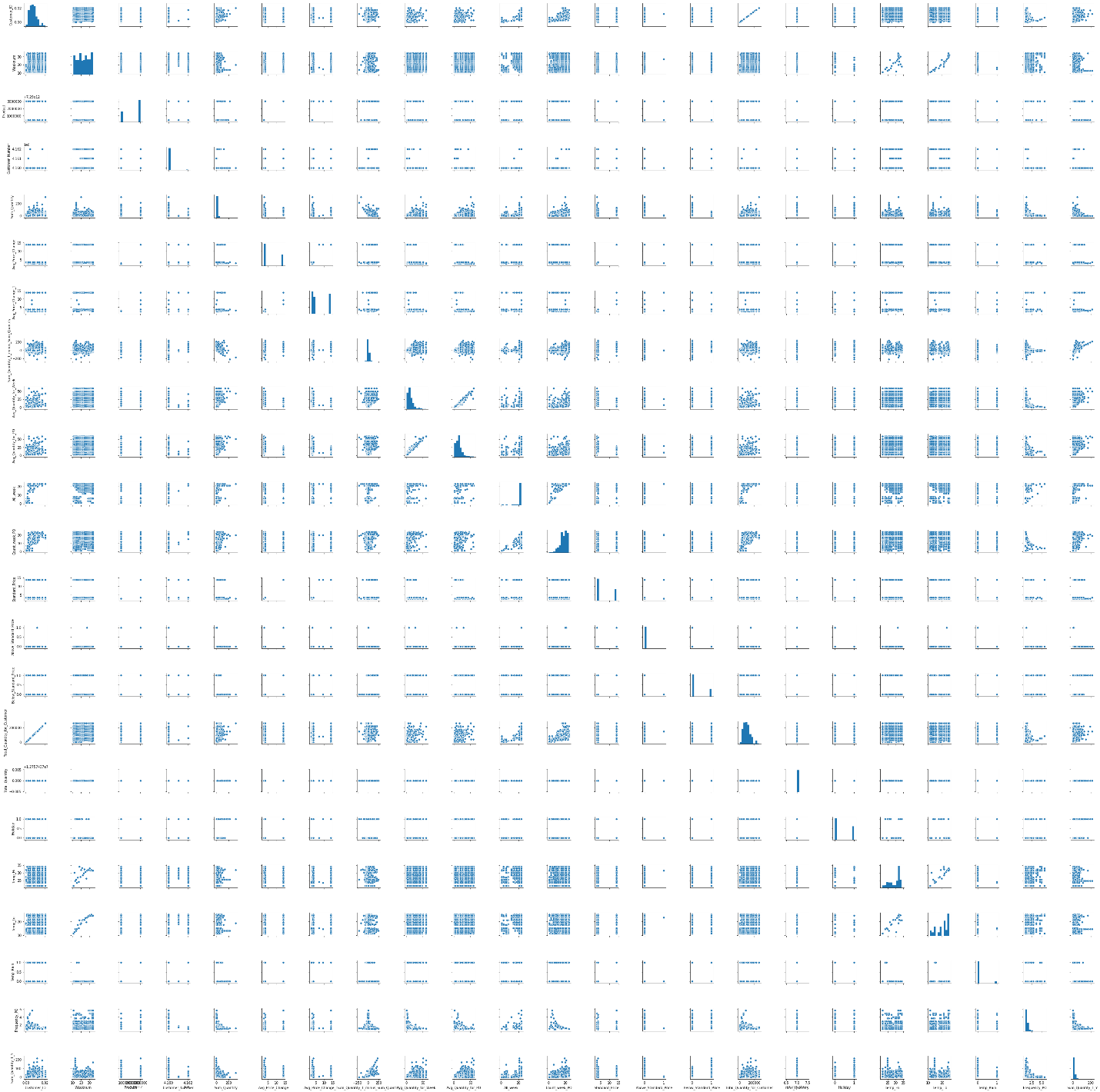
### Step 1 – Data retrieval

The data source is 3Products CSV dataset, from the describe we can learn about the dataset column and values



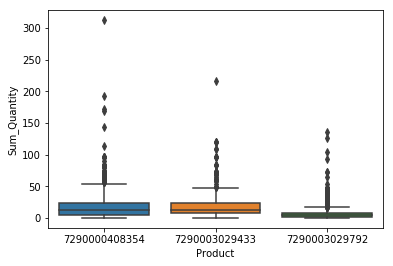
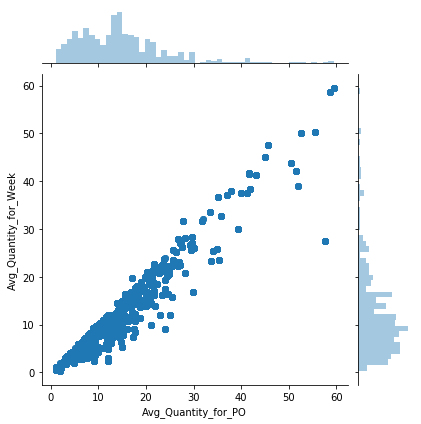
Step 2 – Pair plot

All 23 parameters are presented in a pair plot allowing us to look for patterns in the data



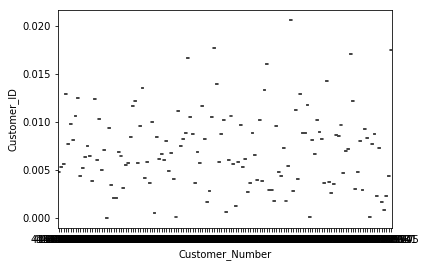
### Step 3 – Box and join plots

A few box plots show us how the data is distributed, in this example since we have only 3 products it can be seen more clearly than in the 75

### Step 4 – Client ID, Size

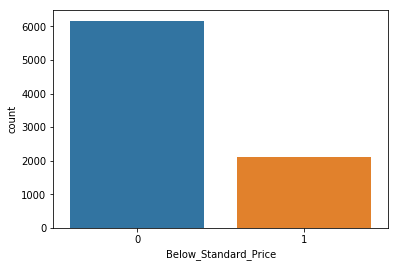
The identification of each franchisee was calculated according to its size, in the perspective of quantities, it allows us to have a meaningful number in the client\_id instead a useless continues number that might be taking in to account by the algorithm, by doing so the below presents the franchises based on size



### Step 5 - Mechkar lib

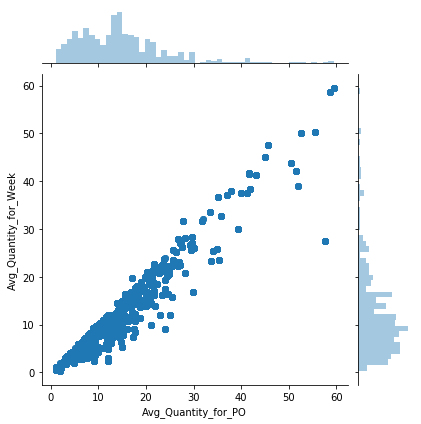
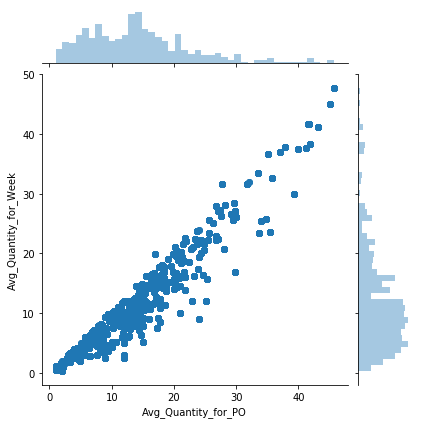
You can find the results of Mechkar in the git under Documents

One of the interesting plot is *count vs below standard price*, here we can see how many order have been made while the products where on special price



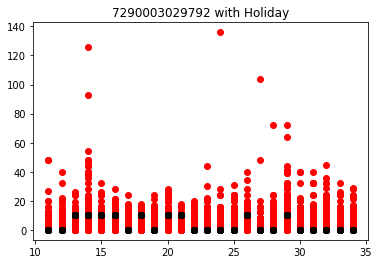
### Step 6 – Z score

In this step we used Z score to detect extraordinary values and based on that we have define trash holds to a few parameters, those thresholds removed outliers from the dataset

For example, see the below join plots of average quantity per week vs average quantity for PO 

### Step 7 – Single product test

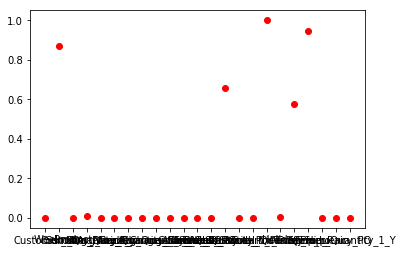
We have extracted a single product form the dataset to find out if we can learn something from it and how a single product effect the overall prediction. The extraction of a single product allowed us to see how the Holidays are reflected in the quantities of this product (the Holidays are in black, not-Holiday = 0, a week with a Holiday = 10)



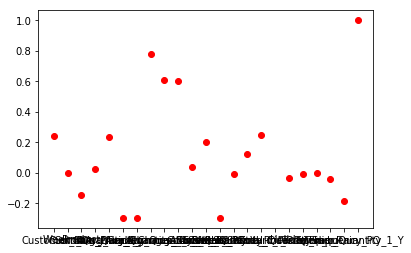
### Step 8 – P Value and correlation

In this step we looked for a correlation between the different parameters, to be used in following steps. In order to find it we used two methods, P value and Pearson correlation, we have performed it between all parameters.

In the plot you can find the p-values of the different parameters:



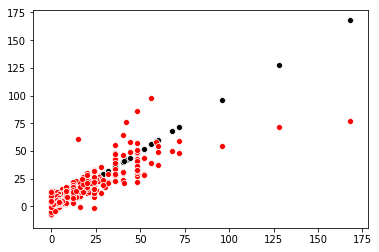
While the Pearson correlation looks much different



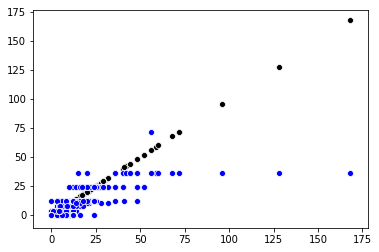
### Step 9 – Initial prediction calculation

In this step we have initiate the prediction algorithms while looking for the best result.

* Linear regression results ware low, with accuracy of 96 or 813 and mean absolute error of 4.15



* The best result has been achieved using Ada boost with accuracy of 351 of 813 and MAE of 4.19



### Step 10 – High correlation dataset

In this step we took the results of the p-value and Pearson correlation and cleaned the dataset accordingly, 7 parameters have been removed from the original dataset, and a new calculation has been done on the updated dataset.

In most cases the results ware lower or relatively the same as with the original dataset.

Step 11 – Cross validation

For this operation we keep on using the best algorithm we had and ran it using sklearn kfold, the best results we got was with cv=50 with up to 0.544 accuracy score

## Annex 3 – High level QA Protocol

|  |  |  |  |
| --- | --- | --- | --- |
| S/N | The operation | Required | notes |
| 1 | Data extraction | Required | Extract the data from the ERP system |
| 2 | Data accuracy | Required | Make sure that there are no animalities in the data |
| 3 | Data accuracy | Required | Make sure that the data is of the right product(s) |
| 4 | Data fulfilment | Required | Make sure that all fields are full |
| 5 | Data fulfilment | Optional | Add additional internal data |
| 6 | Data fulfilment | Optional | Add external data |
| 7 | Upload | Required | Upload the data to the model and check its availability in it |
| 8 | Run | Required | Run the model on the data |
| 9 | Results test | Required | Check the results you get from the model and compare to previous results you got from it |

## Annex 4 – Data models, results comparison

|  |  |  |
| --- | --- | --- |
| Method | 75 product DB (Accuracy, Mean Absolute Error) | 3 Product DB (Accuracy, Mean Absolute Error) |
| Linear regression | Acc 531/11897  MAE 14.782 | Acc 96/813  MAE 4.15 |
| SVM | Acc 2741/11897  MAE 29.072 | Acc 166/813  MAE 9.102 |
| Ada Boost | Acc 4081/11897 (34.4%)  MAE 22.605 | Acc 351/813 (43.2%)  MAE 4.194 |
| Bayes | Acc 52/11897  MAE 167.405 | Acc 6/813  MAE 27.861 |
| Logistic regression | Acc 2768/11897  MAE 30.933 | Acc 166/813  MAE 9.102 |
|  |  |  |

1. <https://towardsdatascience.com/disruption-in-retail-ai-machine-learning-big-data-7e9687f69b8f> [↑](#footnote-ref-1)
2. An order can include a few lines, each line is an order of a single product [↑](#footnote-ref-2)
3. In it unique products are being sale [↑](#footnote-ref-3)
4. The summer holiday is define in low traffic. [↑](#footnote-ref-4)
5. See tables in Github <https://github.com/eyalfarkash/TCDS-Final.git> under Data [↑](#footnote-ref-5)
6. A test should be performed since CPU cost less but may take more time, while GPU is faster and more expansive [↑](#footnote-ref-6)
7. Orders in the retail industry are based on packaging, and therefore the ordered quantity is round up to package size. [↑](#footnote-ref-7)
8. For us, since Adaboost is sensitive to outliers and noisy data, getting the best results from it can be an acknowledgment on the data cleansing operation we have done [↑](#footnote-ref-8)