CS4642: Data Mining and Information Retrieval

Kaggle Assignment Report

Group - 21

Team members

K.Kokulan - 120314K

K.Linganesan - 120337H

I.Suventhan - 120649P

M.V.Vithulan - 120678D

1. INTRODUCTION

The assignment report describes the approach the team took to come up with a solution for the problem "Grupo Bimbo Inventory Demand" given in the Kaggle website. According to the Kaggle, the basic introduction to the problem is Maximize sales and minimize returns of bakery goods; problem description given in the competition site shown below.

"Planning a celebration is a balancing act of preparing just enough food to go around without being stuck eating the same leftovers for the next week. The key is anticipating how many guests will come. Grupo Bimbo must weigh similar considerations as it strives to meet daily consumer demand for fresh bakery products on the shelves of over 1 million stores along its 45,000 routes across Mexico.

Currently, daily inventory calculations are performed by direct delivery sales employees who must single-handedly predict the forces of supply, demand, and hunger based on their personal experiences with each store. With some breads carrying a one week shelf life, the acceptable margin for error is small. "

In this competition, we have to develop a model to accurately forecast inventory demand based on historical sales data. In a broad sense we will forecast the demand of a product for a given week, at a particular store. Inventory is delivered weekly to stores along delivery routes. Unsold inventory, from the previous week, is returned. The ideal solution will produce the minimum difference between the current week's delivered inventory and the following week's returned inventory.

2. BACKGROUND

2.1 Client: Grupo Bimbo

The client here is the Mexican bakery product manufacturing company called Grupo Bimbo. Grupo Bimbo has over 1 million stores and 45,000 routes in Mexico. Its annual sales is US$14.1 billion, and it operates in 22 countries. Since the global baking industry is still a growing market, it is important to meet the demands of the customers.

Grupo Bimbo's current inventory calculations are performed by their direct delivery sales employees. They single-handedly predict the forces of supply, demand and hunger based on their personal experiences in each store. For the products which have a shelf life of one week, the error margin is very small regarding inventory planning.

Grupo Bimbo's current model for estimating demand is purely human estimated. Managers estimate demand from the sales of products and their returns and make a judgement for the following week's orders. This human model offers a very high business value for implementing automated machine learning and data science methods to more accurately predict demand.

2.2 Competition rules

The target variable is 'Adjusted Demand', which will be an integer representing the demand of a product, in units. Each model will be evaluated for accuracy using the Root Mean Squared Logarithmic Error (RMSLE). This measure for estimating error will penalize models that under-predict more than a model that over-predicts.

$$\epsilon = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(log(p_i+1) - log(a_i+1)\right)^2}$$

Where:
is the RMSLE value (score)
n is the total number of observations in the data set,
p_i is the prediction of demand
a_i is the actual demand for i
$log(x)$ is the natural logarithm of x

Another limitation of this data set is that demand is approximated by subtracting the supply from the following week's return of inventory. This would naturally allow for an accurate estimate of demand when supply is higher than demand, but would not accurately estimate demand in cases where the product is under-supplied.

3. DATA EXPLORATORY ANALYSIS

3.1 Understand the Data

Data is provided in CSV files and include primary information like train and test data which are contains the data of 9 weeks of sales transaction in Mexico. Likewise, secondary data about products, agencies and clients is also provided. All data points are properly described and, aside from some client naming variances, seems to be very clean. As mentioned before this whole datasets are focused on 9 weeks of sales transaction in mexico, train dataset contains weeks 3-9 (7 weeks) and test dataset contains weeks 10 and 11 (2 weeks). 6 datasets are given:

File name	Description
train.csv	the training set
test.csv	the test set
sample_submission.csv	a sample submission file in the correct format
cliente_tabla.csv	client names (can be joined with train/test on Cliente_ID)
producto_tabla.csv	product names (can be joined with train/test on Producto_ID)
town_state.csv	town and state (can be joined with train/test on Agencia_ID)

The main attributes in the train.csv file are:

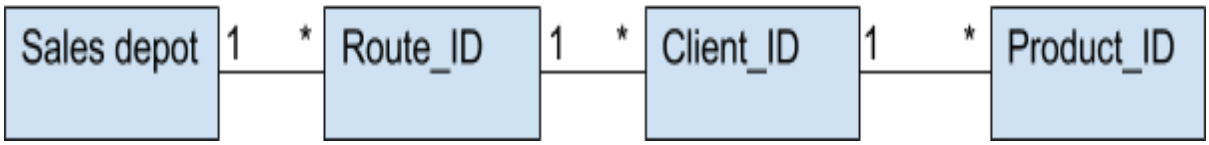
File name	Description
Semana	Week number (From Thursday to Wednesday)
Agencia_ID	Sales Depot ID
Canal_ID	Sales Channel ID
Ruta_SAK	Route ID (Several routes = Sales Depot)
Cliente_ID	Client ID
NombreCliente	Client name
Producto_ID	Product ID
NombreProducto	Product Name
Venta_uni_hoy	Sales unit this week (integer)
Venta_hoy	Sales this week (unit: pesos)
Dev_uni_proxima	Returns unit next week (integer)
Dev_proxima	Returns next week (unit: pesos)
Demanda_uni_equil	Adjusted Demand (integer) (This is the target you will predict)

We dig into the train and test data files for the preliminary exploration and we found following informations.

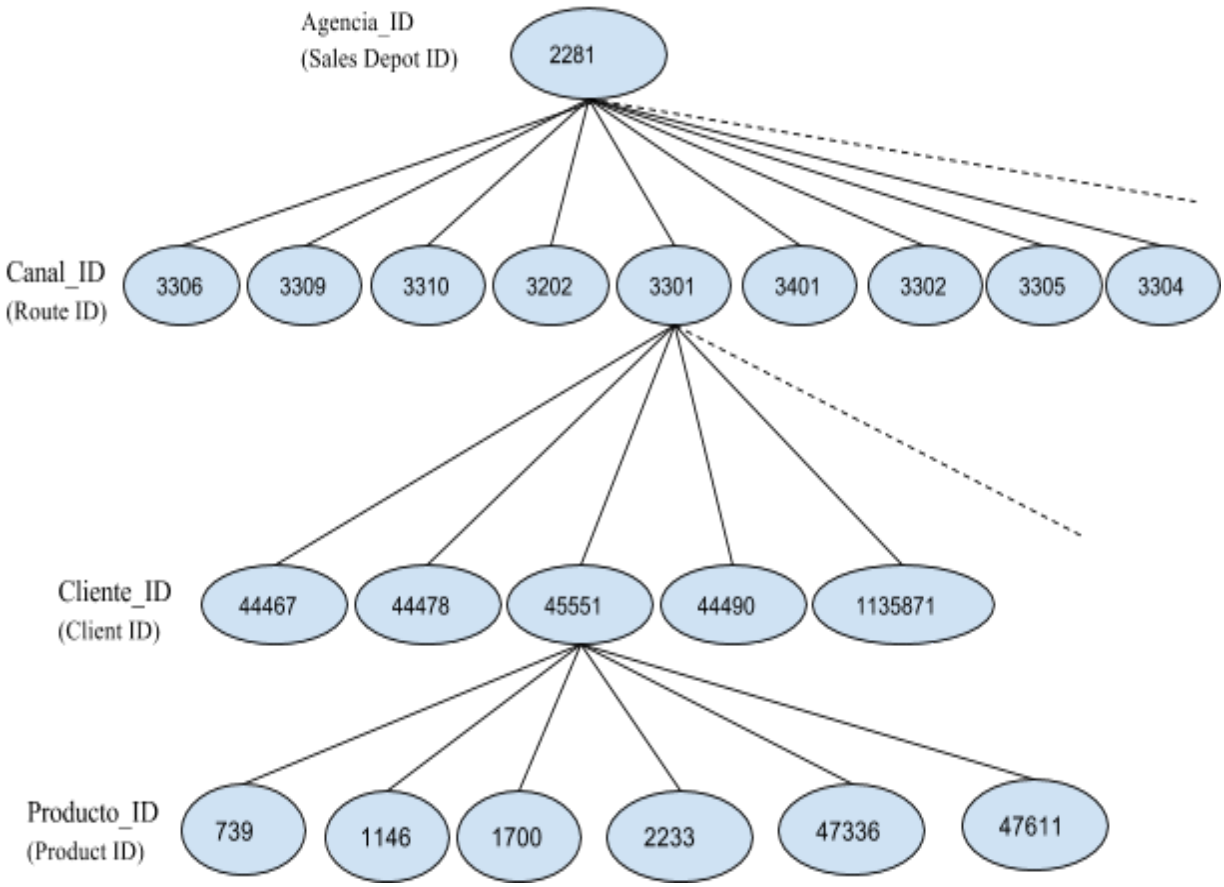
File	Attribute	Unique values	Range of values	Mean	std
train.csv	Semana	7	3-9	5.950021	2.013175
	Agencia_ID	552	1110-25759	2536.508561	4075.123651
	Canal_ID	9	1-11	1.383181	1.463266
	Ruta_SAK	3,603	1-9991	2114.855291	1487.744180
	Cliente_ID	880,604	26-2015152015	1802119	2349577
	Producto_ID	1,799	41-49997	20840.813993	18663.919031
	Venta_uni_hoy	2,116	0-7200	7.310163	21.967337
	Venta_hoy	73,515	0-647360	68.544523	338.979516
	Dev_uni_proxima	558	0-250000	0.130258	29.323204
	Demanda_uni_equil	2,091	0-5000	7.224564	21.771193
File	Attribute	Unique values	Range of values	Mean	std
test.csv	Semana	2	10-11	10.494462	0.499969
	Agencia_ID	552	1110-25759	2504.462764	4010.228289

	Canal_ID	9	1-11	1.401874	1.513404
	Ruta_SAK	2,608	1-9950	2138.014097	1500.391868
	Cliente_ID	745,164	26-2015152015	181912	2938910
	Producto_ID	1,522	41-49997	22163.069696	18698.156308

We can simplify the connection between fields using the diagram given below, which shows each sales depot contains multiple routes, each route contains multiple clients and each client buys at least more than one product.



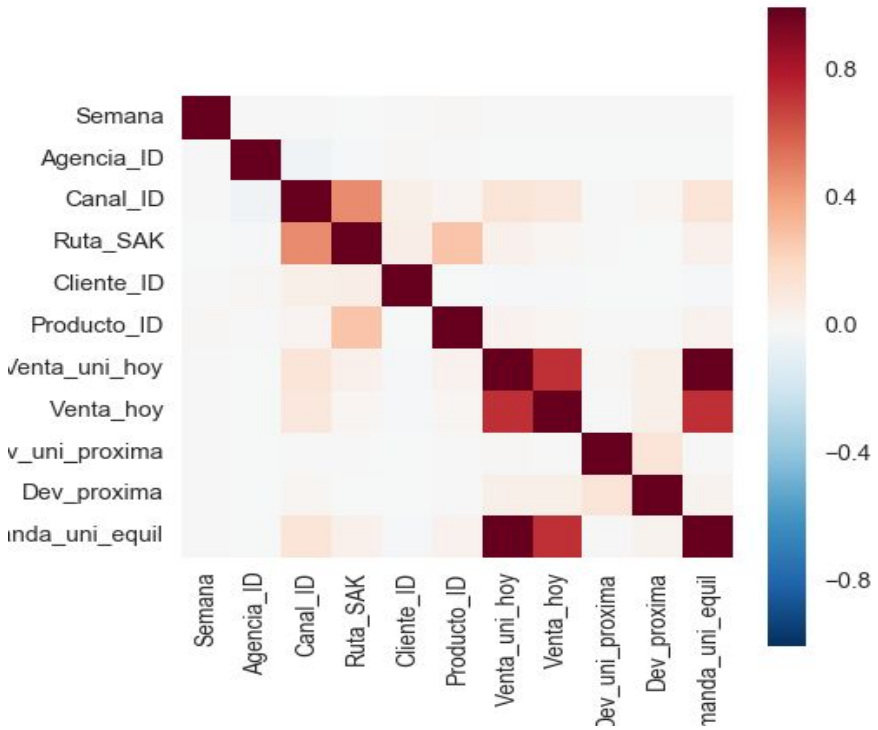
For an example supply chain is given below for Agencia_ID (Sales_depot_ID) equals 2281:



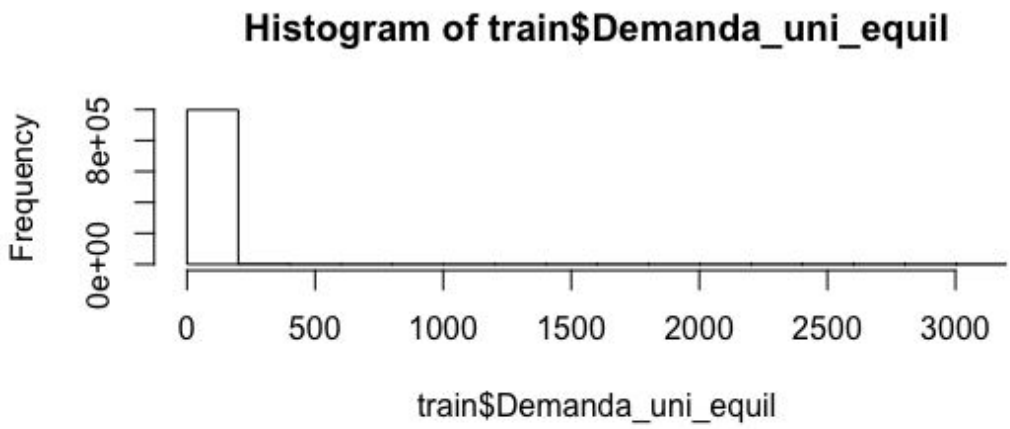
In this tech blog[1], blogger put whole datasets into a database with meaningful english translated table and attribute names. The database schema is given in the blog, which simplifies idea of the relationship between each datasets.

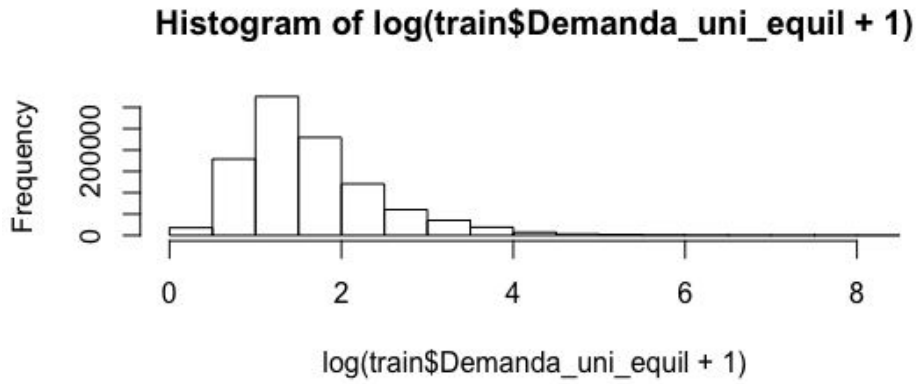
3.2 Exploratory Analysis

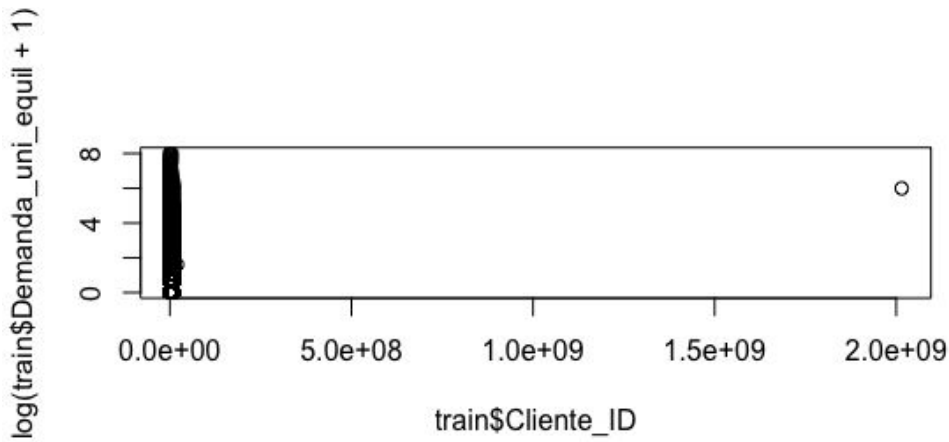
During the initial data exploration step, each of us independently explored the data using summary statistics and visualizations. We examined several relationships between locality and demand and returns.The correlation plot of train dataset is given below. Obviously it shows sales (Venta_uni_hoy) is highly correlated with demands (Demanda_uni_equil) because demand is equal to the sales minus the returns.

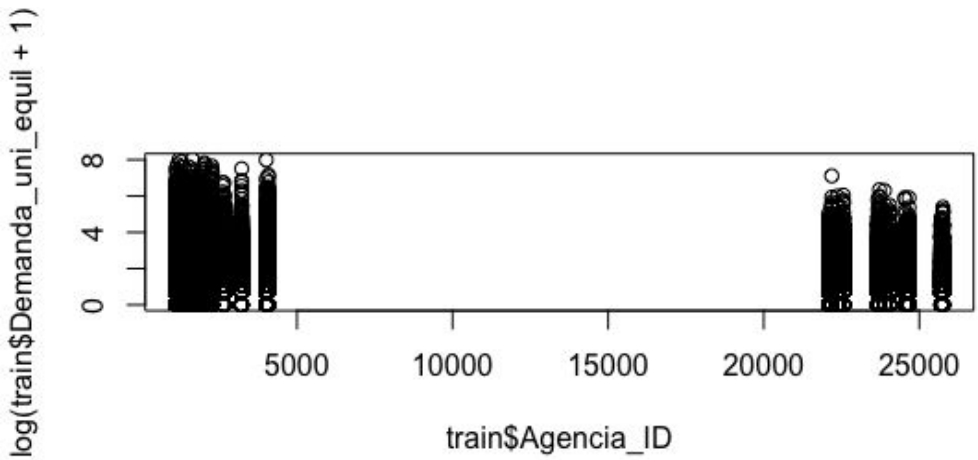


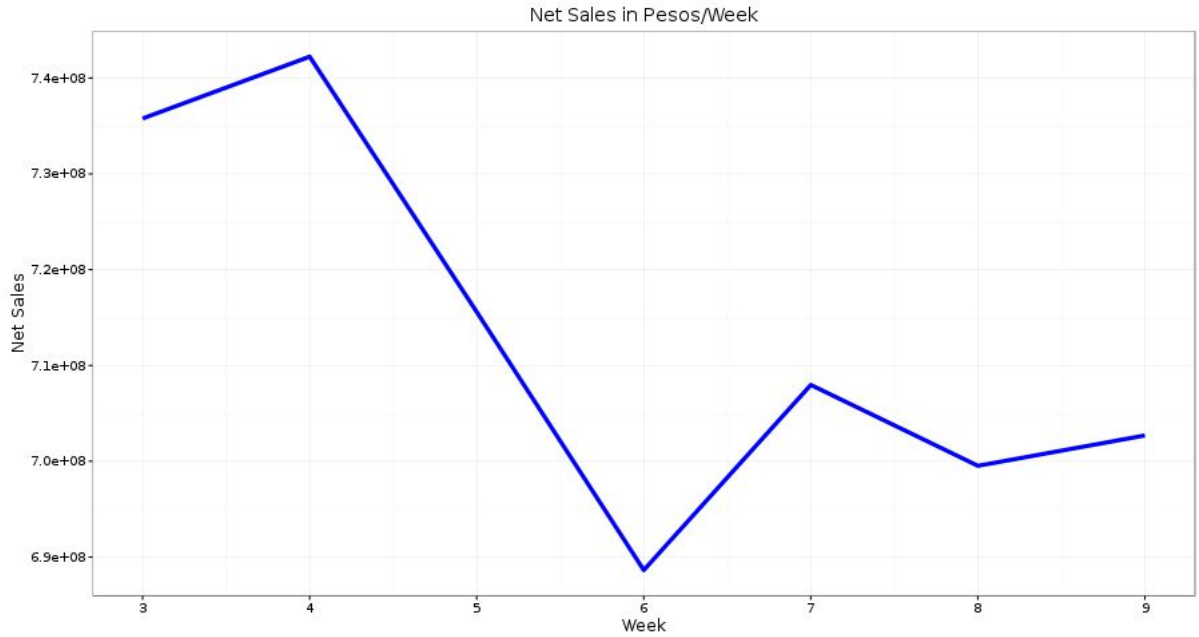
We also analysed on adjusted demand on train set with various attributes. These histograms were generated by using R.




Histogram of log(train$Demanda_uni_equil + 1)
Frequency
0
200000
0
2
4
6
8
log(train$Demanda_uni_equil + 1)


log(train$Demanda_uni_equil + 1)
0
4
8
0.0e+00
5.0e+08
1.0e+09
1.5e+09
2.0e+09
train$Cliente_ID


log(train$Demanda_uni_equil + 1)
0
4
8
5000
10000
15000
20000
25000
train$Agencia_ID



More advanced dataset explorations could be found in these two ipython notebooks [2,5] which are posted in kaggle forums.

4. APPROACH

This section discusses about the approach followed by our team to prediction the adjusted demand. We developed a model, which used log means to calculate the adjust demand for the test data. This model will simply take the log mean demand for particular attribute combinations and use that to predict future demand for those attribute combinations.

First we select train set with Agencia_ID, Ruta_SAK, Cliente_ID and Producto_ID fields, then we replace the adjusted demand values in train set by the its log value, Since we'll be taking the log error, rather than look at the raw data, we'll take the log transformed demand.

We select the crucial field groups in train set, such are:

1) Producto_ID
2) Cliento_ID
3) Producto_ID, Agencia_ID
4) Producto_ID, Ruta_SAK
5) Producto_ID,Agencia_ID,Cliente_ID

For each group given above we created a new feature in the train set which contain the mean value of log transformed demand value. For an example, if we consider the Producto_ID and Agencia_ID as a field group, there will be a new field in the train set called '**Mean_demand_PA**'.

Now our train set has 9 fields, such are:

Producto_ID	Product ID
Agencia_ID	Sales Depot ID
Ruta_SAK	Route ID
Cliente_ID	Client ID
Demand_uni_equil	Log transformed adjusted demand value
Mean_demand_P	Mean value of log transformed demand, grouped by Producto_ID
Mean_demand_C	Mean value of log transformed demand, grouped by Cliente_ID
Mean_demand_PA	Mean value of log transformed demand, grouped by Producto_ID and Agencia_ID
Mean_demand_PR	Mean value of log transformed demand, grouped by Producto_ID and Ruta_SAK
Mean_demand_PCA	Mean value of log transformed demand, grouped by Producto_ID, Cliente_ID and Agencia_ID.

Now we removed any duplicates in the train set grouped by (Producto_ID, Agencia_ID,Ruta_SAK, Cliento_ID).
Based on each new field group in train set, we create a new separate data set and removed the duplicates in each set. For an example there will be a new dataset called '**MeanPA**' created and it contains the unique tuple values of (Producto_ID,Agencia_ID,Mean_demand_PA). So now we have some new sets which are contain unique tuple values. The new sets are:

Dataset name	Fields
MeanP	Producto_ID, Demand_uni_equil(log mean value)
MeanC	Cliente_ID, Demand_uni_equil(log mean value)
MeanPA	Producto_ID, Agencia_ID, Demand_uni_equil(log mean value)
MeanPR	Producto_ID, Ruta_SAK, Demand_uni_equil(log mean value)
MeanPCA	Producto_ID, Cliente_ID, Agencia_ID, Demand_uni_equil(log mean value)

Now we called the test data and merge each new sets with it. For an example merge the 'MeanPA' with test dataset by Producto_ID and Agencia_ID. So now we have the test dataset with 8 fields except Demand_uni_equil (Adjusted demand). So we create a new column in the train set called 'Demand_uni_equil' with **'null'** values for each tuple.

The final and important step in this model is calculate those adjusted demand in the test set. As we mentioned in the data exploratory section, test dataset could contain new data in product and client which will be not appeared in the train set.

We calculate the demand values in test dataset by in the order given below,

test[‘Demand_uni_equil’] = a1*test[‘MeanPCA’].inverseLog() + b1* test[‘MeanPR’].inverseLog() + c1

For any null values found in test[‘Demand_uni_equil’]
test[‘Demand_uni_equil’] = a2* test[‘MeanPR’].inverseLog() + c2

For any null values found in test[‘Demand_uni_equil’]
test[‘Demand_uni_equil’] = a3*test[‘MeanPA’].inverseLog() + c3

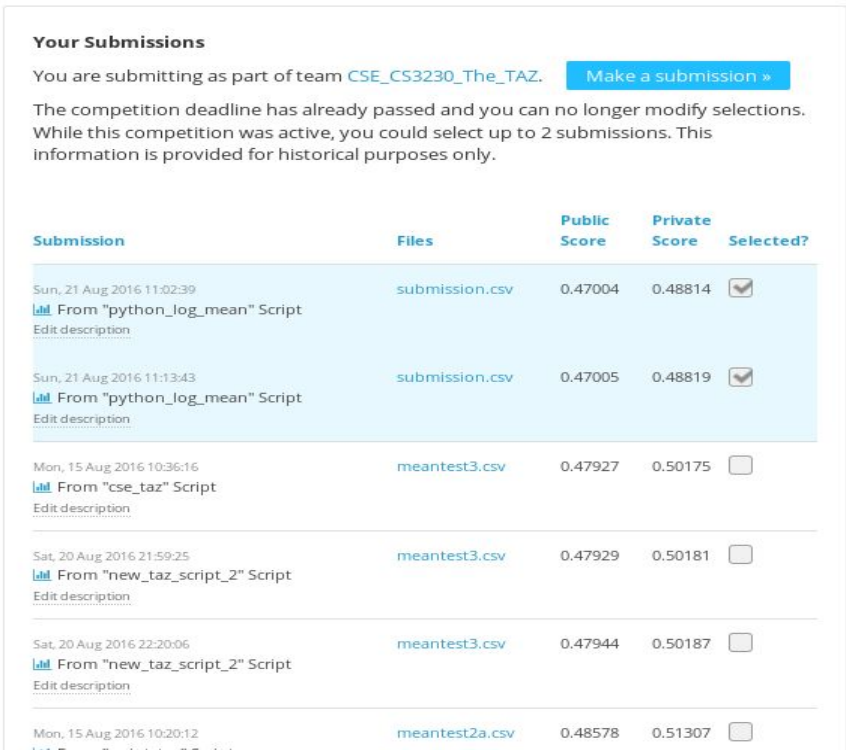
For any null values found in test[‘Demand_uni_equil’]
test[‘Demand_uni_equil’] = a4*test[‘MeanC’].inverseLog() + c4

For any null values found in test[‘Demand_uni_equil’]
test[‘Demand_uni_equil’] = a5*test[‘MeanP’].inverseLog() + c5

For any null values found in test[‘Demand_uni_equil’]
test[‘Demand_uni_equil’] = a6*(Mean demand value of train set).inverseLog() + c6

Those coefficients are predicted by linear regression using weeks 3..8 to fit to week 9 in the train set. The related source code for linear regression can be found below:
https://github.com/linganesan/Programming-Challenges/blob/master/kaggle/grupo_bimbo/linear_regression.py

Evidence of the Work Done
Our submissions and scores while we changed those coefficients mentioned above.



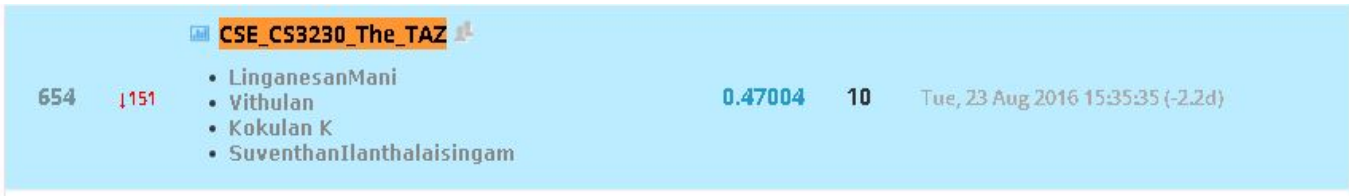
Your Submissions

You are submitting as part of team CSE_CS3230_The_TAZ. Make a submission »

The competition deadline has already passed and you can no longer modify selections. While this competition was active, you could select up to 2 submissions. This information is provided for historical purposes only.

Submission	Files	Public Score	Private Score	Selected?
Sun, 21 Aug 2016 11:02:39 From "python_log_mean" Script Edit description	submission.csv	0.47004	0.48814	☑
Sun, 21 Aug 2016 11:13:43 From "python_log_mean" Script Edit description	submission.csv	0.47005	0.48819	☑
Mon, 15 Aug 2016 10:36:16 From "cse_taz" Script Edit description	meantest3.csv	0.47927	0.50175	☐
Sat, 20 Aug 2016 21:59:25 From "new_taz_script_2" Script Edit description	meantest3.csv	0.47929	0.50181	☐
Sat, 20 Aug 2016 22:20:06 From "new_taz_script_2" Script Edit description	meantest3.csv	0.47944	0.50187	☐
Mon, 15 Aug 2016 10:20:12	meantest2a.csv	0.48578	0.51307	☐

Our final leader board score level,



The related source codes can be found below:
https://github.com/linganesan/Programming-Challenges/tree/master/kaggle/grupo_bimbo

5. DISCUSSION

In the dataset, we are given 9 weeks of sales transaction in mexico. The training dataset consists of 7 weeks and test dataset consists remaining 2 weeks. Our first step in the solution approach was to pre-process the data attributes to match the solution model. Under preprocessing, initially we analyzed the data attributes. Thus, we learnt that rather than blindly using the training data for the prediction model, we must preprocess the data by removing inconsistencies and improving their usability.

We used both Python and R, for our purposes, we mainly used Python with numpy,pandas and scikit and mostly R was used for data exploratory. The main problem we encountered is high memory error, we can't load whole train and test data into the memory still we used a 8GB memory machine, then we found this solution in kaggle forum [6] which is resolve our problem in python using numpy.

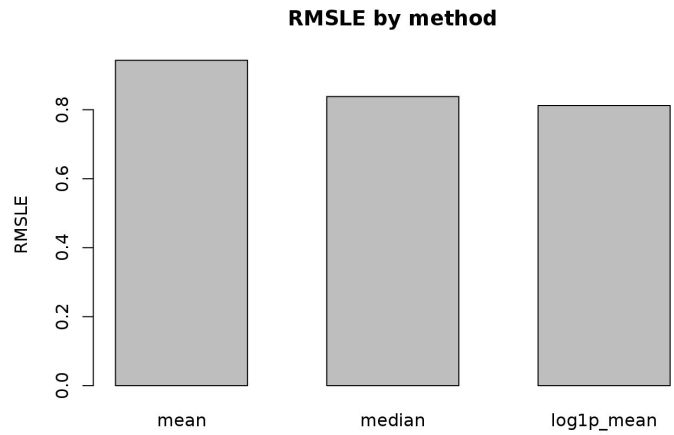
Low memory solution

```
In [4]: # This bit of memory-saving code is from Eric Couto, https://www.kaggle.com/ericcouto/grupo-bimbo
        -inventory-demand/using-82-less-memory/code
        types = {'Semana':np.uint8, 'Agencia_ID':np.uint16, 'Canal_ID':np.uint8,
                 'Ruta_SAK':np.uint16, 'Cliente_ID':np.uint32, 'Producto_ID':np.uint16,'Venta_uni_hoy':np
        .uint32,
                 'Dev_uni_proxima':np.uint32, 'Demanda_uni_equil':np.uint32 }
        train = pd.read_csv('../input/train.csv', usecols=types.keys(), dtype=types)
        print(train.info(memory_usage=True))
        new_names = ["week","depot","channel","route","client","prod","sales_units","return_units_next_we
        ek","demand"]
        train.columns = new_names

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 74180464 entries, 0 to 74180463
        Data columns (total 9 columns):
        Semana               uint8
        Agencia_ID           uint16
        Canal_ID             uint8
        Ruta_SAK             uint16
        Cliente_ID           uint32
        Producto_ID          uint16
        Venta_uni_hoy        uint32
        Dev_uni_proxima      uint32
        Demanda_uni_equil    uint32
        dtypes: uint16(3), uint32(4), uint8(2)
        memory usage: 1.7 GB
        None
```

The type of products they distribute to different regions are different. So, we can intuitively get an idea that to predict the demand of a certain product in a particular store, the region to which the store belongs to and the product's brand should be taken into consideration.

Reason behind why we used log mean rather than mean and median, is the mean of the log transformed demand gets the best RMSLE beating both the mean and median [7]. The median also does much better than the mean - this is because of the demand distribution being asymmetric. Also that the goal of minimizing RMSLE for x is the same as minimizing RMSE for log(x). This means that if we calculate the log of our target, we can use RMSE when modeling to evaluate performance. Once we've created our model, we simply exponentiate the results to undo the log transform and put our predictions back on the correct scale.



Other than using this simple log mean model most of kagglers used XGBoost method for better prediction. Extreme Gradient Boosting (XGBoost) model, which has become one of the most popular algorithms to use for Kaggle competitions. This algorithm is a combination of both a linear model and decision tree algorithm, which supports objective functions including regression, classification, and ranking. XGBoost only works with numeric vectors, so the biggest effort is preparing the data in order for it to run properly [8,9].

This included joining additional indicator variables to the train dataset from the other datasets provided and creating lag demand variables. Our goal was to have the model include: demand, lag of demand from previous weeks, weight, pieces, and brand of the product. Some people used neural network model to predict the future demands, but it seems required high cost in memory and time. Concludingly, a best prediction model should be an ensemble of the different models.

REFERENCES & USEFUL LINKS

Here we have mentioned some reference used in this report which could emphasis our facts regarding the competition and other Useful Links which We've used get information and solution.

[1] https://mtalavera.wordpress.com/2016/06/30/kaggle-grupo-bimbo-inventory-demand/

[2] https://www.kaggle.com/fabienvs/grupo-bimbo-inventory-demand/grupo-bimbo-data-analysis/notebook

[3] https://www.kaggle.com/swetabajaj/grupo-bimbo-inventory-demand/analyzing-data/output

[4] http://online.mrt.ac.lk/mod/forum/discuss.php?d=10754

[5] https://www.kaggle.com/vykhand/grupo-bimbo-inventory-demand/exploring-products

[6] https://www.kaggle.com/ericcouto/grupo-bimbo-inventory-demand/using-82-less-memory/comments#134096

[7] https://www.kaggle.com/apapiu/grupo-bimbo-inventory-demand/mean-vs-medians-a-mathy-approach

[8] http://dmlc.cs.washington.edu/xgboost.html

[9] https://www.analyticsvidhya.com/blog/2016/01/xgboost-algorithm-easy-steps/