Corporación Favorita Grocery Sales Forecasting

Capstone Project

Hemant N Yadav
30th November, 2017

Project Overview

I. Definition

(approx. 1-2 pages)

Project Overview

Corporación Favorita (CF) is one of the Ecuadorian-based grocery retailers, operating hundreds of stores, with over 200,000 different products which includes perishable products also such as egg, milk etc. It is always difficult for them to understand which products to stock at which store and also when? They need good estimate of required inventory at each store so that they can purchase and update inventory accordingly. This estimate eventually benefits two ways to FC. One, help FC not to overstock under popular products specifically perishables and another is have enough inventory in anytime for each product so that customers do not disappointed because not having stock of product

This project is going to apply machine learning model and compare their estimation. Specifically here we will compare performance of Randomforest against deep learning model Long and Short Term Memory (LSTM, a variant of Recurrent Neural Network).Here, models are not much tuned but used with minimum configuration so that fair comparison is possible.

CF has provided data of sales from 2013 to August 2017. Apart from sales data (Productid, Unit Sale, Date, Store Number) other important data is also provided that includes store details, holiday details, product details. Oil prices affects sales seriously so that data is also provided for period of time which can help in understanding hikes in sales. Only oil is not affecting sales, but promotions can also have some impact on it, so data of promotion is provided whenever promotion is running on specific product.

Problem Statement

Sales forecasting is remained the one of the important requirement of any grocery stores. There are many statistical models used for task of predicting sales such as ARIMA[1]. Thos models mainly based on time so are univarient timeseries preditions. CF sales are affected by many other factors which promotes application of more complex model.

Here, predicting the sales for each product is considered as main problem to be solved. Generally, grocery stores use traditional models for prediction with little data taken in consideration. It is important to evaluate which would be better for forecasting sales of CF among two different kinds of models. One is time series based model and another is regression based on causality. Here, comparison is using LSTM for time-series forecasting and Random forest for causal forecasting. This will allow as understanding which is better for forecasting of sales.

Planned Project workflow is as bellow:

1. Dataset Size reduction: As training data is quite large consisting of 125497040 observations which when loaded in memory takes more than 50 GB. So, first step would be to reduce dataset using data transformation or sampling.
2. Data analysis: This includes study the effect of each factor on sale such as,
3. Kind of change in sale due to promotion
4. Effect of oil price changes
5. Sale volume on each store
6. Highest consumer's city
7. Data preprocessing: It includes,
8. Treatment of null, negative or NaN values in each column.
9. Removal of unnecessary features such as transaction id.
10. Concatenation of data such as store number and sales data to find sales at each store.
11. Experimenting with models: It includes
12. Data preparation which comprises onehotencoding for categorical features, time series data modeling etc to train in LSTM. For random forest division of test, train is needed.
13. Removal of unnecessary features such as transaction id.
14. Concatenation of data such as store number and sales data to find sales at each store.

15. Also, building randomforest with different estimators.
16. Testing the Models: Over here I will compare model with themselves for different hyperparametes and also with each other. Also, modifications to step 4 is expected by this comparison.

Metrics

The method being used as Evaluation metrics is the Normalized Weighted Root Mean Squared Logarithmic Error (NWRMSLE),

This metric is suitable when predicting values across a large range of orders of magnitudes. It avoids penalizing large differences in prediction when both the predicted and the true number are large: predicting 5 when the true value is 50 is penalized more than predicting 500 when the true value is 545[2].

Calculated as follows:

$$NWRMSLE = \sqrt{\frac{\sum_{i=1}^{n} w_i(\ln(\hat{y}_i + 1) - \ln(y_i + 1))^2}{\sum_{i=1}^{n} w_i}}$$

Where for row i,

yhat(i) is the predicted unit_sales of an item

y(i) is the actual unit_sales

n is the total number of rows in the test set.

W(i) is a weight for each item. Perishable items are given a weight of 1.25 where all other items are given a weight of 1.00

II. Analysis

(approx. 2-4 pages)

Data Exploration

Dataset for this project contains following datasets

train.csv: It contains information about sales of product date and store wise and has following features.*[id, store_nbr, item_nbr, unit_sales, onpromotion]*

	id	date	store_nbr	item_nbr	unit_sales	onpromotion
0	0	2013-01-01	25	103665	7.0	NaN
1	1	2013-01-01	25	105574	1.0	NaN
2	2	2013-01-01	25	105575	2.0	NaN
3	3	2013-01-01	25	108079	1.0	NaN
4	4	2013-01-01	25	108701	1.0	NaN

Sample Data

Where,

id is a unique identification of each transaction

date is a date of transaction

store_nbs is store number particular from where sale happened

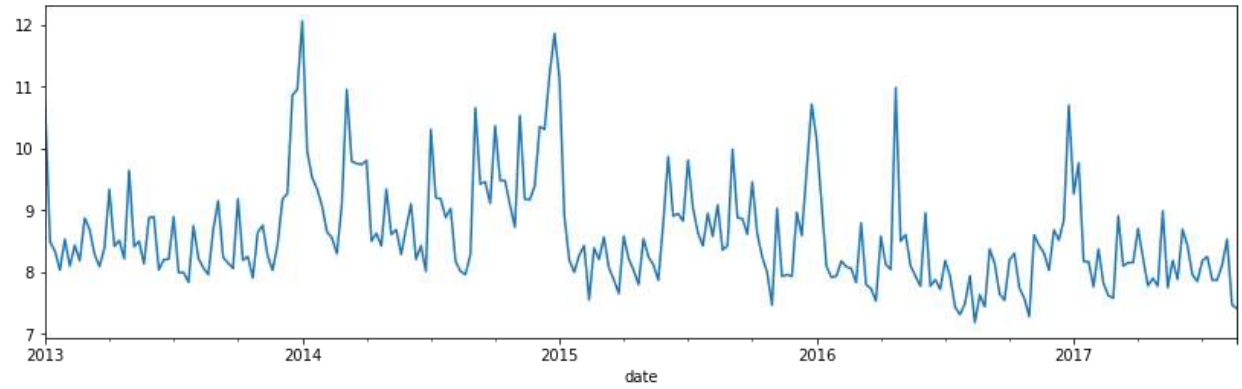
item_nbr is a product which is being sold

unit_sales is a number of sales for a particular product on particular date

It contains total 125497040 transactions for the duration of 2013-01-01 to 2017-08-15. It has onpromotion column which has most values(16%) NaN. NaN in onpromotion means not on promotion.

Also plot below of unit sales indicates hikes at end of every year and also regular hikes in between. There are total 54 stores and 4036 products being sold between years 2013 to 2017.

unit_sales also contains some negative values which contributes to return of a sold product.



unit_sales vs time

store.csv: It contains details about each store which include [store_city,State, type, Cluster]

Store_city and state are location of store.

store_type: there are five different type of stores named as A,B,C,D,E.

Cluster is a grouping of similar stores.

Total 22 cities are in list where stores are located. Apart from that total 16 states and 17 are clusters of similar stores.

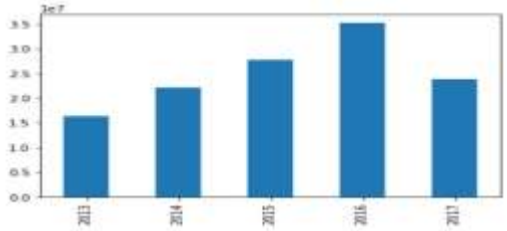
oil.csv: Sales are affected by oil as Ecuador is an oil-dependent country so this data contains daily oil prices for given duration.

items.csv: It provides details about individual product. Details include [Family, class, perishable]
Family gives category of product,
Class is product classification on the shelves of stores,
Perishable if true means product is perishable such as milk.

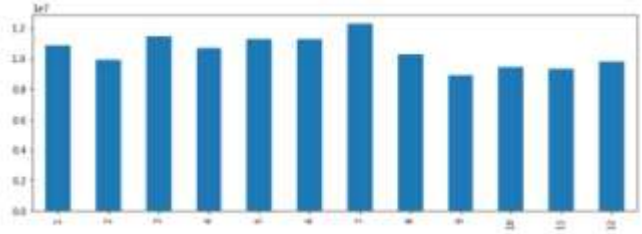
holiday_events: It provides details about holiday and special events. In this data transferred type of holiday is also given. Transferred means holiday was no scheduled on calendar day but moved to some other day by government. As well as some additional holiday declared by government also given as they might not be on calendar day.

Exploratory Visualization

- **Sales on timeframe:** It will be useful to understand how sales are distributed over year, month, and day for understanding effect of time on sales.

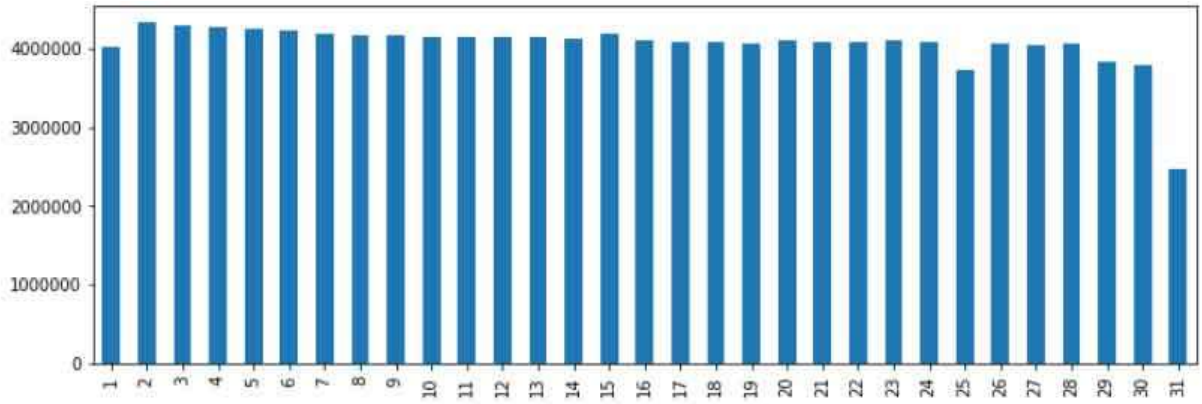


Sale on each year

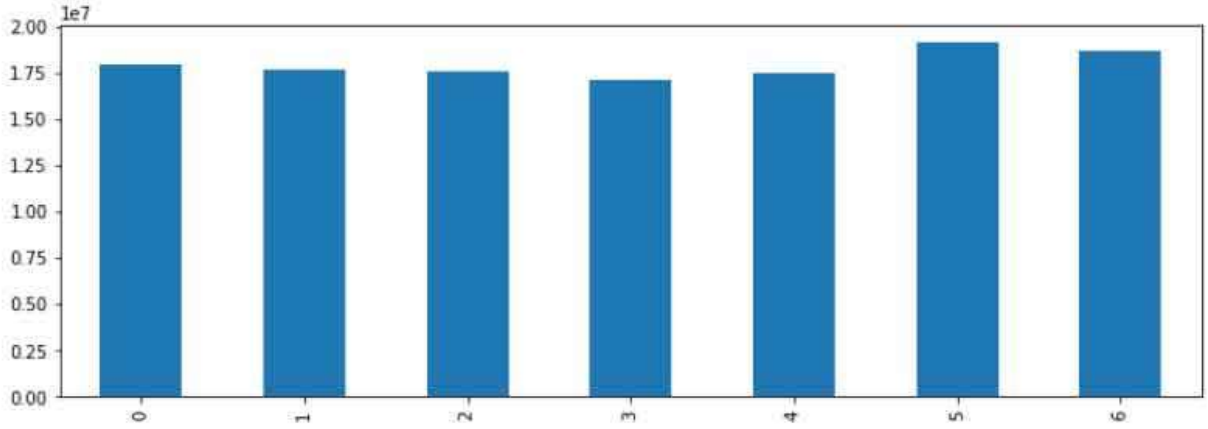


Sale in each month

Above chart on each year clarifies sale is increases each year. On the other hand maximum sale is happening in the month of July.



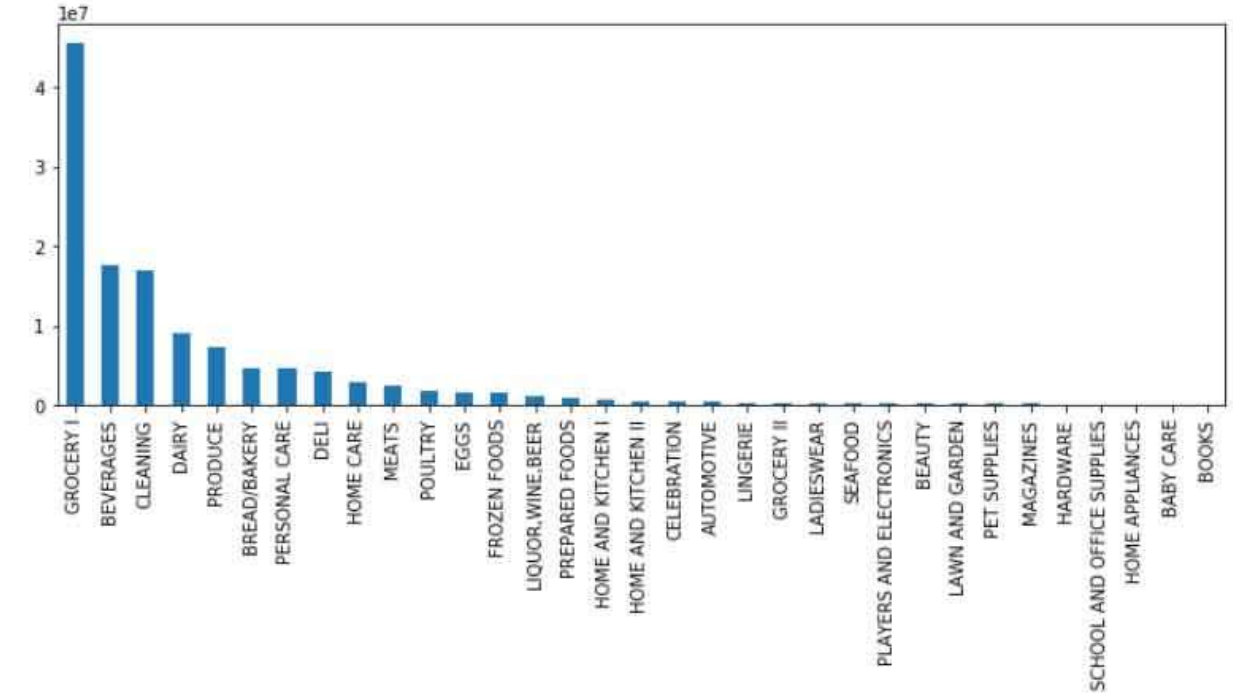
This above chart of sale by each day of month indicates that there are less sale happing in at last and first day of each month.



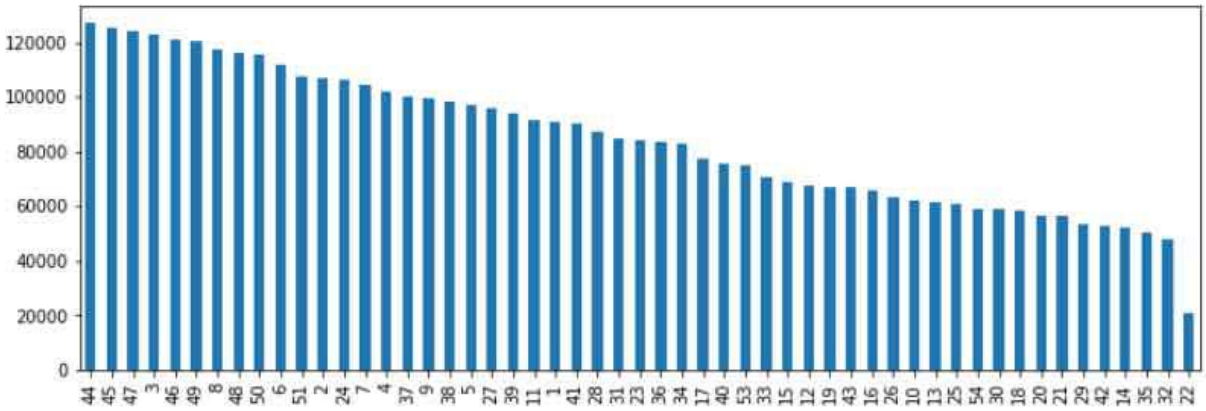
Above chart indicates that there is no much effect of weekend on sale.

- **Sales of each product:**

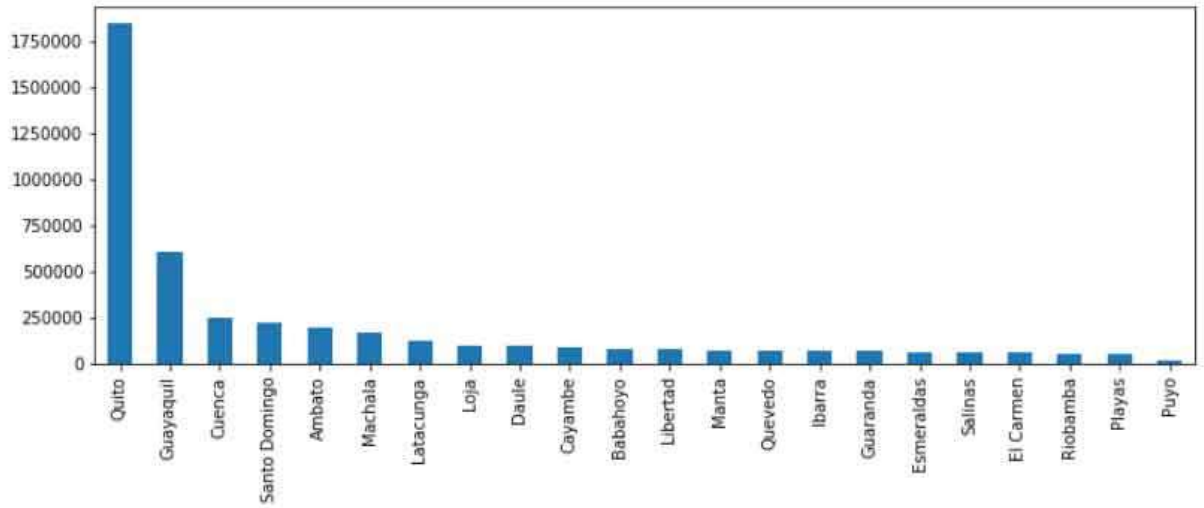
  Following plot says that it has maximum sale of Grocery I type as well as many products has almost no sale.



- **Sales on each store:** This plot indicates that sale is also dependent on store number.



- **Sales at each city:** From below chart it is clear that sale is highly concentrated on some city.



Algorithms and Techniques

Here, two different algorithms going to be used are RandomForest and LSTM(Long and Short Memory). Over here we will use most basic versions of both of them so that they could be compared.

Random Forest regressor:

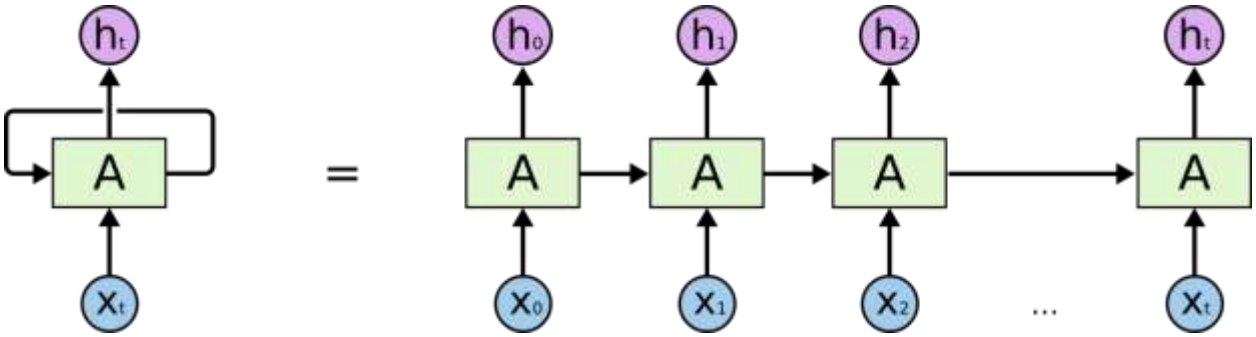
It is a Decision tree based bagging technique. It creates multiple trees while training and returns the mean of all trees as prediction [3]. It is considered as one of the most powerful algorithm for classification but has some deficiency in regression task. It is like a blackbox where we have little control over what is happening inside but there little parameters that can be used to tune it.

Random forests are based on divide and concur strategy to reduce the variance and improve performance. The basic idea behind random forest is to hide inefficiencies of weak learner and combine them with strong learners.

Random Forest basically creates decision tress on randomly selected data from original dataset. This will help in overcoming problem of over-fitting by decision tree by averaging over-ffitting out.

LSTM(Long Short Term Memory):

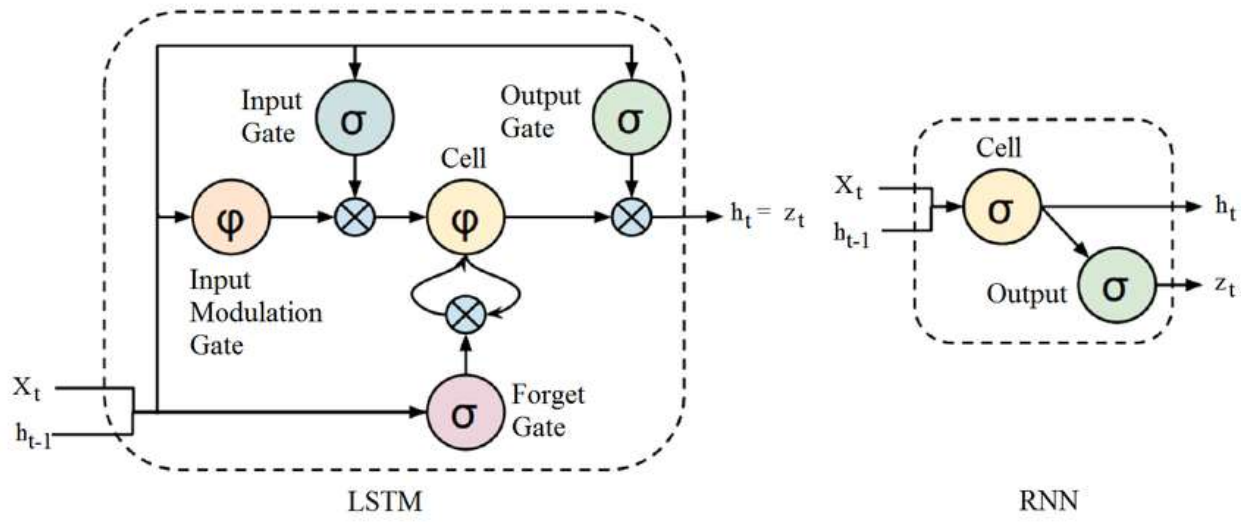
LSTM are special case of Recurrent Neural Network. Recurrent neural networks are known for its feedback mechanism which is again suffering from gradient decent. The concept of RNN is pretty simple as illustrated in following figure. The figure shows that at each step previous step is feed along with input to next step. Concept behind feeding back previous step is to create some sort of remembrance. For example, if we have information of former word we can predict next word. This can be illustrated by predicting last word in sentence “the clouds are in the *sky*,” as this is pretty obvious dependency on previous words. RNN good at predicting using recent past but they tend to be poor in understanding long term dependencies. For example, predicting last word in a long paragraph. It is because of gradient descent which reduces by every iteration during back propagation.



RNN Neuron working [4]

LSTM addresses this issue of vanishing gradient and implements more complex structure of memory cell at each node. Specifically LSTM node composed of three gate (mainly, there some variations to it also) namely, input gate, output gate, forget gate and memory cell.

Understanding LSTM node by following diagram.



LSTM Vs RNN Cell[5]

Input Gate: It has weight matrix that calculates what to store in memory cell from input.

Output Gate: It is analogues to where to emphasis from whole input.

Forget Gate: Though name is forget, it decides what to preserve and what to delete from memory cell.

Memory Cell: It is basically a memory matrix that is kind of hidden state in normal RNN.

Finally, each gate is working just to maintain gradient at each iteration of Neural Network. Leaning of each of matrix is a gain a part of learning from neural network.

Benchmark

As I was expecting to consider Random forest as a benchmark and LSTM to perform better as LSTM is Neural Network based model but LSTM is performing worse if lot of causality is present in data and prediction is made just as univariant time series.

III. Methodology

(Approx. 3-5 pages)

Data Preprocessing

Following steps for preprocessing data.

1. Removing Negative sales
2. Removing NaN from onpromotion
3. Preprocessing Holiday Events
4. Preprocessing Perishable Items
5. Converting Categorical to Numeric
6. Merging data
7. Transforming date and encoding object data
8. Train test split
9. Time Series modeling LSTM
10. Shape conversion for LSTM

Implementation

- Random forest

  Random Forest is one of the most complex models employed here with its default parameter configuration. As, here we are comparing it with LSTM parameter tuning is not applied. Following is a detail of fitted model.

  RandomForestRegressor
  (bootstrap=True, criterion='mse', max_depth=None,
  max_features='auto', max_leaf_nodes=None,

min_impurity_decrease=0.0, min_impurity_split=None,

min_samples_leaf=1, min_samples_split=2,

min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,

oob_score=False, random_state=None, verbose=0, warm_start=False)

- LSTM:

  It is a variant of RNN. Here we have used single layer of LSTM with on dropout layer and one fully connected dense layer. Experiments have been conducted to stacking multiple layers of LSTM, but it was not improving so to keep it simple and efficient only one layer is kept. Activation layer added is linear and LSTM has internal gates based on tanh and sigmoid. Optimizer is RMSPROP with its learning rate 0.001 and MSE is used as loss function. LSTM is experimented with different batch sizes amount all them 10 is selected for memory efficiency. Total epoch is 15 as after that no improvement is found.

  Following is network architecture.

  ```
  model = Sequential()

  model.add(LSTM(50, input_shape=(X_train_values.shape[1], X_train_values.shape[2])))

  model.add(Dropout(0.3))

  Dense(model.add(Dense(1)))

  model.add(Activation('linear'))

  model.compile(loss='mse',optimizer='rmsprop')
  ```

Refinement

There are multiple refinements applied during implementations which includes followings,

1. *Started with all data:* It was become resource eater to take all data and train model. So data as reduced by sampling. Also one more attempts was to select data of specific month only so that it could be used to predict month of final year but it was not worked due to over fitting.
2. *Data without city and state*: Initially it was considered not to use these data but addition of those parameters affected the model most in some city as seen from data exploration part.
3. *Data of particular month for all years*

4. *Data of single product from all years*
5. *Data of single product from single year*
6. *Data of single product from all year*

Finally, data of one product for whole period from 2013 to 2017 is used for fitting in the model. Product selected is not dependent on any store or city but it is estimated how much would be sale of that product in future.

Though there is scope for further improvement in the model itself but for the sake of simplicity I haven't documented them. Also, changes in model parameter was not giving enough improvement in loss so it might be possible that it may required lot many changes including many layers of LSTM, and huge data.

IV. Results

Model Evaluation and Validation

For random forest, changes in random state, tree size and maximum depth is adjusted with all data, data of months only (Data of January is selected from all year and used to predict January only), and on sampled data (0.1, 0.01, 0.001, 0.05, 0.5, .005). Considering all these experiments one giving minimum NWRMSLE (As set evaluation parameter is selected) is with 0.01 sampled data and default parameters. But, final model selected is based on single product over all time period. Which gives following results.

RF accuracy: TRAINING 0.771417463539

RF accuracy: TESTING 0.748201438787

NWRMSLE RF 0.322782339177

It is based on following model parameters

max_features = "auto",

min_samples_leaf = 50,

n_estimators = 100,

random_state =50,

oob_score =True

2013 to 2016 data were considered as training data and 2017 data were considered as testing

data.

Apart from that more LSTM layer were added which resulted in over-fitting. Also, use of activation other that linear was applied (Sigmoid and tanh) but as both of them are already part of the LSTM node,

they do not had any improvement. Number of Epoch from 50 to 1 is tested and 10 is looking enough as beyond that no any improvement in loss was looking. LSTM was considered Sensitive to batch size , the model is tested for batch sizes from 150 to 1 and 3 is giving maximum loss and below 75 batch size have not shown any improvement is model but remains stable.

Final model selected was following based on lag of 7 observations.

```
model = Sequential()
model.add(LSTM(4, input_shape=(1, look_back)))
model.add(Dense(10))
model.add(Dropout(0.2)) // To prevent overfitting
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam',metrics=['accuracy'])
```

Finally, for selected product id 502331 Random forest gives better results than that of LSTM.

Justification

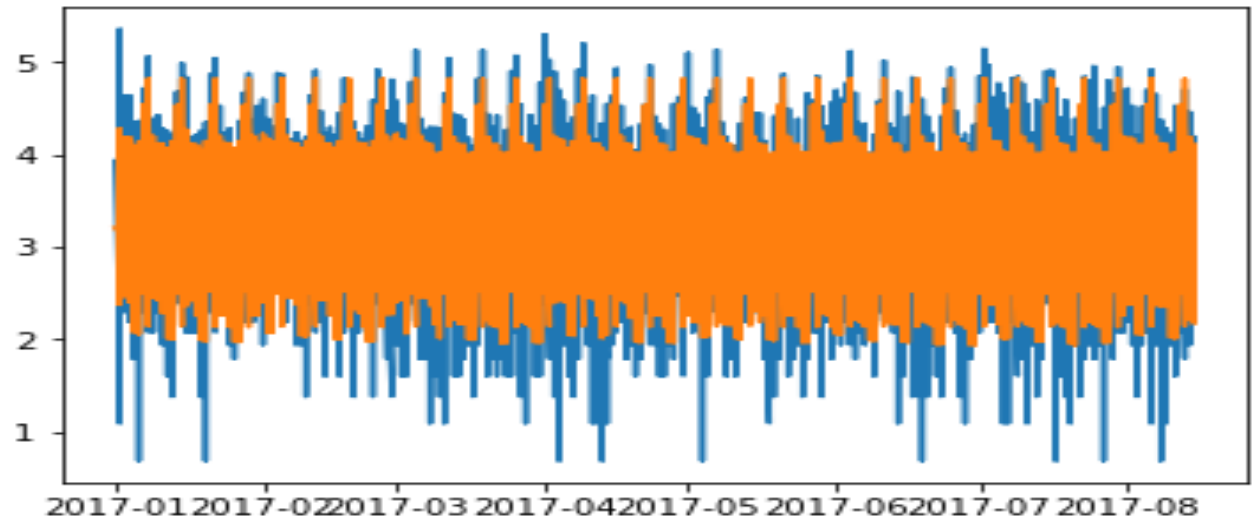
Considering the results it is clear that random forest outperforms to LSTM. This is because this data has mare causality than time dependency. Training LSTM with multiple timestep also not improving results although result of LSTM is acceptable but random forest gives better accuracy and error.

V. Conclusion

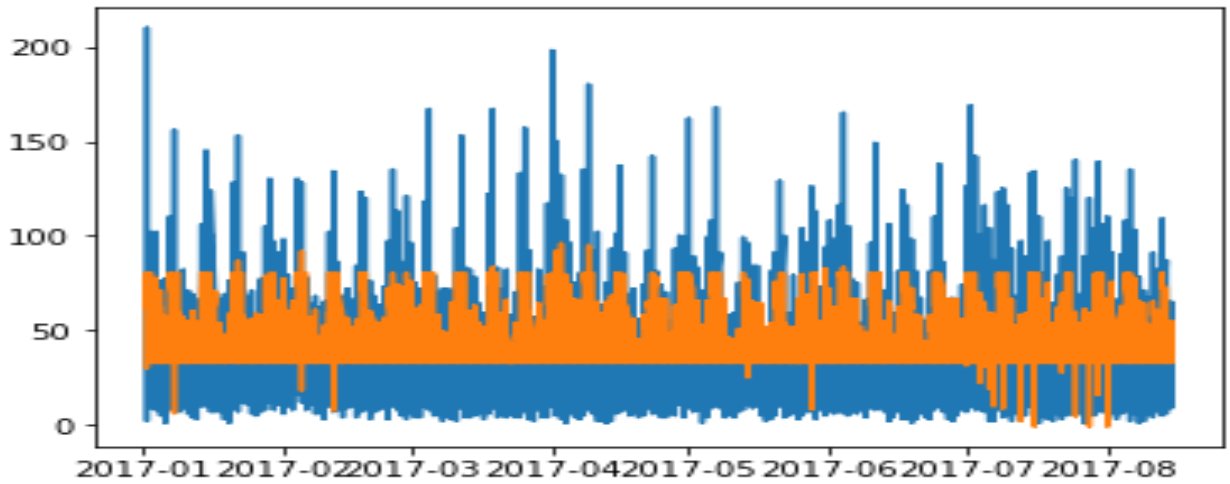
Free-Form Visualization

Following visualizations show that randomforest fits better to testing data than that of LSTM.

Following is a actual vs prediction plot.



Following is a visualization of actual vs predictions.



Reflection

This project was taken from Kaggle competition when I was novice to such competitions but due to my interest and only objective to build kaggle projects I had selected this project for my CAPSTONE. There were different phases I had gone through this project as below but overall I find it is very difficult for novice to compete.

- *I had read descriptions of project and tried to search for solutions which was really wrong direction for starting a project.*
- *From first mistake I learned that it is must to first understand data, So I started to visualize different aspect of data in which I found many correlations between different variables such a oil price change, holiday, weekends and also growth of overall sales by each year.*
- *Based on analysis I come to know that this data has two aspects one is causality and another is time dependency and based on that I had decided to make proposal which would be better forecasting model and I have selected RandomForest for causality based forecasting and LSTM for time-series forcasting.*
- *This was not what I found as a problem but when I had tried to fit LSTM or RandoForest I come to know that my existing resources will not work so I had employed google cloud service and also azure cloud services. This was also not enough because even after employing 50 GB of memory and 8 Core processor data was not loaded fully.*
- *I have decided to reduce data and use only portion of data and experimented lot with many combination of data that looks meaningful. Finally I have selected on most selling product to predict over time and based on features.*
- *Apart from experiments with data the hardest part I found in this project was to train LSTM. I had tried lot many ways to reduce over fitting and under fitting which had taken most portion of total experiments in this project.*
- *Finally, I ended up with reasonably good model with LSTM and also with RandomForest and found that this data has lot of causality which is why time series furcating will underperforms.*

Improvement

This project is about understanding application of two different methods for predicting sales. One used is random forest which is state of the art boosting algorithm based on Decision tress. Random-Forest is used to check causality based prediction. In these experiments, I have used it for one single product and it is performing quite well and better than LSTM. Random forest with k-fold crosses validation for better parameter tuning.

LSTM is used with lag of 7 observations which again need improvement. With availability of better of high end resources w can create multivariate time series in LSTM. During experiments I have observed LSTM will improve with more number of data and quite sensitive to batch size but tuning both of these parameters need huge amount of time and also processing

power. I have tried many layers and nodes of LSTM but I could not see improvement but experiments combination of CNN over here might be helpful in learning causality of such sales data.

In both of the experiment on can also improve in number of data as total of 5 GB data is available to make model learn better. I haven't tried fb prophet here, but it is also considered one of the good candidates for time series forecasting.

References

[1]. https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average

[2]. https://www.kaggle.com/c/favorita-grocery-sales-forecasting#evaluation

[3]. https://en.wikipedia.org/wiki/Random_forest

[4]. http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[5]. https://github.com/llSourcell/LSTM_Networks/blob/master/LSTM%20Demo.ipynb