Walmart Weekly Sales Forecasting

Sasmit Khokale **|** Capstone Project **|** 4/13/2019

PROJECT OVERVIEW

Retail giant Walmart is an American multinational retail corporation that operates a chain of hypermarkets, discount department stores, and grocery stores. For retail of this scale it is crucial to have best possible demand numbers or sales forecast available at each given time. For better forecasting it is important to invest in data science efforts.

Knowing better forecast numbers will help Walmart in following ways –

1) Balancing demand and supply, this consists of replenishing products at right times, right places to ensure availability when customer needs it.
2) Plan in advance by partnering with manufacturers to have the products available at given time

PROBLEM STATEMENT

For this project, goal is to use Walmart's historical sales data to develop models that predicts future sales, steps include –

1) Download data from available source -

   https://www.kaggle.com/iamprateek/wallmart-sales-forecast-datasets

2) Perform Exploratory analysis by visualizing and analyzing important dimensions
3) Preprocess data by adding more dimensions as needed
4) Train different models using preprocessed data and choose best model
5) Predict sales for future using test data
6) Use test data to analyze final model performance

We can apply supervised learning to this use case; specifically this is a regression problem where we will be predicting future sales based on other known factors

EVALUATION METRICS

Root Mean Squared Error is a common evaluation metric in this context as we are trying to predict the value which should be equal to or very close to the actual value.

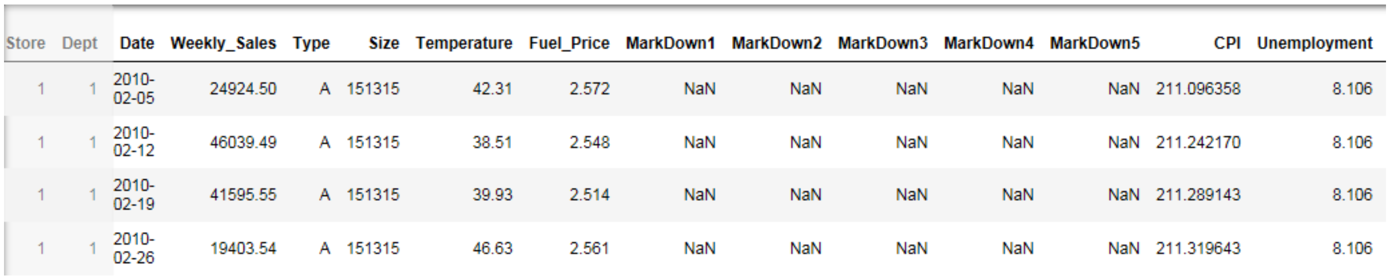
RMSE = 1/n $\sum$ (Predicted value – Actual value)^2

RMSE was preferred over the other evaluation metrics since errors are squared before they are averaged, RMSE gives a relatively high weight to large errors. Thus RMSE is more useful when large errors are undesirable which is exactly what is desired in this problem.

ANALYSIS

DATA EXPLORATION

The data had sales information for about 45 stores along with some feature characteristics for each store. Each store has been classified into either of 3 types: A, B or C. Each store also has about 99 departments, weekly sales of which have been provided. The Sales information was merged with the store features' information. The data final has a total of about 421K rows. A sample of the dataset is as below:



Store	Dept	Date	Weekly_Sales	Type	Size	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment
1	1	2010-02-05	24924.50	A	151315	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106
1	1	2010-02-12	46039.49	A	151315	38.51	2.548	NaN	NaN	NaN	NaN	NaN	211.242170	8.106
1	1	2010-02-19	41595.55	A	151315	39.93	2.514	NaN	NaN	NaN	NaN	NaN	211.289143	8.106
1	1	2010-02-26	19403.54	A	151315	46.63	2.561	NaN	NaN	NaN	NaN	NaN	211.319643	8.106

Below are the two types of variables in the dataset:

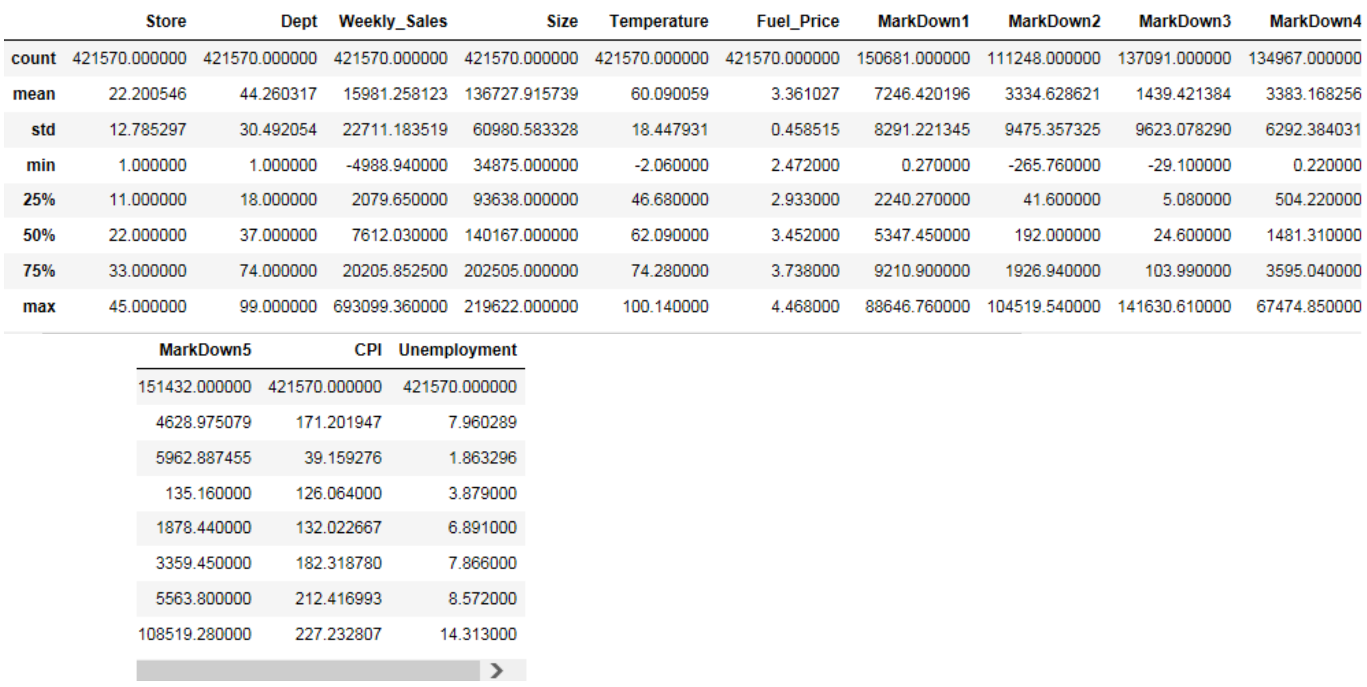
Continuous Variables -

1) Historical sales
2) Temperature
3) Fuel Price
4) Mark down price (if any)
5) CPI – Consumer Price Index
6) Unemployment rate

Categorical Variables -

7) Whether there was holiday or not
8) Store Type
9) Markdown exist (Yes/No)

The data was initially statistically explored using the describe() function. From the output it was noticed that stores did not have markdowns every week, which ideally should be the case and so had null values. All other columns did not have any missing values. It was also noted that the range of Weekly Sales spread from $15981 to $693099. Knowing this range will be helpful in determining if the RMSE value we obtain at the end is within the range. Output of the describe function looked like below:



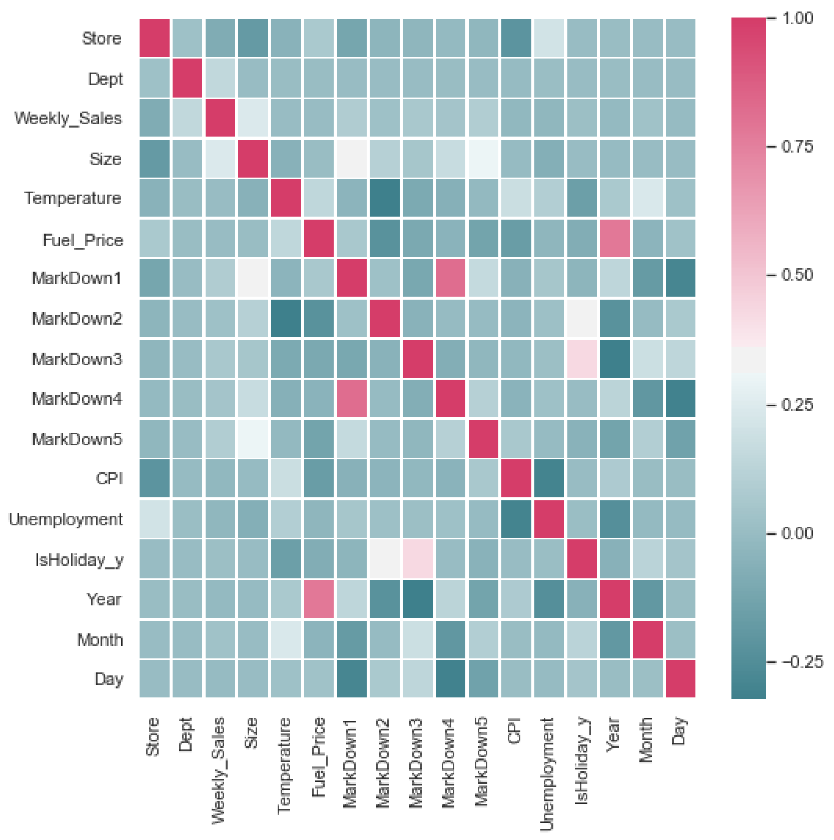
	Store	Dept	Weekly_Sales	Size	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment
count	421570.000000	421570.000000	421570.000000	421570.000000	421570.000000	421570.000000	150681.000000	111248.000000	137091.000000	134967.000000	151432.000000	421570.000000	421570.000000
mean	22.200546	44.260317	15981.258123	136727.915739	60.090059	3.361027	7246.420196	3334.628621	1439.421384	3383.168256	4628.975079	171.201947	7.960289
std	12.785297	30.492054	22711.183519	60980.583328	18.447931	0.458515	8291.221345	9475.357325	9623.078290	6292.384031	5962.887455	39.159276	1.863296
min	1.000000	1.000000	-4988.940000	34875.000000	-2.060000	2.472000	0.270000	-265.760000	-29.100000	0.220000	135.160000	126.064000	3.879000
25%	11.000000	18.000000	2079.650000	93638.000000	46.680000	2.933000	2240.270000	41.600000	5.080000	504.220000	1878.440000	132.022667	6.891000
50%	22.000000	37.000000	7612.030000	140167.000000	62.090000	3.452000	5347.450000	192.000000	24.600000	1481.310000	3359.450000	182.318780	7.866000
75%	33.000000	74.000000	20205.852500	202505.000000	74.280000	3.738000	9210.900000	1926.940000	103.990000	3595.040000	5563.800000	212.416993	8.572000
max	45.000000	99.000000	693099.360000	219622.000000	100.140000	4.468000	88646.760000	104519.540000	141630.610000	67474.850000	108519.280000	227.232807	14.313000

The data was then divided into train and test data for further analysis and running the models on.

EXPLORATORY VISUALIZATION:

Correlation Matrix:

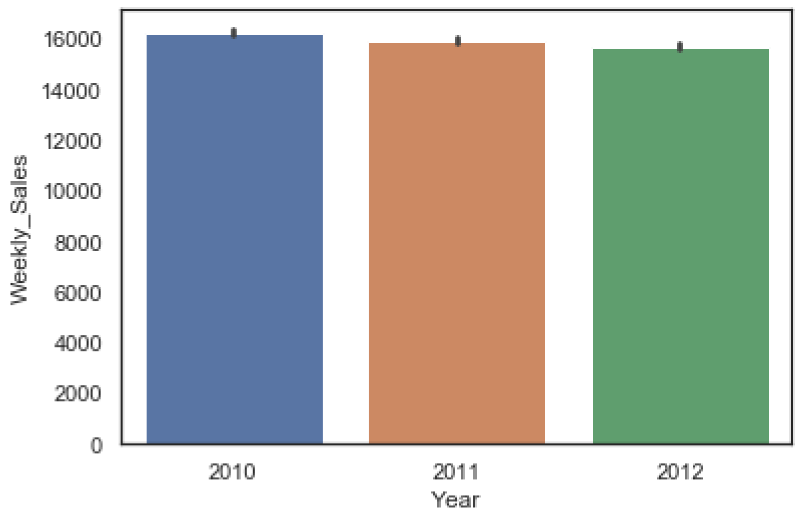
The first and foremost visualization used to explore the data was creating a correlation matrix to understand correlation between the different features of the data.



From the matrix, it was interesting to note that Department and Weekly Sales were positively correlated- meaning bigger department numbers had higher sales. It was also noted that discounts are correlated and that unemployment was inversely co-related to the Consumer Price Index. Is_Holiday flag was noted to be positively correlated with Markdown_3 and so were Fuel Prices with Year, indicating Fuel prices increased with every consecutive year. The Fuel Prices as expected were negatively correlated to the Markdown flags. Unfortunately, there wasn't enough relationship between fuel prices, Temperature, Holidays with the Weekly_Sales numbers.

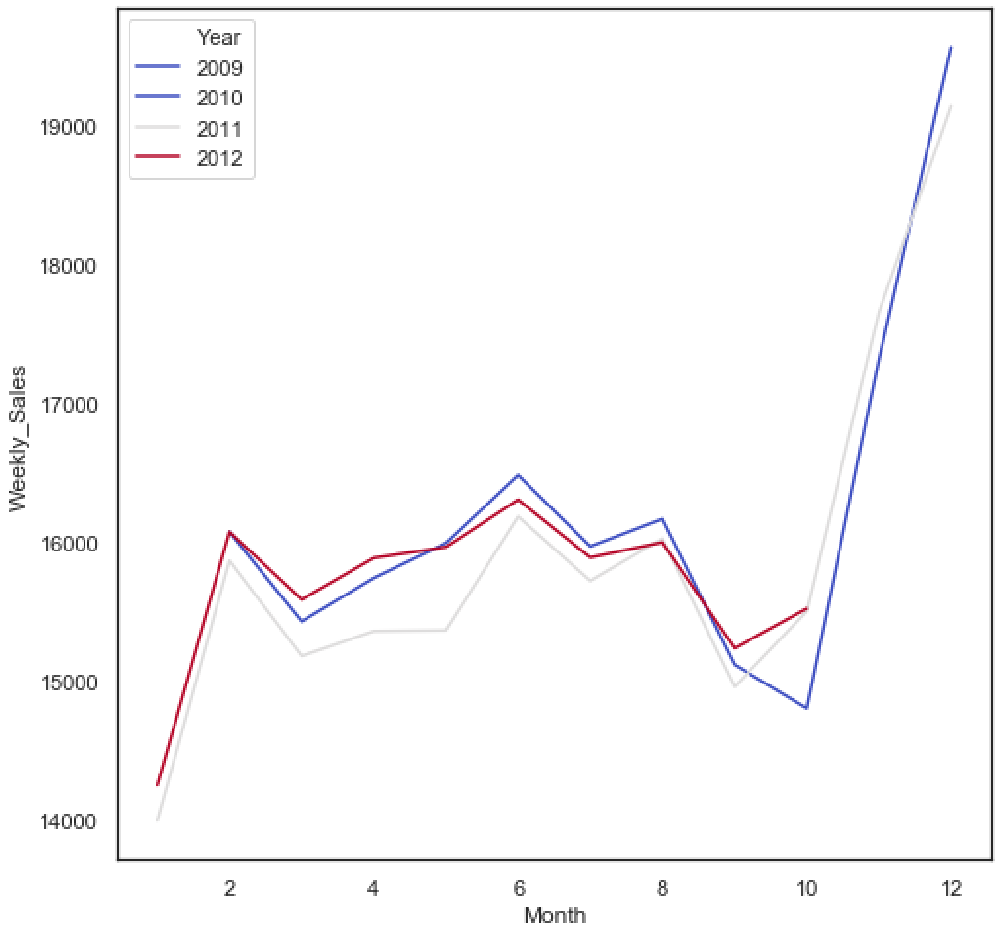
Sales by Year:

The Sales seem to have decreased over the years starting from 2010 to 2012.



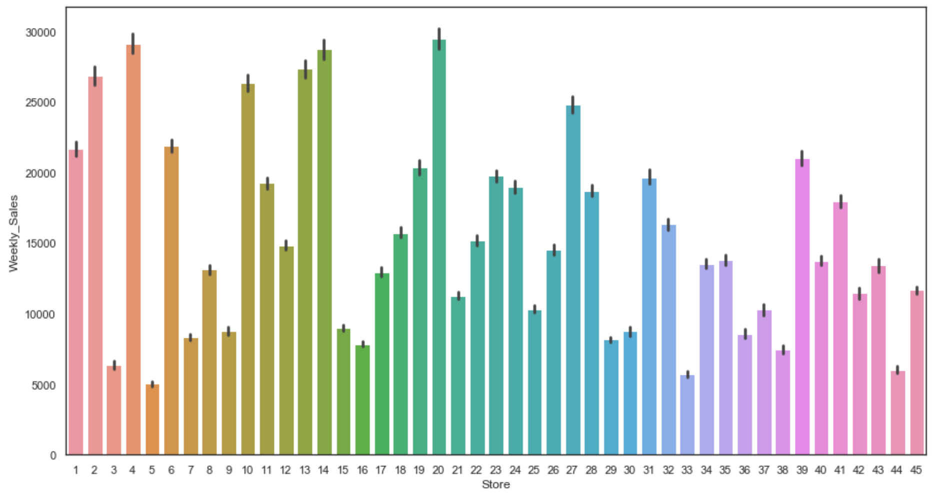
Sale Trends by Month:

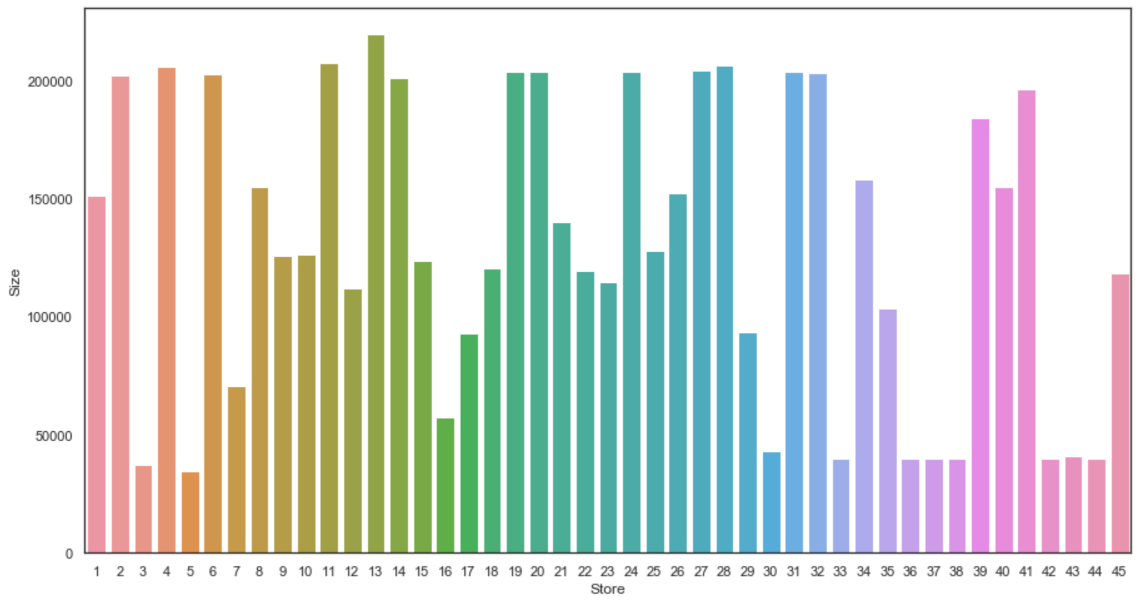
From the figure, the sales spiked in the month of November and December, indicating clear shopping trends during the holiday season which is also consistent over the years.



Store Size Vs Sales:

From the below charts, it was noticed that Stores with larger Size had higher Weekly Sales total.





ALGORITHMS AND TECHNIQUES

Machine Learning techniques being used are K Neighbors Regression & Random Forest Regression.

K Neighbors Regression -

KNN can be used for both classification and regression problems. The algorithm uses '**feature similarity**' to predict values of any new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set, these are steps followed by KNN algorithm

1) First, the distance between the new point and each training point is calculated.
2) The closest k data points are selected (based on the distance).
3) The average of these data points is the final prediction for the new point

Random Forest –

Random forest is like bootstrapping algorithm with Decision tree (CART) model. Say, we have 1000 observation in the complete population with 10 variables. Random forest tries to build multiple CART model with different sample and different initial variables. For instance, it will take a random sample of 100 observation and 5 randomly chosen initial variables to build a CART model. It will repeat the process (say) 10 times and then make a final prediction on each observation. Final prediction is a function of each prediction. This final prediction can simply be the mean of each prediction.

BENCHMARK

To set a benchmark, I ran a simple linear regression model on the dataset with no tuning at all. The obtained RMSE value was 21536.4. My aim was then to develop a model that was more accurate and had a lower RMSE value than this.

Methodology

DATA PREPROCESSING

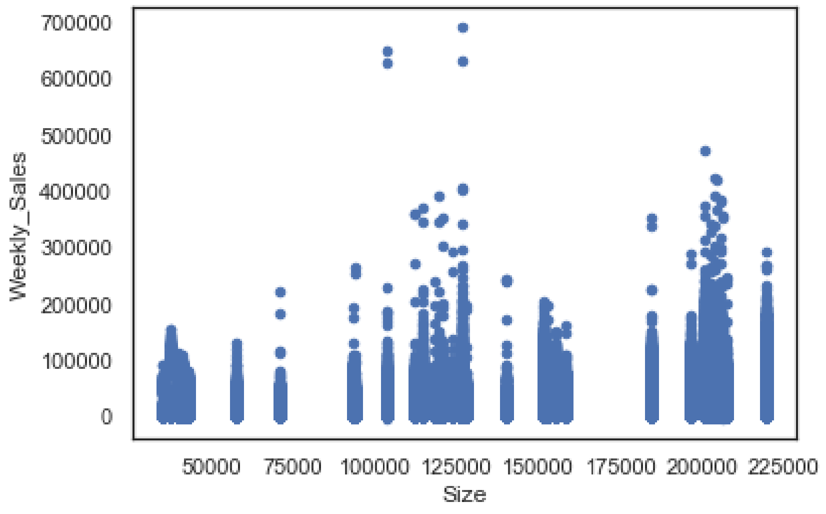
From the initial observations, the data seemed pretty clean with only a value nulls in the markdown columns. These nulls indicate that there were no active promotions during that week of sales. Thus, as far as the missing values were concerned, the data was clean except for the markdowns so a lot of time was then spent on feature engineering.

The first step was to create dummy variables for all the categorical variables. Thus, dummy variables were created for store 'TYPE' and 'ISHOLIDAY'.

Non- active markdowns were marked with a 0 and dummy variables were then created for the five markdown columns.

Outlier Identification:

The next step was to identify if there were any outliers in the sales and the conditions during which the outliers occurred. A scatter plot between Weekly Sales and Stores was then plotted. It was noticed that were indeed a few outliers. The sales looked dense until the $300,00 mark but were thin after that point. So, I decided to filter my data at $300,000 to check if I could identify any trend.



It was noticed that most of these sales occurred in the month of November and December.

Dummy Variables: Date

To address the outliers during the holiday season, two dummy variables named 'BF_surge' and 'Christmas' were created. The sales during the Black Friday days (11/26/2010 and 11/25/2011) and during Christmas (12/23/2010, 12/24/2010, 12/23/2011, 12/24/2011) were marked '1'.

Lagged Sales:

A lagged variable for the previous week's sales was added. A column with the Lagged week's dates was created and was named 'Lag_Week'. These dates were then looked up in the original data file to find their corresponding sales. The sales were brought down as a new column called 'Lag_Sales' of feature into the data set.

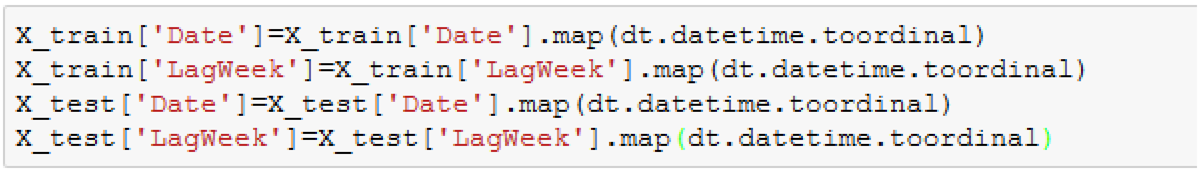
About 8K of the Lagged Week dates did not have corresponding sales and so these null values were replaced by the monthly median sales.

IMPLEMENTATION

After the pre-processing on data was performed and because there were other numerical data values, the data had to be first scaled before using it any model.

My first instinct was to use Robust Scaler to scale the data as there were visible outliers and so it was applied on the features Size, Temperature, Fuel_Price, Unemployment and Lagged_Sales. The data was then split into training and testing data. There were about 337256 records in training and 84314 records in testing data.

The first challenge faced before running the models was dealing with date fields as the models do not support date type variables. After performing a little research, it was found that the date values could be changed to ordinal data.



```
X_train['Date']=X_train['Date'].map(dt.datetime.toordinal)
X_train['LagWeek']=X_train['LagWeek'].map(dt.datetime.toordinal)
X_test['Date']=X_test['Date'].map(dt.datetime.toordinal)
X_test['LagWeek']=X_test['LagWeek'].map(dt.datetime.toordinal)
```

Once the data was ready to be used, it was now the decision to choose the models to run on this data. Having Simple Linear Regression set as the baseline model, I decided to first implement the KNeighbors Regression model. The idea of implementing this model was obtained while reading through some of the discussions on Kaggle. This was an interesting find as I was not very familiar with this model and after some online research found that this model predicts the target by local interpolation of the targets associated with the nearest neighbors in the training set. I decided to let the model use the 'auto' algorithm to fit into the data.

The next algorithm I wanted to implement was Random Forest Regression because of its robust nature and to check if ensembling would help improve the accuracy of the model. After a few trial and errors, I decided to go for 100 estimators. N_estimators upto 50 did not have a significant change in the accuracy and so even though using more estimators meant using more resources I decided to use 100 for a good accuracy. The performance measures before tuning the model further were as below:

Linear Regression [**RMSE: 21528.3, MAE: 14511.6**]

KN Regression [**RMSE: 21070.6, MAE: 12976.5**]

Random forest Regression [**RMSE: 3916.4, MAE: 1548.4**]

REFINEMENT

After running the models initially, I realized that the results were very good and r2 score were very high. That is when I realized that the model was overfitting as I was also inputting Lagged Week Sales data into the model. This was then removed from the input list and the models after which it seemed to perform well.

Even though the results looked good so far, I wanted to try another method of scaling to check if that improved the performance. MinMax Scaler was applied as it limits the numbers between the range of 0 and 1 and have lower standard deviations. This ultimately suppresses the effect of outliers. The performance of Random Forest model improved compared to when using Robust Scaler.

Results

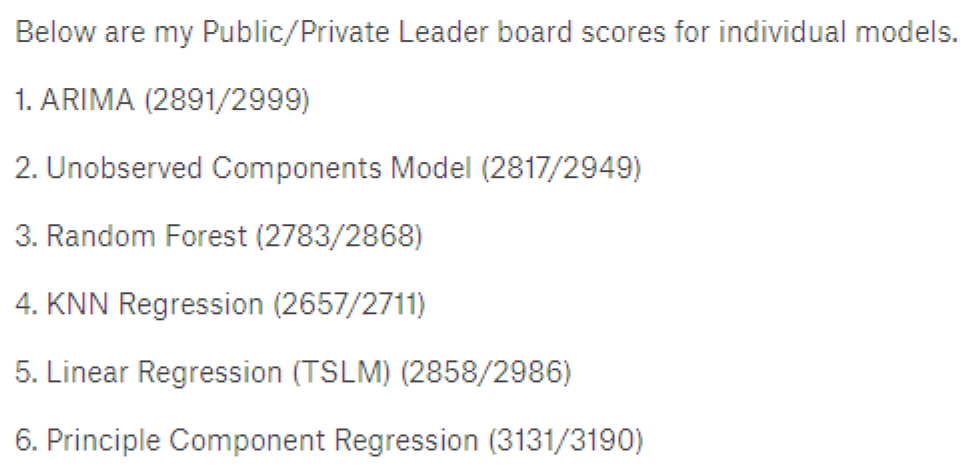
MODEL EVALUATION AND VALIDATION

After a few iterations, the number of estimators and number of nearest neighbors were chosen because they performed the best among all the other tried combinations. KNeighbors Regressor and the better performing RandomForest had the following parameters:

```
knr=KNeighborsRegressor(n_neighbors=5)
knr.fit(X_train,Y_train)
pred_knr=knr.predict(X_test)
```

```
rf=RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train,Y_train)
pred_rf=rf.predict(X_test)
```

To verify if the model was performing was well, I checked with the discussions on Kaggle and found that the leader on the scoreboard had used similar models and was placed #2 in the overall competition. The snapshot of increasing points with each model implemented are as below:


Below are my Public/Private Leader board scores for individual models.

1. ARIMA (2891/2999)
2. Unobserved Components Model (2817/2949)
3. Random Forest (2783/2868)
4. KNN Regression (2657/2711)
5. Linear Regression (TSLM) (2858/2986)
6. Principle Component Regression (3131/3190)

JUSTIFICATION

Linear Regression [**RMSE: 21528.3, MAE: 14511.6**]

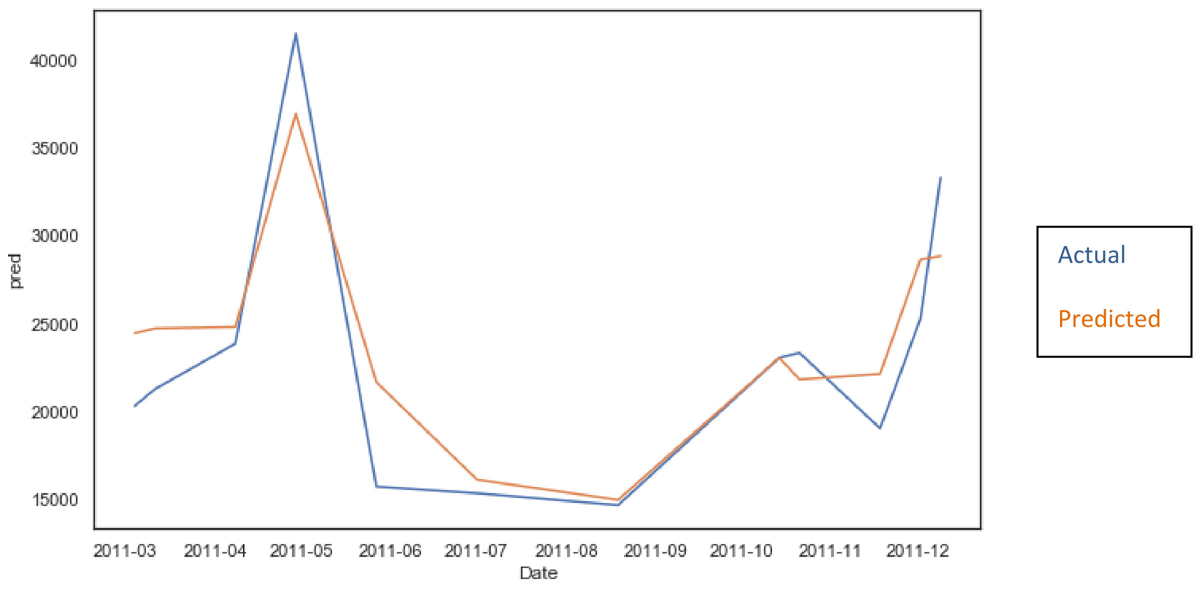
KN Regression [**RMSE: 19241.8, MAE: 12020.0**]

Random forest Regression [**RMSE: 3844.6, MAE: 1518.8**]

The RMSE value of Random Forest is way better than the RMSE value of Linear Regression which was also the benchmark model. The un- tuned Random Forest Regression model had a higher RMSE value than this and so I decided to consider that as my final model. Also, the Weekly Sales range from $15981 to $693099 so RMSE of 3989.3 is comparatively a very lower value.

Conclusion

FREE FORM VISUALIZATION



Above graph is an actual vs Forecast trend for store 1 and department 1 for the year 2011, blue line is actual and orange line is predicted. From the figure, Predicted sales are reasonably close to Actual Sales which indicates that the model is performing pretty well.

REFLECTION

The Steps are summarized below:

1. The initial problem and dataset was found from an online Kaggle Competition
2. The data was downloaded and pre-processed
3. An initial benchmark was set
4. Two models were trained multiple times until the best results were obtained
5. Random Forest was found to perform the best and provided more accurate results to forecast the weekly sales

The most difficult part of this project was to perform feature engineering. Addition of Lagged week dates and sales as additional features to the model was interesting. Also, identifying that the sales spiked during the holiday season and so incorporating the seasonal trends improve the prediction was also very interesting.

IMPROVEMENT

To achieve more accuracy and to be able to forecast the sales more efficiently, more feature engineering can be performed. I could bring down the Root Mean Square Error to 3989.2 but improvement to further reduce would be great.

It would be interesting to note how python equivalent statistical methods like Auto-regressive Integrated Moving Average (ARIMA) which is a class of statistical models that analyze and forecasts time series data forecasts the sales and with what accuracy. One other interesting statistical model that I came across in the discussions was Unobserved Components Model (UCM). It is also a structural time series model, that transform data/ time series into trend, seasonal, Cyclical and other irrelevant components.

REFERENCES:

https://www.analyticsvidhya.com/blog/2014/06/introduction-random-forest-simplified/

https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/

https://www.quora.com/Are-there-instances-where-root-mean-squared-error-might-be-used-rather-than-mean-absolute-error