Rossmann Stores Sales Prediction: Technical Documentation

31.01.2022

—Vivek CP

Data Science Trainee at Alma Better.

Problem Statement

Rossmann is a drug store chain operating over 3000 drug stores in over 7 European countries. We have in hand, the daily sales data of 1115 stores over the period of two and a half years. We are required to develop a machine learning model, which would help the firm predict the sales for next 6 weeks, hence equipping them with knowledge to handle their inventory.

Introduction

The goal of this project as mentioned in the problem statement, is to develop a supervised machine learning regression model which can predict the target variable (Sales) of a drug store chain, with the help of given feature variables.

Regression models are used to predict target variables which are continuous in nature, such as sales.

In addition to regression models, time series analysis is used for such problems as well, which is a more common and practical approach to problems concerning forecasting demand, sales, stock price or any target variable, when we have historical timely data in hand.

Developing a machine learning model involves following steps:

1. Data Description: Understanding the characteristics of the dataset in hand.
2. Handling Null Values
3. Exploratory Data Analysis to understand the underlying patterns in the data.
4. Handling Outliers.
5. Feature Conditioning: Scaling and encoding (categorical features)
6. Feature Selection
7. Model Implementation and prediction

Methodology:

Regression: Regression analysis is used to define the relationship between dependent variable (sales) and set of independent variables. It helps us to understand how the dependent variable changes with variations in an independent variable while keeping all other independent variables fixed. We use it to predict variables that are continuous in nature such as sales in this case.

Following are the regression methodology I've employed to predict the sales:

- **Linear Regression:** Linear regression is a machine learning methodology that learns the relationship between two or more linearly related variables.
- **Linear Regression with Regularization:** Regularization is a technique used to encounter the problem of overfitting. Overfitting occurs when the model complexity increases and the model learns the instances in the dataset too closely, including the noise as well. This may occur when the model parameters are too large.

  Ridge and Lasso regularization are used to compensate or regularize such parameters, and reduce overfitting.

- **Random Forest Regression:**

  Random forest is a bagging decision tree ensemble method. It makes use of multiple decision trees working on different subsets of the dataset (bootstrapping) and aggregates the results of each of the trees by taking the average. Random forest regressor has advantages over linear regression such that it doesn't assume linear relationship between variables or the assumption of multicollinearity, and it tends to yield better accuracy too.

Time Series Analysis:

In addition to regression, I've used time series analysis as well. Time series analysis is a technique for analyzing time series data, such as the one we have in hand, and extract meaningful statistical information and characteristics of the data. The major objective of the analysis is to forecast future values. I've used Facebook Prophet here for the time series analysis.

- **Facebook Prophet:** FB Prophet is a forecasting package in both R and Python that was developed by Facebook's data Science research team. The goal of the package is to give business users a powerful and easy-to-use tool to help forecast business results.

Libraries Used:

- **Numpy**: Numpy is a library for the Python language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
- **Pandas**: pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

- **Seaborn**: Seaborn is a Python data visualization library based on matplolib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- **Scikit Learn**: Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Dataset Prepping:

Features:

We have a dataset containing daily sales entries of over 1115 Rossman stores over a period of two and a half years (2013-01-01 to 2015-07-31), along with supplemental data about the competitional stores as well. Let's look at a few of the columns in the dataset.

- Id - an Id that represents a (Store, Date) duple within the test set
- Store - a unique Id for each store
- Sales - the turnover for any given day (this is what you are predicting)
- Customers - the number of customers on a given day
- Open - an indicator for whether the store was open: 0 = closed, 1 = open
- StateHoliday - indicates a state holiday. Normally all stores, with few exceptions, are closed on state holidays. Note that all schools are closed on public holidays and weekends. a = public holiday, b = Easter holiday, c = Christmas, 0 = None
- SchoolHoliday - indicates if the (Store, Date) was affected by the closure of public schools
- StoreType - differentiates between 4 different store models: a, b, c, d
- Assortment - describes an assortment level: a = basic, b = extra, c = extended
- CompetitionDistance - distance in meters to the nearest competitor store
- CompetitionOpenSince[Month/Year] - gives the approximate year and month of the time the nearest competitor was opened
- Promo - indicates whether a store is running a promo on that day
- Promo2 - Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating
- Promo2Since[Year/Week] - describes the year and calendar week when the store started participating in Promo2
- PromoInterval - describes the consecutive intervals Promo2 is started, naming the months the promotion is started anew. E.g. "Feb,May,Aug,Nov" means each round starts in February, May, August, November of any given year for that store

The sales is our target variable here that we need to predict.

Dataset Shape:

Rossman Store df: (1017209,9)

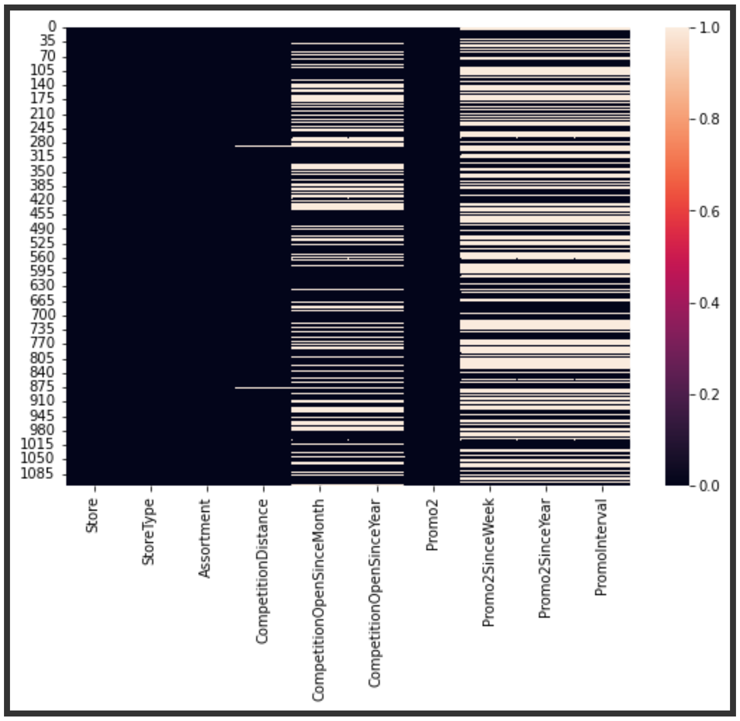
Store_data_shape: (1115,10)

Handling NaN values:

The distribution of Null Values in the dataset is as shown.

We've got null values in following features:

1. CompetitionDistance: Null values replaced by median
2. CompetitionOpenSinceMonth: Null values replaced by mode
3. CompetitionOpenSinceYear: Null values replaced by mode
4. PromoInterval, Promo2SinceYear, Promo2SinceWeek: Null values replaced by 0 or 'None' according to datatype.

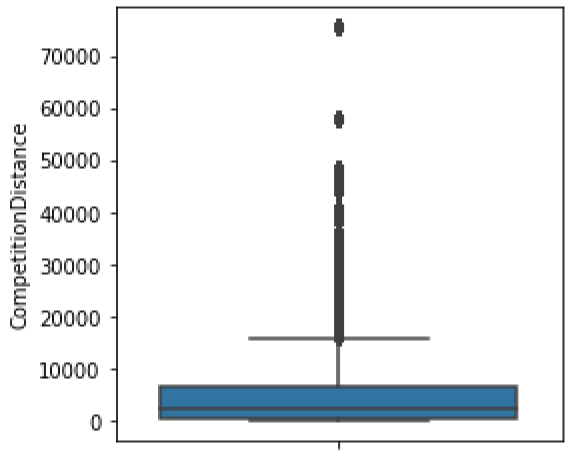


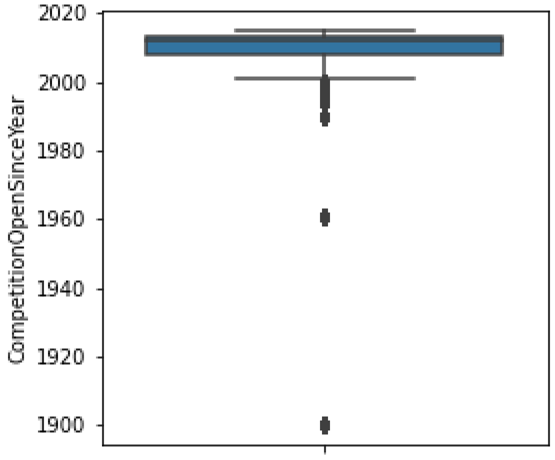
Upon null value clean up, the two datasets were merged on store id to obtain a (1017209,18) shaped dataset.

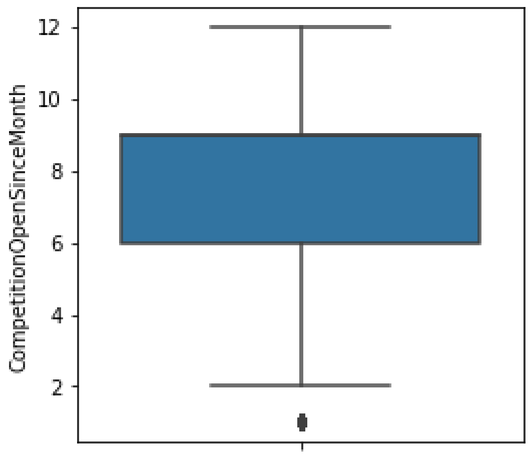
Handling Outliers:

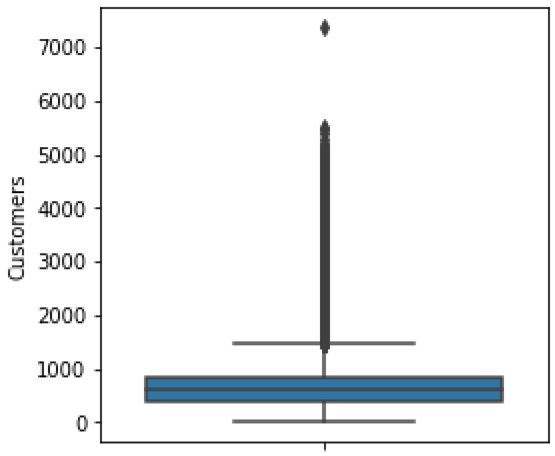
Outliers are data points that highly deviate from the rest of the datasets. These can heavily influence the linear regression model.

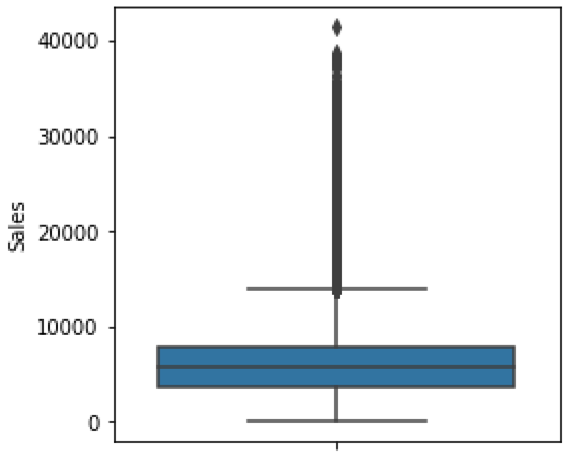
I used boxplot to detect outliers in the numerical features and here are the fields the outliers were detected in:


CompetitionDistance
70000
60000
50000
40000
30000
20000
10000
0


CompetitionOpenSinceYear
2020
2000
1980
1960
1940
1920
1900


CompetitionOpenSinceMonth
12
10
8
6
4
2


Customers
7000
6000
5000
4000
3000
2000
1000
0


Sales
40000
30000
20000
10000
0

CompetionDistance, CompetitionOpenSinceYear, CompetitionOpenSinceMonth, Customers and Sales are found to contain outliers. But if we observe the outliers in Sales and Customer columns, we can see they are quite large in numbers and uniformly distributed. That is to say, there's some pattern to it that we don't want to strip off our model from. So we'll leave our Customers and Sales column out of the outlier removal process.

The outliers in the rest of the columns were removed using the IQR method.

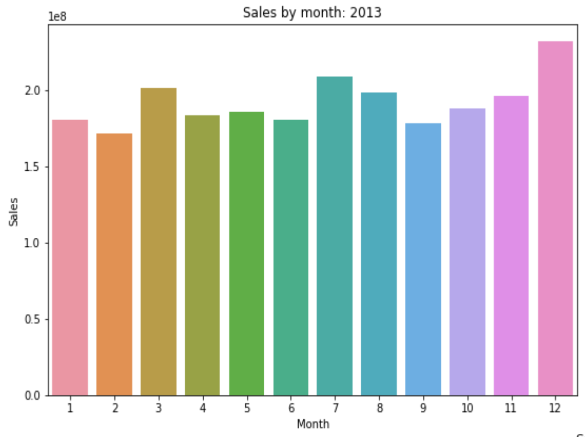
Upper limit: 75th Percentile+1.5*IQR

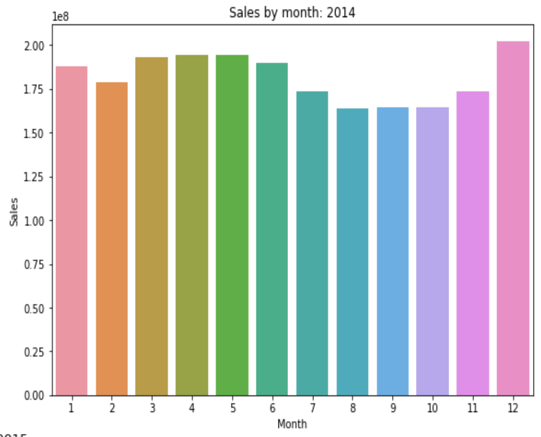
Lower Limit: 25th Percentile- 1.5*IQR

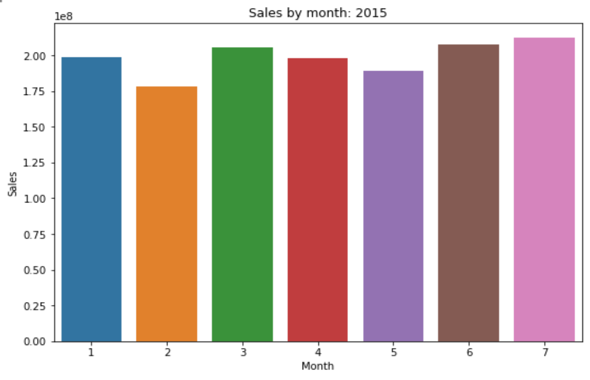
Exploratory Data Analysis:

Once the dataset is cleaned up and merged we look for intrinsic characteristics and patterns in the data. Here are few of the major findings:

1. **Sales by month**: Here we try to understand the variation in sales across the month. Since for the year 2015, we don't have the data beyond the month of July, taking cumulative average would yield misleading conclusions. So we look at sale by months of each year:

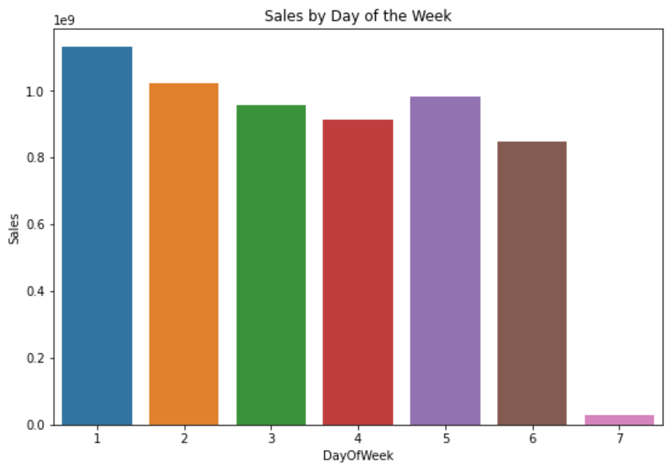






Remarks: We can observe that the month of december has recorded highest sales in the year 2013, and 2014. We can safely assume that the sales generally tend to grow towards the year end.

2. Sales by day of week:



Remarks:

There's an interesting pattern to be observed here: The sales gradually decrease from Monday to Thursday, and are met with a rise on Fridays. This implies the presence of some correlation between the sales and which day of the week it is.

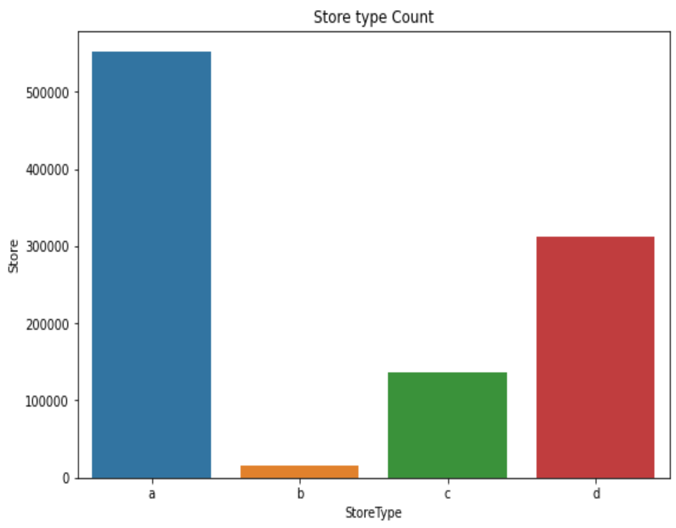
3. Store and Customer visits:

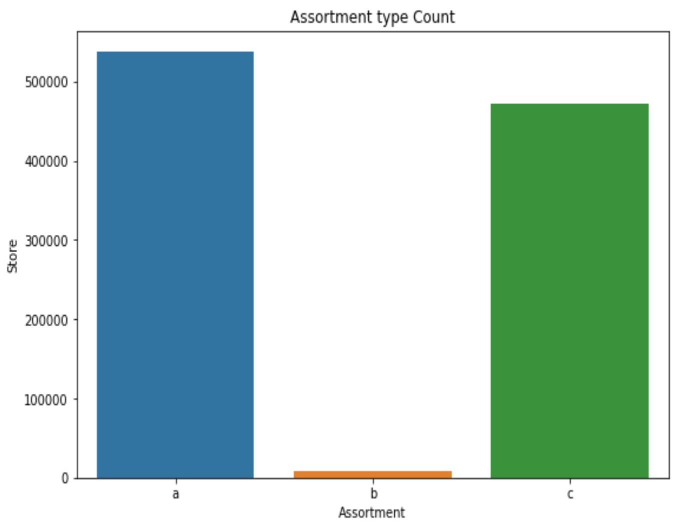
These are top 5 stores with most customers:

Store ID	Customers
733	3206058
262	3204694
562	2924960
769	2902351
1114	2509542

Remarks Store number 733 has the most number of customer visits, and is closely followed by store 262. Store number 543 has had least customer traffic. Studying characteristics of these stores might help us to understand the drivers of sales.

4. Store Type and Assortment Types

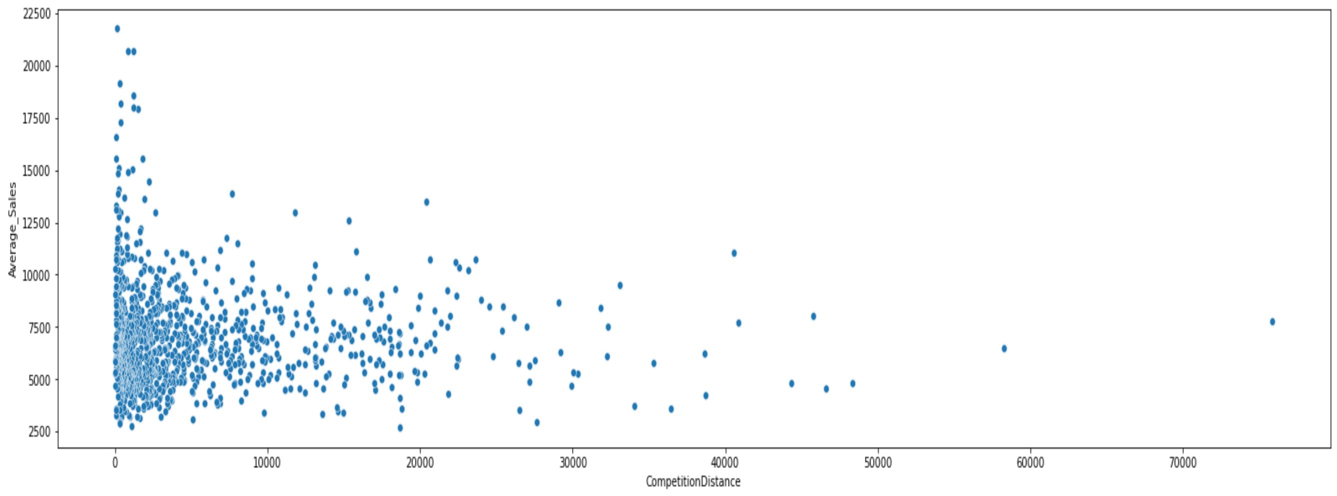




Here’s the count of assortment type and store type in the dataset. As we see, store type b and assortment b are significantly low in number as compared to the rest of the types. This may constitute an imbalance in the dataset when we try to fit our models.

5. Competition Distance:

Here we analyze how the average sales of rossmann stores varies with variation in distance of nearest competitional store.



Remarks

I originally expected the Rossmann store sales to go up as they become more remote. That is, since they don't have much competition in the near vicinity, the customers wouldn't be left with much choice, and just go to Rossmann. But that's not the case.

As we see in the scatter plot, most of the higher range of average sales are concentrated towards the lower competitional distances (0-10000 m).

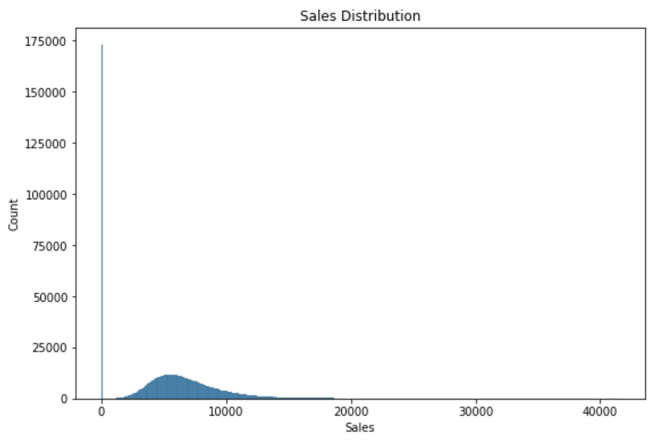
My reasoning would be that the stores that have competitions around their close vicinity are located in dense urban areas, where the population and spending power both are high. This would be able to drive the sales up for these stores. The stores that are in remote areas on other hand would not be having much of customer traffic to help with the sales.

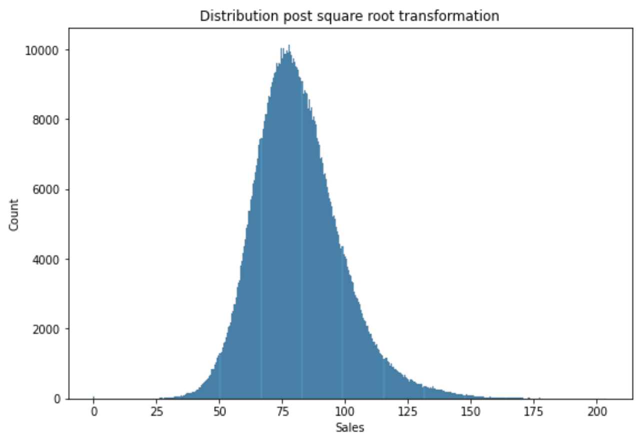
Feature Engineering:

Target feature conditioning:

Linear regression model assumes that the features follow a gaussian distribution. Here's what our sales column distribution looks like as it is:

We can see that it does approximate a gaussian distribution, but is slightly skewed to the right. We can fix this by carrying out a square root transformation.



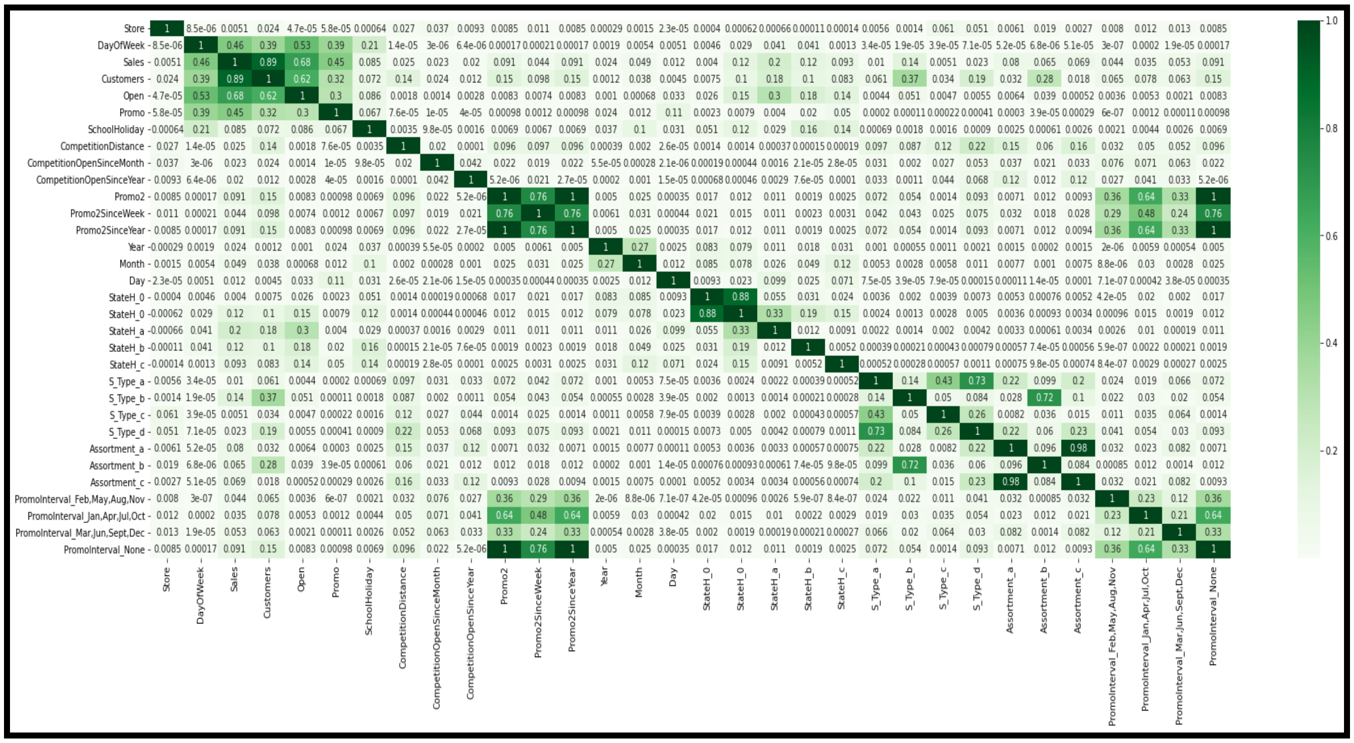


Categorical feature encoding: For our model to recognize the categorical features in the dataset, it's important that we encode them with numerical values. I've done encoding here using one hot encoding method. This method creates a new column for every value in the categorical column, and marks it with value '1' at instances it's active at and with '0' where it's not. Following are the columns that underwent one hot encoding: `'StateHoliday','StoreType','Assortment','PromoInterval'`

. After encoding these we have 32 columns in our dataset.

Feature Selection:

For our linear regression model to perform well, it's important that we carefully choose the independent features. These features should be decently correlated to the target feature and also be free of multicollinearity. Multicollinearity arises when the independent features are correlated to each other. Here's a heatmap depicting the correlation between features in the dataset:



The dark spots in the map represent instances of high correlation.

To start off our feature selection process, from the heatmap, we can observe that there are features whose correlation to our target variable sales is very low. We can take these off right away.

Followed by this, we can look at other features that are showing high correlation to each other. We do not need to include both of these to the model, so we select the one which is more correlated to the target feature.

With this manual method I was able to remove following features:

```
'Store','Assortment_b','S_Type_b','S_type_c','StateH_0','S_Type_c','Promo2
','CompetitionDistance','CompetitionOpenSinceMonth','CompetitionOpenSinceY
ear','Promo2SinceWeek','Day','S_Type_d','PromoInterval_Jan,Apr,Jul,Oct','S
ales','Promo2SinceYear','Promointerval_None','PromoInterval_Feb,May,Aug,No
v','PromoInterval_Jan,Apr,Jul,Oct','PromoInterval_Mar,Jun,Sept,Dec','S_Typ
e_a','Date', ‘assortment c’.
```

As for the rest of the features, I used **VIF method,** to further filter out multicollinearity:

Variable	**VIF**
DayOfWeek	8.553595
Customers	4.969602
Open	13.777009
Promo	1.995819
SchoolHoliday	1.346284
Year	32.935062
Month	4.212767
StateH_a	1.225258
StateH_b	1.118830
StateH_c	1.075896
Assortment_a	2.123727
PromoInterval_None	2.072111

Here we can see that the feature ‘Year’ has high VIF value, indicating multicollinearity.

Here’s how the VIF values look upon removing ‘Year’ feature:

Variables	**VIF**
DayOfWeek	3.210494
Customers	4.969416
Open	7.482442
Promo	1.902145
SchoolHoliday	1.336638
Month	3.818019
StateH_a	1.070407
StateH_b	1.061704
StateH_c	1.060964
Assortment_a	2.046793
PromoInterval_None	2.020194

As we can see our VIF value has now been contained within a reasonable threshold of 10.

Here are the 11 final independent features that we'll use to train our model:

`'DayOfWeek','Customers','Open','Promo','SchoolHoliday','Month','StateH_a','StateH_b','StateH_c','Assortment_a','PromoInterval_None'.`

Since these features have different scales and units associated with them, we'll scale them all to the same comparable scale using **StandardScaler** available in the Sklearn library.

Once we are through with our feature selection, we can move ahead with model implementation.

Model Implementation:

We split the dataset into test and train with a test size of 30%.

X train shape: (712046, 10)

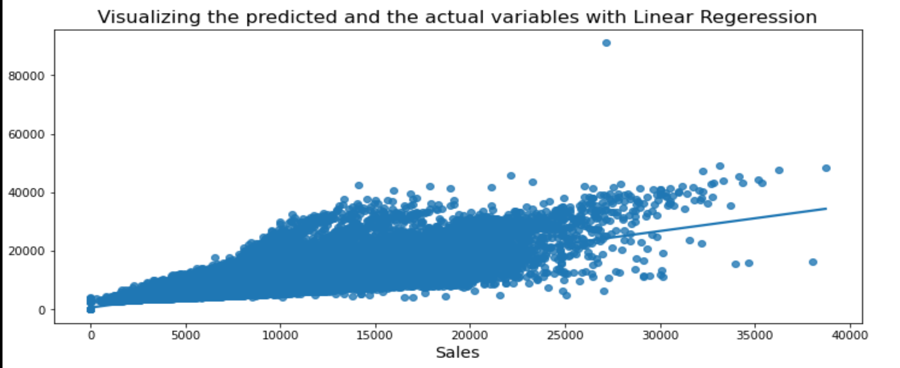
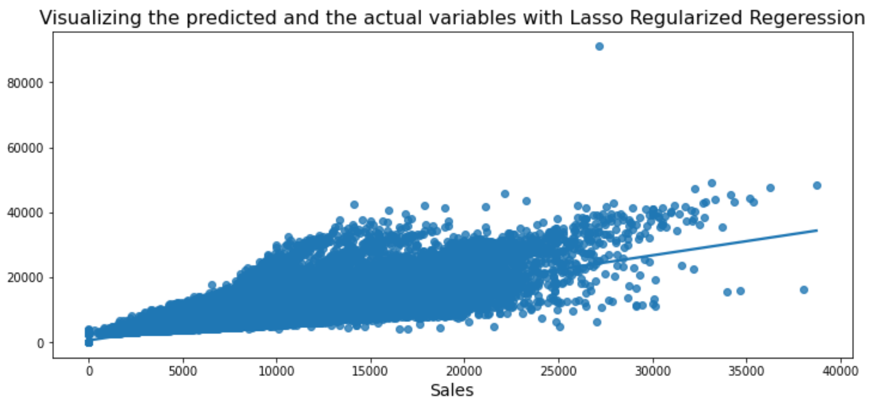
Y train shape: (712046,1)

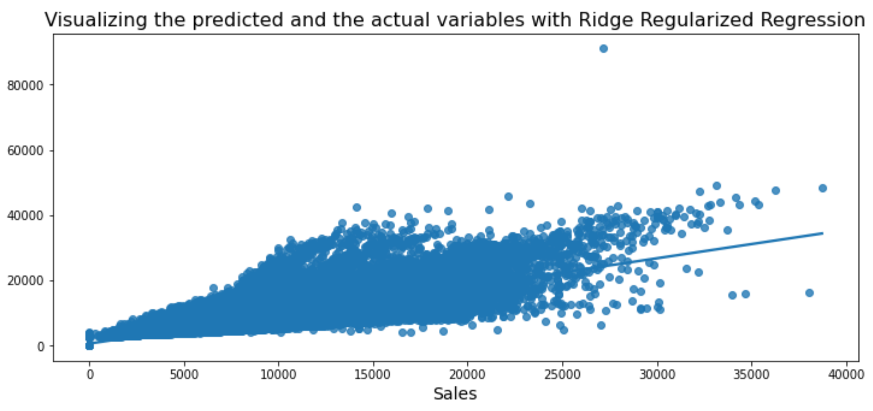
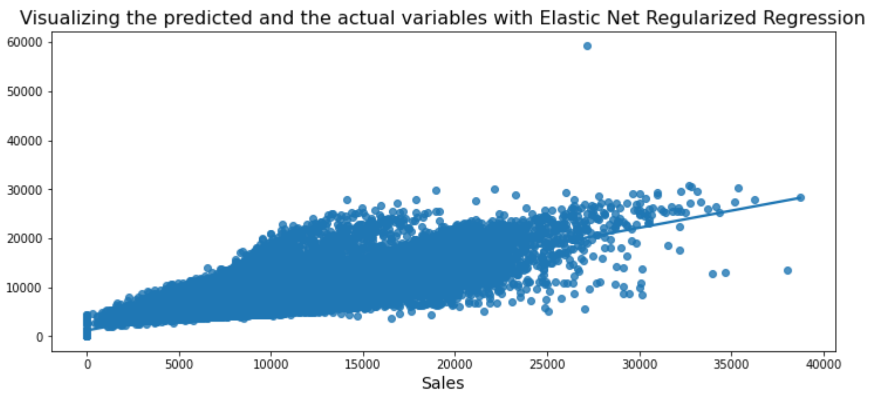
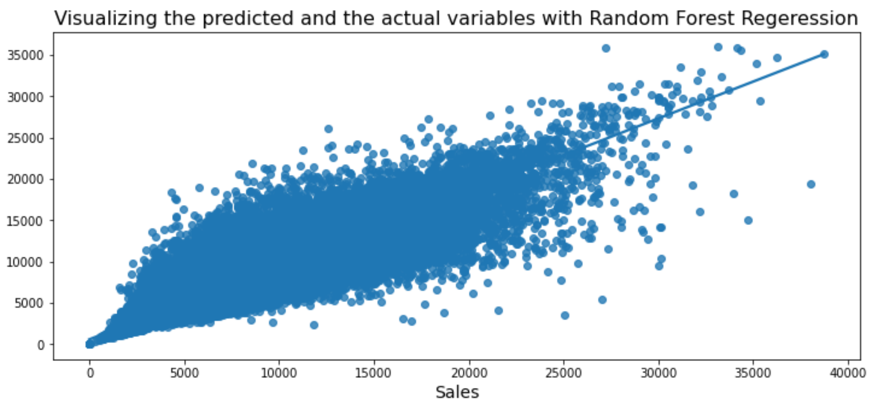
X test shape: (305163, 10)

Y test shape: (305163,1)

1.Regression:

Following table summarizes the regression model results and visualizations:

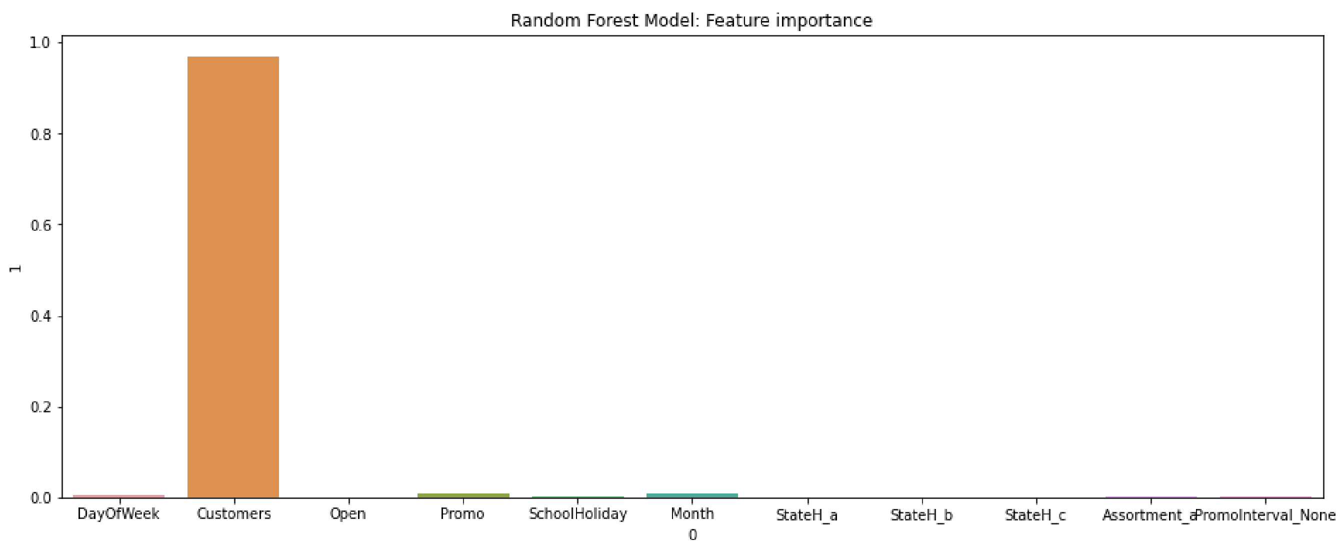
Linear Regression: Regression Score: 93.86% MSE : 2650046.85 RMSE : 1627.89 R2 : 0.8207 Adjusted R2 : 0.8207	
Lasso Regularized Regression Regression Score: 93.86 MSE : 2650046.44 RMSE : 1627.89 R2 : 0.8207 Adjusted R2 : 0.8207	

Ridge Regularized Regression Regression Score: 93.86% MSE : 2650037.11 RMSE : 1627.89 R2 : 0.8207 Adjusted R2 : 0.8207	
Elastic Net Regularized Regression Regression Score: 87.68% MSE : 3168146.48 RMSE : 1779.92 R2 : 0.7856 Adjusted R2 : 0.7856	
Random Forest Regression: Regression Score: 98.31% MSE : 1997817.68 RMSE : 1413.44 R2 : 0.8648 Adjusted R2 : 0.8648	

Linear Regression Model Coefficients:

Feature	Coefficient
'DayOfWeek'	-0.99871
'Customers'	19.857750
'Open'	13.966787
'Promo'	4.188622
'SchoolHoliday'	0.130462
'Month'	0.502287
'StateH_a'	-0.706267
'StateH_b'	-0.535271
'StateH_c'	-0.184255
'Assortment_a'	-1.249139
'PromoInterval_None'	-0.271236

Random Forest Model Feature Importance:



Remarks:

We've formulated a number of regression models, of which Random Forest regression has performed the best at predicting our target variable.

Looking at the model coefficients of linear regression can tell us why lasso and ridge failed to bring about any improvement. The model most likely isn't affected by the problem of overfitting to begin with.

Disadvantages:

The issue with the models prepared so far, is painfully obvious when we take a look at linear regression model coefficients and random forest feature importances.

All these models almost completely relies on the number of customers to predict sales. We do not really need a machine learning model to predict sales if we have in hand the footfall for the day.

Solutions:

This problem can be tackled through two methods:

1. Create a regression model to predict the number of customers on a given day, using the rest of the features. We then use this predicted footfall value to predict sales. The model performance is bound to go down here, since we are incorporating the error in the customer predictions as well. But this is nonetheless the more practical approach.

2. Create a time series model which forecast sales by studying the seasonal trend in the data.

2.Sales Prediction Using Predicted Customer Values:

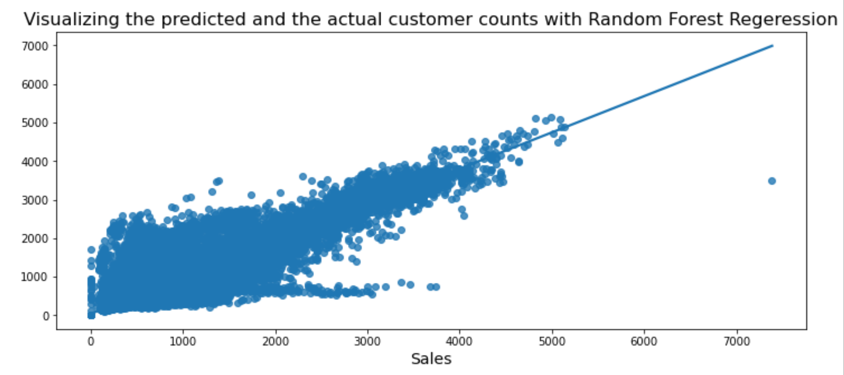
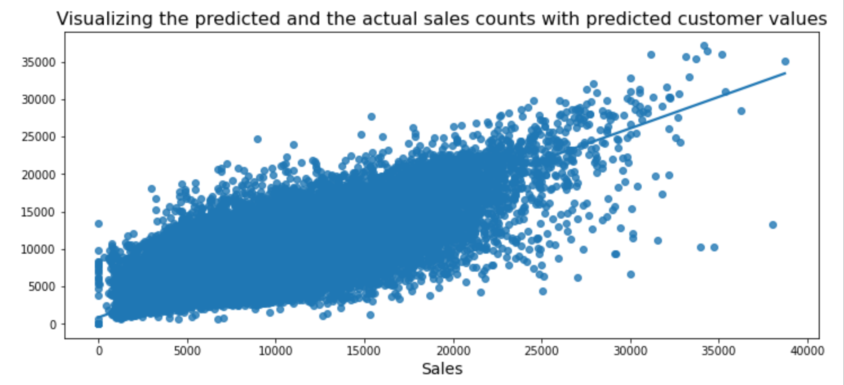
I used random forest regression to predict customers with following features:

```
'Store','DayOfWeek','Open','Promo','SchoolHoliday','CompetitionDistance','
CompetitionOpenSinceMonth','CompetitionOpenSinceYear','Promo2','Promo2Sinc
eWeek','Promo2SinceYear','Month','Day','StateH_0','StateH_0','StateH_a','S
tateH_b','StateH_c','S_Type_a','S_Type_b','S_Type_c','S_Type_d','Assortmen
t_a','Assortment_b','Assortment_c','PromoInterval_Feb,May,Aug,Nov','PromoI
```

```
nterval_Jan,Apr,Jul,Oct','PromoInterval_Mar,Jun,Sept,Dec','PromoInterval_N
one'
```

All of these features are something that we have knowledge about in advance. So the process would be that the model would be required to be inputted with above variables for the days to come, which would return the forecast of footfall for each instances, and this value would be used to forecast sales.

Here are the performance metrics of the new model:

Customer prediction using Random Forest: Regression Score: 99.78% MSE : 23081.65 RMSE : 151.92 R2 : 0.8931 Adjusted R2 : 0.8931	
Sales prediction using Random Forest with predicted customer values: Regression Score: 99.02% MSE : 2868683.40 RMSE : 1693.71 R2 : 0.8059 Adjusted R2 : 0.8059	

3.Time Series Analysis

Time series analysis is a specific way of analyzing a sequence of data points collected over an interval of time. In time series analysis, analysts record data points at consistent intervals over a set period of time rather than just recording the data points intermittently or randomly. One of the major

Since what we have in hand fulfills that criteria, we can use time series analysis to forecast the sales. I'll use **Facebook Prophet** to do the analysis.

Facebook Prophet:

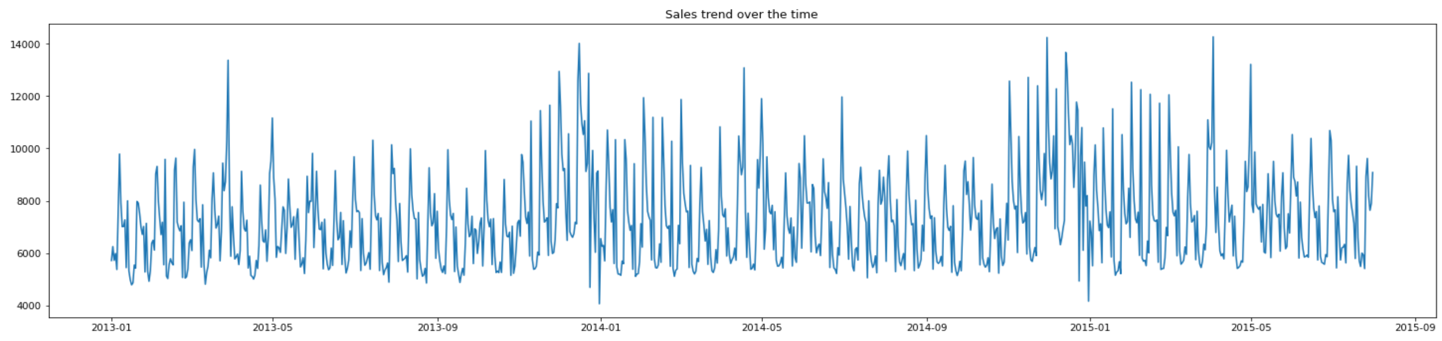
Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data.

Data Prepping:

Facebook Prophet in its most basic form makes use of just timestamps and target variables. It requires the timestamp column to be named as 'ds' and the target column to be named as 'y'.

I'll be using average sales per day as the target variable, and omit the days when the stores were closed.

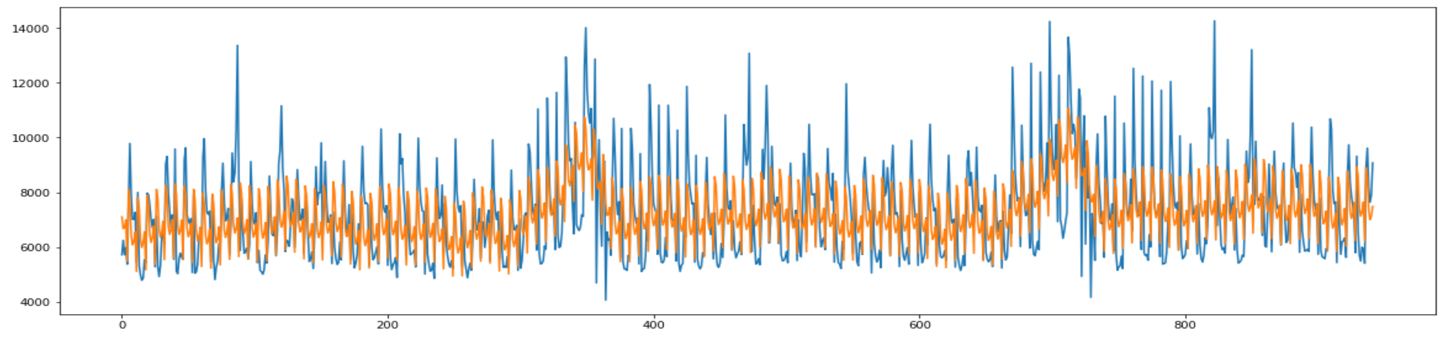
Following figure shows us the sales trend over the years:



Forecasting:

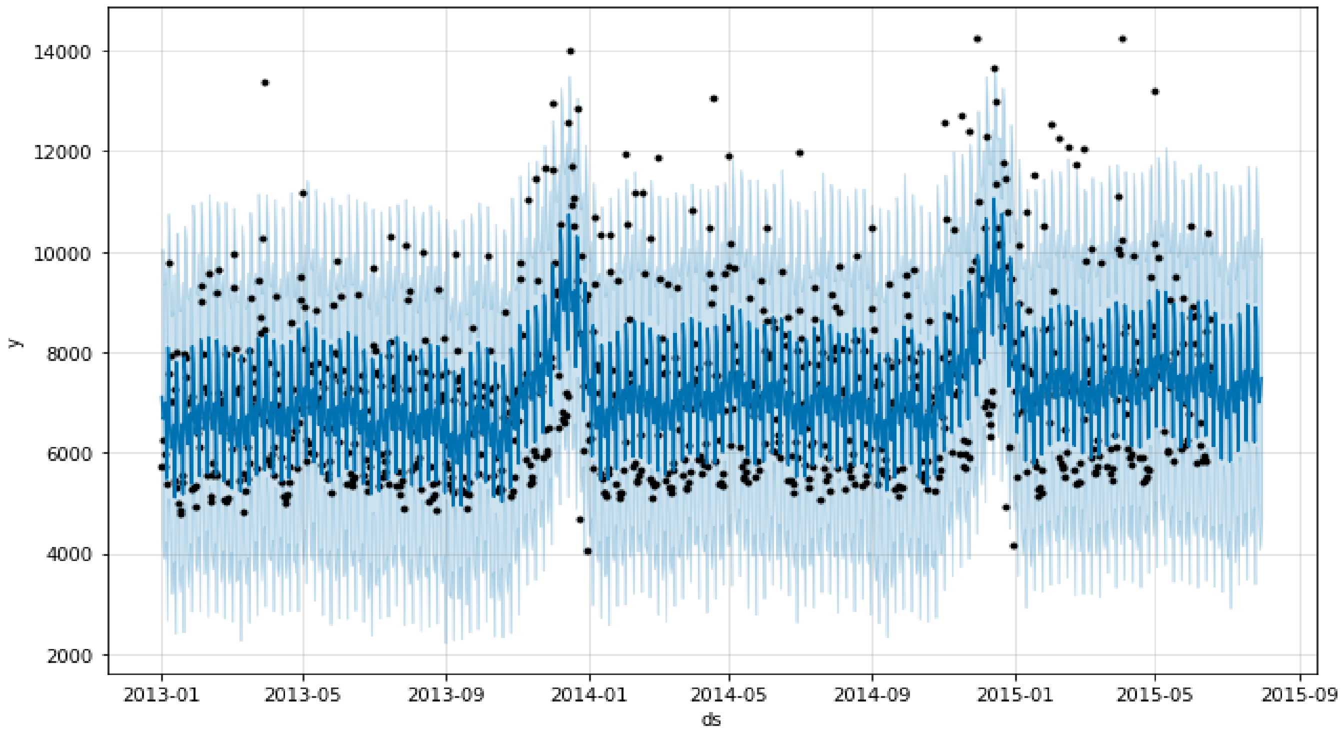
We create a future dataframe with a period of 42 days to be used for forecasting. Here are our actual and predicted values plotted together :

The orange line represents the predicted values.



Although the model has been able to imitate the trend in the data, it hasn't done a good job at capturing the peaks in the sales. We should incorporate some confidence interval into the forecast in order for us to better capture these instances:

Here's how are model looks with 95% confidence interval:



The black spots represent the actual values, the dark blue line represents predictions and the light blue region represents confidence interval.

So as we see, we have been able to capture most of the points with taking into consideration the confidence interval.

Model Diagnosis:

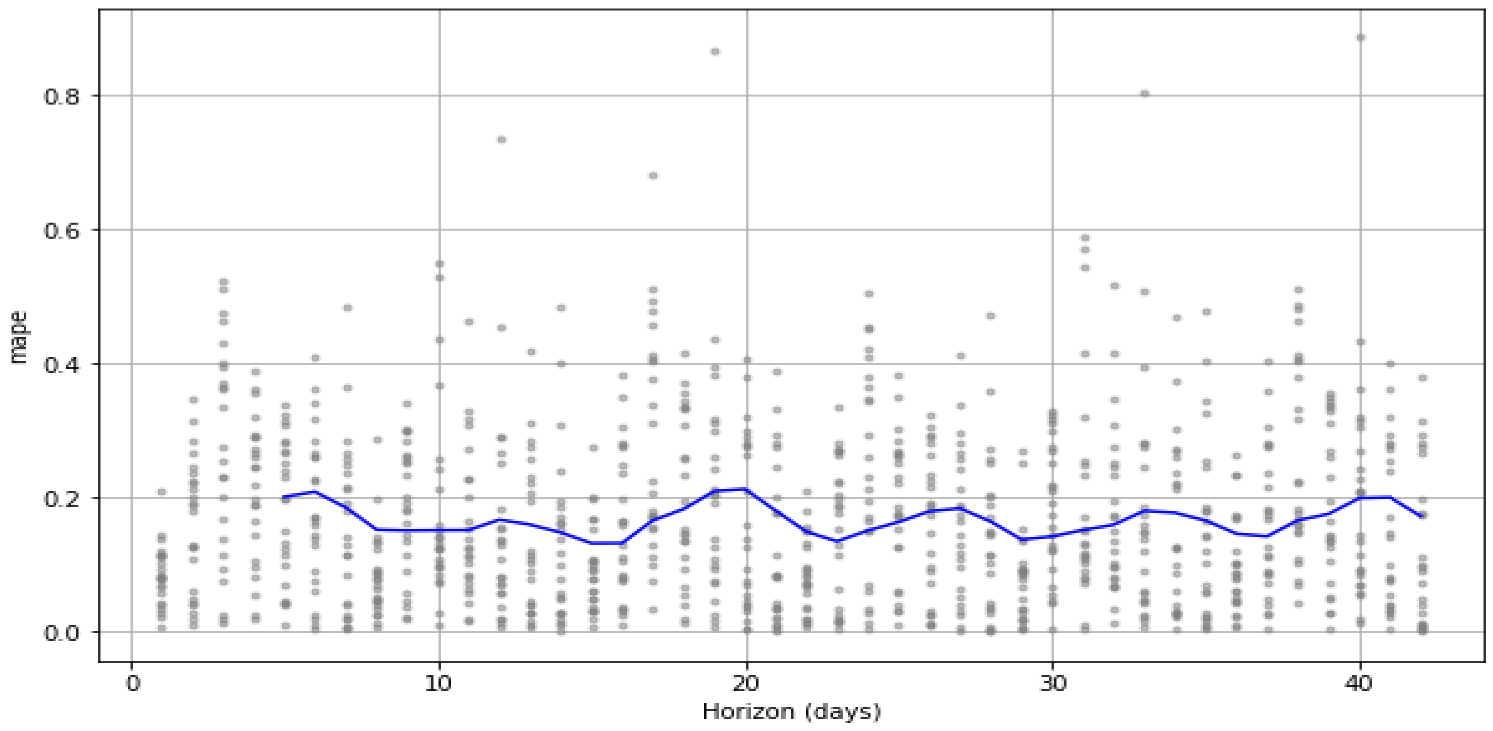
Following is the model diagnosis over a forecast horizon of 42 days:

horizon	mse	rmse	mae	mape	mdape	coverage
5 days	3.305448e+06	1818.089167	1525.276989	0.200589	0.198907	0.880000
6 days	3.146887e+06	1773.946716	1496.153488	0.208230	0.207968	0.901667

horizon	mse	rmse	mae	mape	mdape	coverage
7 days	2.763578e+06	1662.401263	1316.886535	0.184775	0.191111	0.941667
8 days	2.162627e+06	1470.587238	1070.321673	0.151516	0.131131	0.958333
9 days	2.704998e+06	1644.687760	1179.803732	0.150309	0.133051	0.918333
10 days	2.700173e+06	1643.220412	1210.181368	0.150747	0.129637	0.916667
11 days	2.401443e+06	1549.659119	1199.711464	0.150925	0.124833	0.908333
12 days	2.594802e+06	1610.838914	1292.370776	0.166338	0.134940	0.890000
13 days	2.143142e+06	1463.947278	1174.306656	0.159153	0.123500	0.921667
14 days	2.363538e+06	1537.380194	1116.534996	0.147279	0.113773	0.926667
15 days	2.000387e+06	1414.350358	969.471411	0.131164	0.098860	0.946667
16 days	2.400054e+06	1549.210612	1073.900712	0.131444	0.103231	0.916667
17 days	3.012283e+06	1735.592896	1294.884622	0.165568	0.133098	0.888333
18 days	2.662488e+06	1631.713279	1314.875175	0.182242	0.152867	0.906667
19 days	2.902022e+06	1703.532096	1433.578369	0.209129	0.170921	0.890000
20 days	2.527123e+06	1589.692623	1345.455356	0.212302	0.170921	0.921667
21 days	1.787871e+06	1337.113121	1090.853916	0.179858	0.144512	0.953333
22 days	1.292215e+06	1136.756242	877.370232	0.148295	0.096869	0.960000
23 days	1.650145e+06	1284.579517	943.753460	0.134167	0.092726	0.926667
24 days	2.330513e+06	1526.601787	1142.308310	0.150366	0.126251	0.918333
25 days	2.546547e+06	1595.790299	1250.678272	0.162860	0.153718	0.918333
26 days	2.860183e+06	1691.207512	1382.923933	0.178998	0.179391	0.900000

horizon	mse	rmse	mae	mape	mdape	coverage
27 days	2.709656e+06	1646.103315	1347.135247	0.183471	0.179391	0.921667
28 days	2.431545e+06	1559.341071	1193.278050	0.163604	0.171565	0.926667
29 days	1.950427e+06	1396.576708	983.225882	0.136756	0.114914	0.938333
30 days	2.522892e+06	1588.361390	1118.513302	0.141364	0.120403	0.916667
31 days	2.685805e+06	1638.842459	1197.231913	0.150979	0.118484	0.896667
32 days	2.506345e+06	1583.143977	1228.491469	0.158805	0.124701	0.896667
33 days	2.801231e+06	1673.687960	1352.682288	0.179689	0.145369	0.890000
34 days	2.436575e+06	1560.953227	1263.706947	0.176759	0.127256	0.913333
35 days	2.622004e+06	1619.260281	1210.394226	0.165117	0.131181	0.933333
36 days	2.206863e+06	1485.551429	1045.948396	0.145374	0.110200	0.956667
37 days	2.522993e+06	1588.393288	1128.161670	0.141548	0.112305	0.928333
38 days	2.973815e+06	1724.475145	1303.128273	0.165728	0.132730	0.898333
39 days	2.536426e+06	1592.616231	1286.245749	0.174958	0.148378	0.916667
40 days	2.734060e+06	1653.499405	1390.414665	0.198679	0.166052	0.910000
41 days	2.336472e+06	1528.552221	1293.001295	0.199777	0.162556	0.933333
42 days	1.682022e+06	1296.927980	1055.255086	0.171645	0.142285	0.963333

Here's the plot of maps values over the forecast horizon:



Remarks:

1. The model has been able to maintain mape value below 20% through the horizon. Which is an acceptable performance, considering that the model uses only sales data over the year to forecast.
2. We can improve the performance of the model by doing multivariate analysis and include other features into the model.
3. The Prophet has the provision to take holiday data into consideration also. This would help with increasing the performance too.
4. I'll leave these two for the future scope of the project.

Conclusion:

The motive of the project was to predict the sales for a pharmaceutical drug store chain by the name of Rossmann. We were provided with a couple of comprehensive datasets about Rossmann and competitional stores as well.

1. Regression:

With the data filtered for multicollinearity and features trimmed down, we ran a number of regression models on the features, of which Random Forest performed the best with a regression score of 98.31 and adjusted R2 of 86.48% . But these models had an obvious flaw to them, that is they depended on the feature 'Customers' in order to predict sales. The number of customers in a real world scenario wouldn't be something we'd have in hand. And removing this one feature brought down the model performance by a huge factor.

2. Regression with predicted customer values:

So the first solution to this problem was to create a model which predicts the number of customers using the rest of the features in hand. I used the random forest regressor again for this task. The resulting model scored 99.78% and adjusted R2 of 89.31%. Now we can use these predicted values of footfall to go ahead and predict sales.

I again used the random forest regressor to predict sales using predicted customer value. The model scored a regression score of 99.02% and adjusted R2 of 80.59%.

Yes the model performance has come down by a factor 5.89 %, but it remains a much more practical approach.

3. Time Series Analysis:

Finally I used Facebook Prophet to create a time series model, which is capable of learning the seasonality trend in the dataset to forecast sales. The model performed reasonably well considering the fact that it uses only sales data over the year to do the forecast. It was able to maintain MAPE value below 20 % through a forecast horizon of 42 days. The model has the potential to improve if we use multivariate analysis and incorporate holiday data into the model. But I'll keep that for the future scope of the project.

In conclusion, of all the designed models, I recognize the second model to be the most practical and efficient to forecast sales.