CS483 BIG DATA MINING – ROSSMANN SALES FORECASTING REPORT

Project By:

1. Revanth Varanasi- 672049065
2. Vamsi Krishna Mandalapu- 663364228
3. Sai Samhita Chandu- 658474160

Project GitHub Repository link:
https://github.com/vsrrevanth/CS483RossmanSalesForecasting

Introduction:

In today's fast-paced retail industry, accurate sales forecasting is critical for businesses to plan and strategize their operations effectively. Accurate sales forecasting can help retailers make informed decisions, plan for future growth, and efficiently allocate resources. However, the traditional manual methods of forecasting are time-consuming and require a significant the number of human resources. Therefore, there is a need to develop more advanced and efficient techniques for sales forecasting.

This project aims to address this issue by developing a machine learning model that can accurately predict daily sales for up to five months in advance. By leveraging machine learning algorithms, the model will be able to identify patterns and trends in sales data that might otherwise go unnoticed, such as seasonality and the impact of economic factors. By doing so, the model will provide more accurate and reliable sales forecasts, allowing businesses to plan for the future with confidence.

Goal of the project:

The main goal of this project is to build an accurate sales forecasting model that can help retailers plan their business expansion and allocate resources efficiently. The model will consider various factors such as sales trends, seasonality, and other economic factors to provide accurate predictions.

Problem Statement and Objectives:

The scope of this project includes collecting and analyzing historical sales data, identifying relevant variables, and building a machine learning model using advanced algorithms. The model will be trained on the historical data and validated using different performance metrics. The model will then be used to predict future sales for each store up to five months in advance. The project will also include developing a user-friendly interface to allow sales teams to access the predictions easily. The project will not include any physical implementation or integration with other systems.

Dataset Overview:

Our data consists of historical sales data for 1,115 Rossman stores across Europe of 2.5 years from 2013/1/1 to 2015/7/31. We are dealing with two datasets - Store and Sales. Sales dataset has 10,17,209 records and '*Sales*' is the target column which indicates the historical sales across different stores on each day and we are tasked with predicting the daily sales for up to 5 months in advance.

- Number of records: 10,17,209
- Number of columns: 9
- Target feature: Sales
- Target Feature type: Numerical (Continuous)

Data Source: Collected Rossmann store sales data from https://www.kaggle.com/competitions/rossmann-store-sales

Sales Dataset:

Column	Data Type	Type of Feature	Description
Store	int64	Numerical	Store ID
DayOfWeek	int64	Numerical	Day of the week (0-7)
Date	object	DateTime	Date
Sales	int64	Numerical	The turnover for a day (this is the attribute to be predicted).
Customers	int64	Numerical	The number of customers in a da
Open	int64	Numerical	Indicates if store is opened or closed 0=clo 1=open
Promo	int64	Numerical	Indicates if there is a promotion running in the store this day.
StateHoliday	object	Categorical	Indicates if there is a state holiday.
SchoolHoliday	int64	Numerical	Indicates if there is a school holiday.

Store Dataset:

Column	Data Type	Type of Feature	Description
Store	int64	Numerical	Store ID
StoreType	object	Categorical	There are four different store types: a, b, c
Assortment	object	Categorical	Assortment levels: a = Basic b = Extra c = Extended
CompetitionDistance	float64	Numerical	Distance (in meters) to the nearest competitor store.
CompetitionOpen SinceMonth	float64	Numerical	Month of the competitor store opening.
CompetitionOpen SinceYear	float64	Numerical	Year of the competitor store opening.
Promo2	int64	Numerical	Indicates a continuing promotion for some stores
Promo2SinceWeek	float64	Numerical	Indicates the date since the continuing promotion started (week).
Promo2SinceYear	float64	Numerical	Indicates the date since the continuing promotion started (year).
PromoInterval	object	Categorical	Consecutive intervals Promo2 started, naming the months

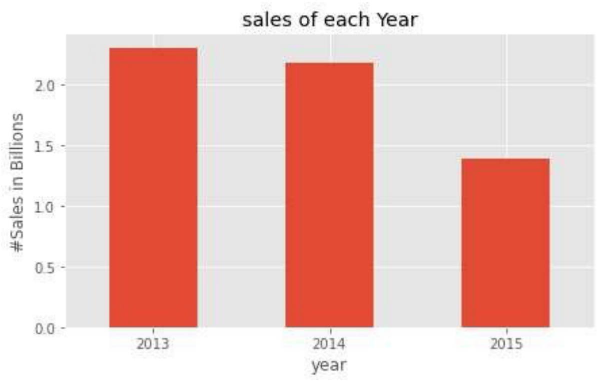
Exploratory Data Analysis:

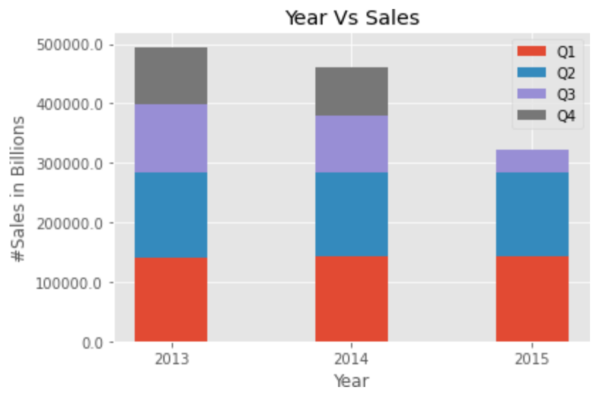
Handling missing values:

Columns	No of Missing values	% of Missing values
CompetitionDistance	3	0.26%
CompetitionOpenSinceMonth	354	31.74%
CompetitionOpenSinceYear	354	31.74%
Promo2SinceWeek	544	48.78%
Promo2SinceYear	544	48.78%
PromoInterval	544	48.78%

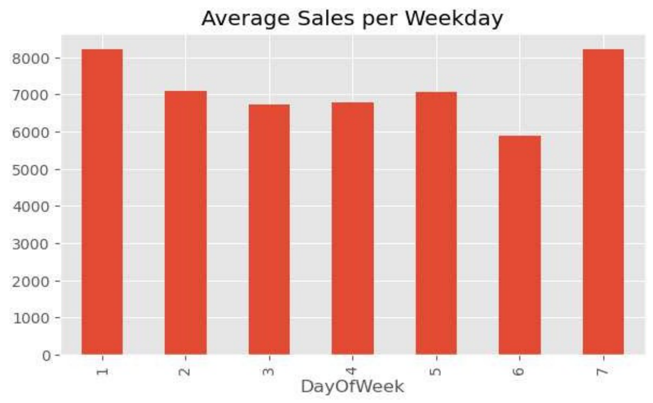
For the *Promo2SinceWeek*, *Promo2SinceYear* and *PromoInterval* columns, we have filled the *NaN* values with zeros as *promo2* column is 0 which indicates that the store is not participating in any kind of promotion.
For the *CompetitionDistance* column, we have filled the 3 missing values by taking the mean value from this column.
For the columns *CompetitionOpenSinceMonth*,*CompetitionOpenSinceYear* we have filled the missing values with the most frequent value i.e. by using the mode() method.

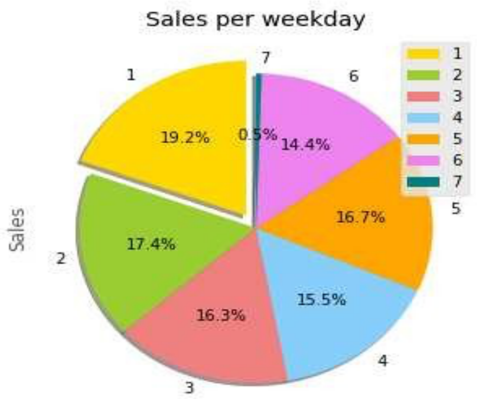
Visualizations:





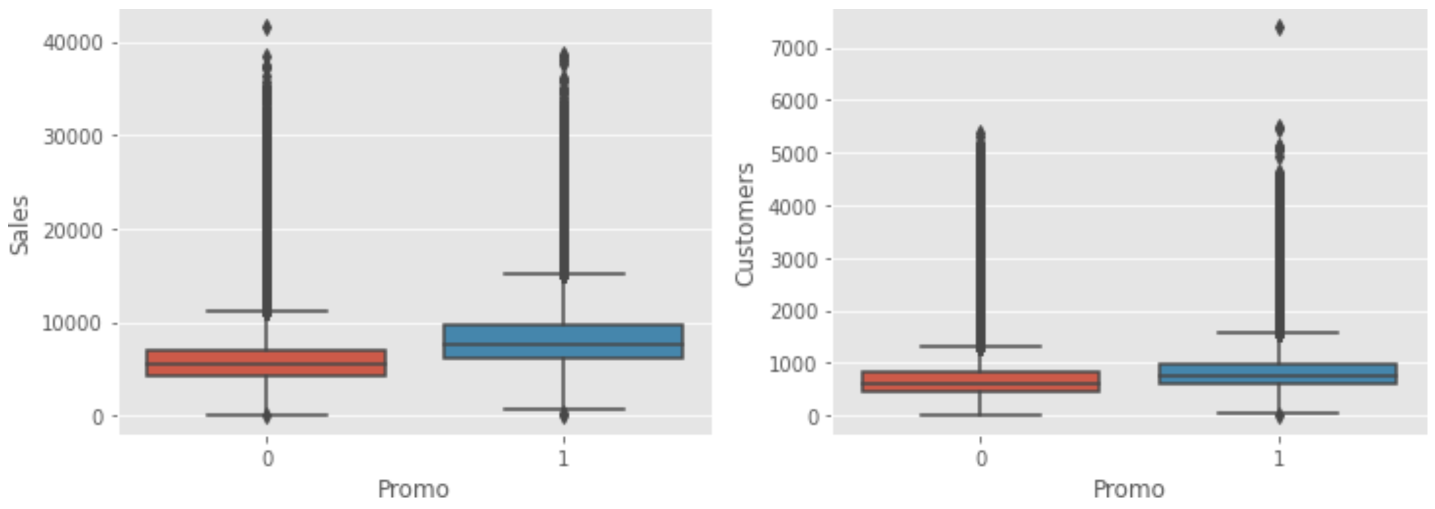
We can infer that Sales decreased from 2013 to 2015, a huge drop observed in 2015, as it has the data till July month.



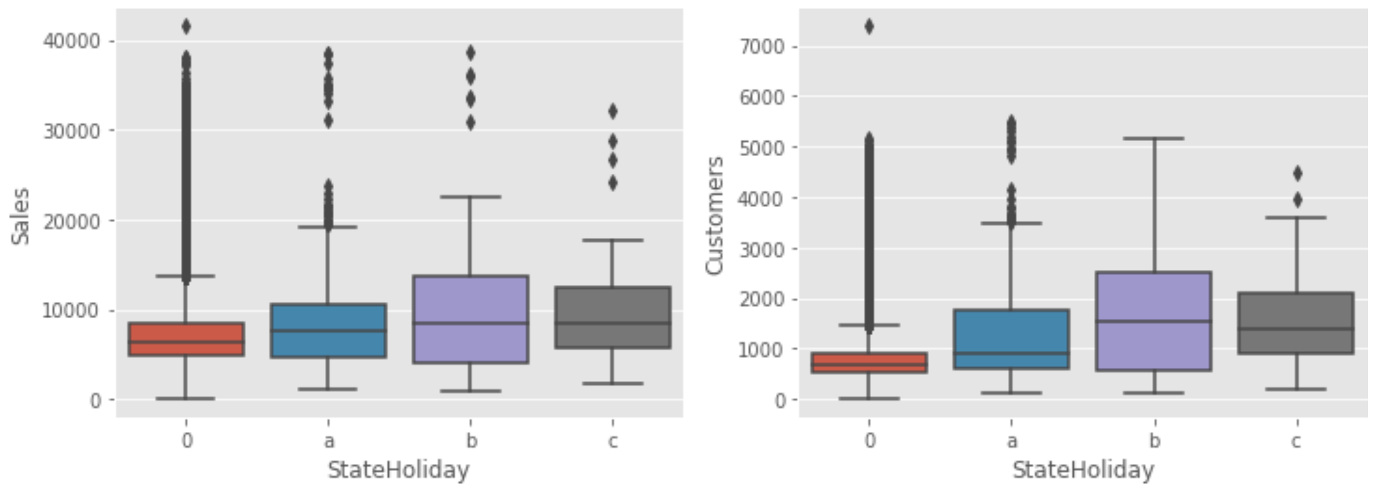


In Average sales per weekday plot, the majority of sales take place during the weekdays 1 & 7 as only store open data was considered.

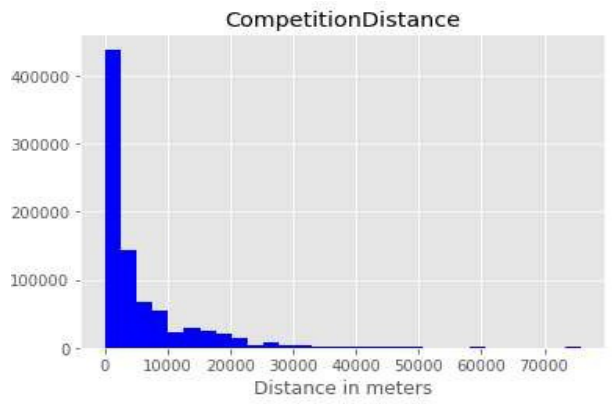
In Sales per Weekday graph, the Sales are very less on Sundays, as the most of the stores were closed.



It is obvious that sales and customers are high when the store is running a product promotion. The median sales under product promotion were significantly higher than those under no promotion. The lower quartiles for sales under promotion exceeded the median sales under no promotion. Therefore, the sales promotion variable appears to be a possible predictor of the sales volumes.

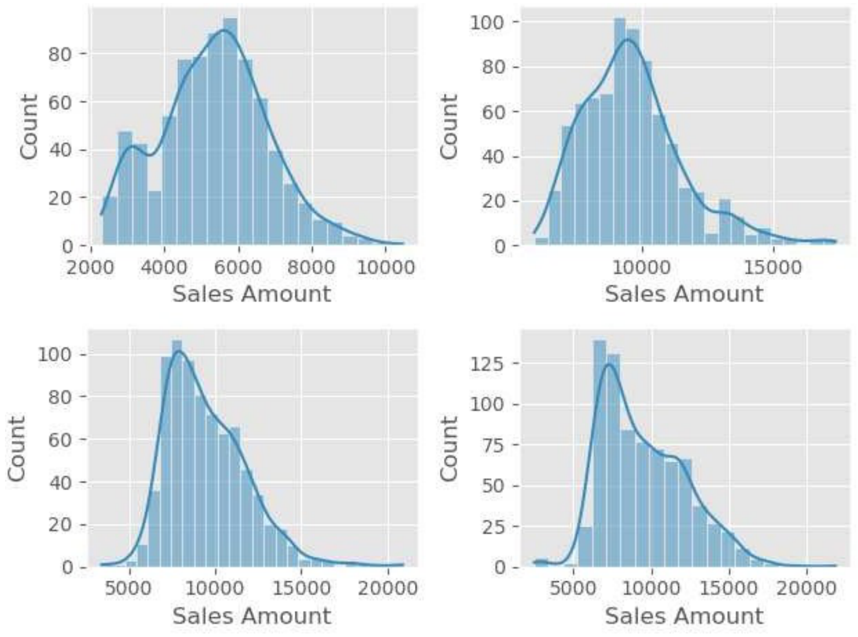


People are more likely to buy more over the Christmas and Easter holidays. Therefore, sales and customer visits are higher. However, there is minor variation in public choice during these festivals.



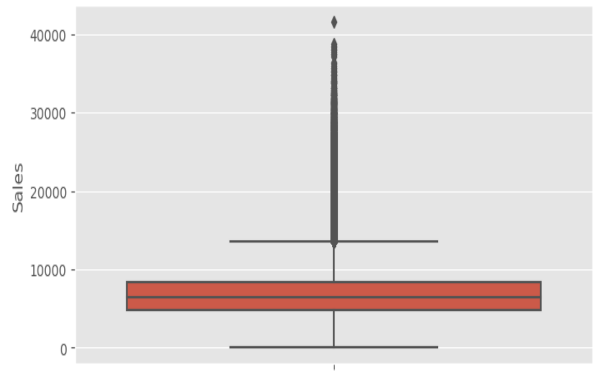
Several competitors are located near Rossman's stores within 12 miles.

*Distribution of Sales column***:**



The above plot shows that the sales are not normally distributed. They are either right skewed or left skewed. To convert sales into normal distribution we can apply log transformation on the Sales column.

Handling Outliers:



As indicated in the box plot there are some outliers which are above the whiskers that may have a negative impact on our analysis. We have used the Median Absolute deviation (MAD) method to identify and filter out the outliers from the sales column as our sales column is not normally distributed (Refer Histogram of Sales). The MAD method is used to quantify variability by determining dispersion of values in the dataset. In our case, 0.63% of outliers were identified for the sales column, due to which we have dropped these outliers.

Feature Engineering:

Date Features: Extracted basic features like year, month, day, weekofyear, dayofyear from Date.

Sales Features: Created new features using Sales and Customers columns.

- sales_perCustomer - Sales for each customer on each day
- Avg_sales_per_store - Average sales for each store
- Avg_customers_per_store - Average customers for each store
- Sales_percustomers_perstore - Average Sales for each customer and store on each day.

Holiday features: Calculated the number of holidays in a particular week for StateHoliday and SchoolHolidays. Finally, we found the correlation between all the features using df.corr(). Sales column is highly correlated with avg_customers_per_store, Promo, DayOfWeek, avg_sales_per_store, Open, Customers.

Data Preprocessing:

One hot encoding: We have three categorical features (Store Type, Assortment, StateHoliday) in our dataset which describes the store, assortment and categories in StateHoliday as machine learning models require all input and output variables to be numeric. So, we have converted the above 3 features into numeric features using One hot encoding.

Trigonometric transformations: We have time-based features such as day of week, day of year, week of year, month etc in our dataset. These features are cyclic in nature and the elegant solution to encode these cyclic features can be using mathematical formulation and trigonometry (sine, cosine) which effectively captures the cyclic nature of these features.

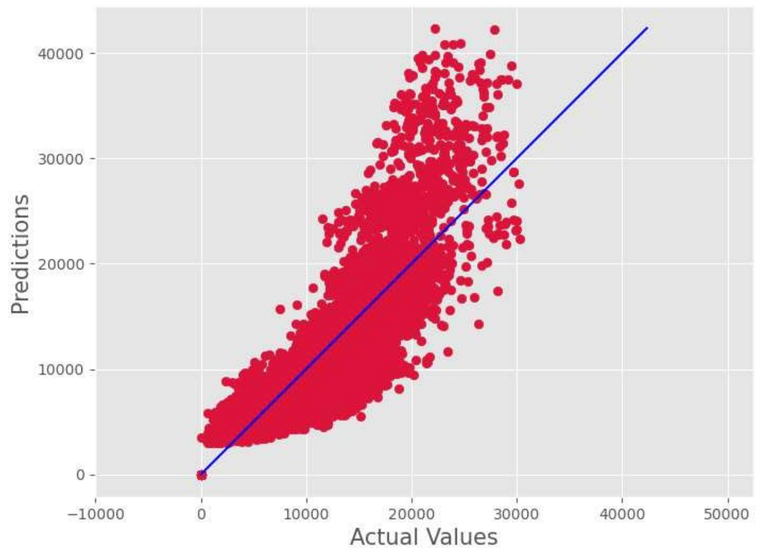
Scaling: Scaling is required to rescale the data and it's used when we want features to be compared on the same scale for our algorithm. And, when all features are in the same scale, it also helps algorithms to understand the relative relationship better. We have scaled below numeric features using StandardScaler().

Model Building and Training:

After analyzing the data, it is now time to model the data accordingly and predict the demand. The modeling process involves splitting the data into train and test, then building the model and evaluating its performance This will be performed using four different algorithms. We used supervised machine learning algorithms like Linear Regression, Decision Tree, Random Forest Regression and XGBoost to predict the Sales column. We are splitting the data into training and testing using Time Series split by considering recent data as our test dataset.

Model Building: Linear Regression

A basic linear regression model performs poorly on the testing dataset because it fails to find linearity between the target variables and predictors. The model evaluation is performed using the MAE and the RMSPE criteria. The model has a MAE of 813.58 and RMSPE of 2.3.



As linear regression doesn’t capture the trend, seasonality and other parameters of time we are going to try different models like Decision tree, Random Forest and Gradient boosting.

Model Building: Decision Tree

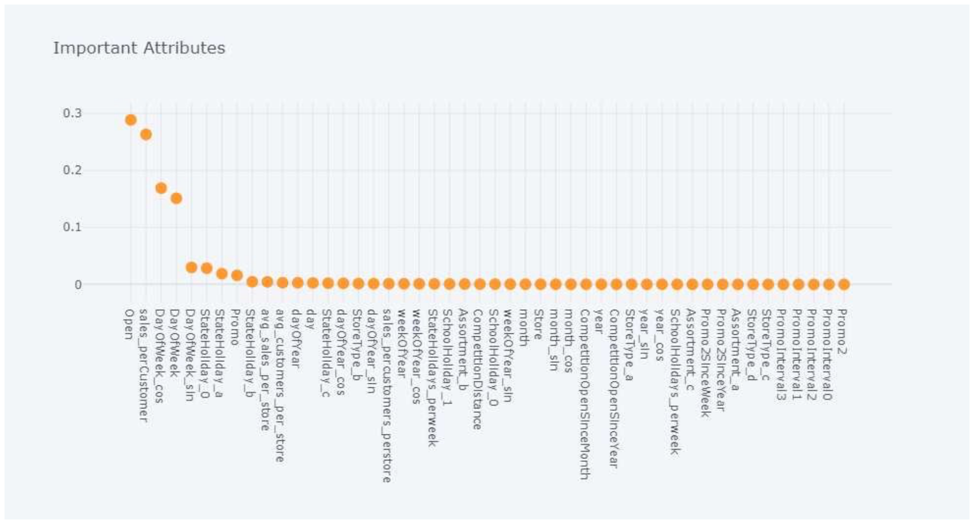
Decision Tree with 10 max_depth gives MAE of 770.94 and RMSPE of 1.85. MAE and RMSPE are reduced when compared to

Linear regression models. To get better results we performed hyperparameter tuning using GridSearchCV() which tunes the model using different max_depth:[2,4,6,8,10,12,15,20,30].

Max_depth of 20 achieved the best results. MAE and RMSPE with max_depth of 20 on test data are 651.09 and 1.53 respectively.

Model Building: RandomForest Regressor

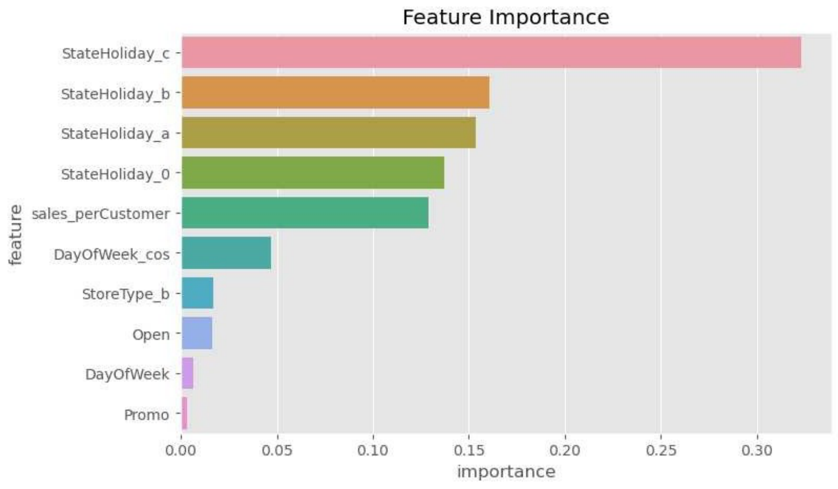
Initially the model is built with n_estimators of 100 which represents the model will be trained on 100 decision trees. RF model achieved MAE of 531.59 and RMSPE of 1.16. There is a drastic decrease in the error when compared to Linear Regression and Decision Tree. RF model yields the following results for variable importance. The top five most important variables were the Open, Sales per customer, day of the week_cos, day of the week_sin.



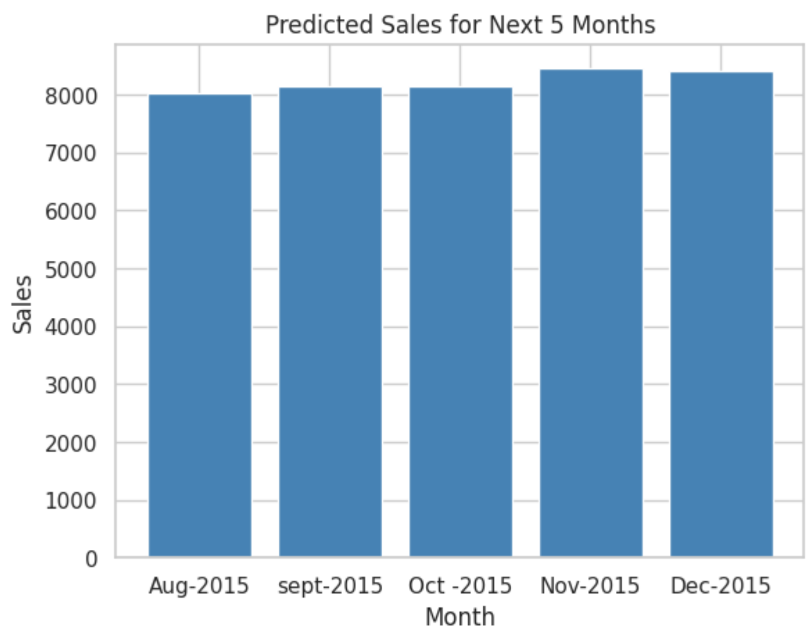
Model Building: XGBoost Regressor

Gradient Boosting Tree, like other boosting methods, creates decision trees. Gradient Boosting Tree's primary notion is to generate the tree as an optimization procedure on an appropriate cost function. Built a basic XGboost model with n_estimators 200 has achieved MAE of 497.99 and RMSPE of 1.15. To get better results we performed hyperparameter tuning using GridSearchCV() which tunes the model using different parameters and best results are achieved with n_estimators=1000, learning_rate=0.2, max_depth=10, subsample=0.9, colsample_bytree=0.7

The XGboost model yields the following results for variable importance. The top five most important variables were the StateHolidays, Sales_perCustomer, Dayofweek_cos. MAE and RMSPE on test data is 450.41 and 1.054 respectively.

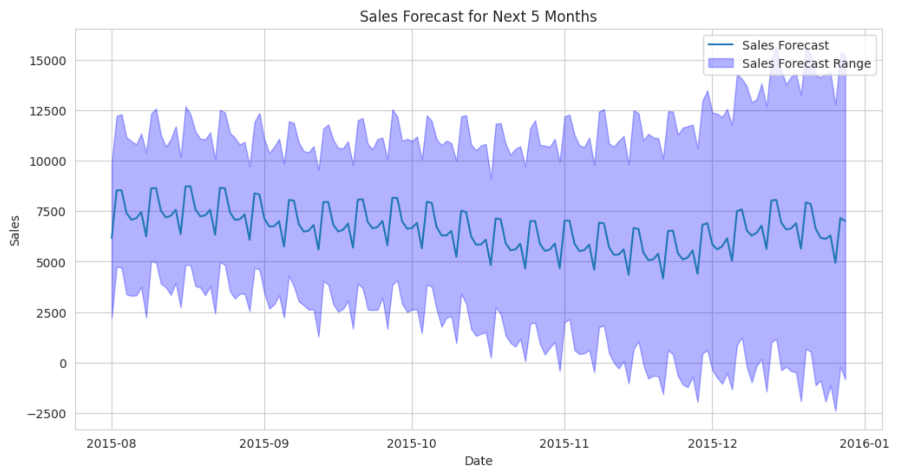


Forecasting sales using XGB:



We used XGB model to forecast sales for 5 months data i.e., from Aug to Dec 2015. And plotted a bar chart for the same. The above image infers that the sales during the months of November and December is relatively high when compared to the rest of the months. The reason could be because of the Holidays & Promotions running during those months.

Forecasting sales using FB Prophet model:



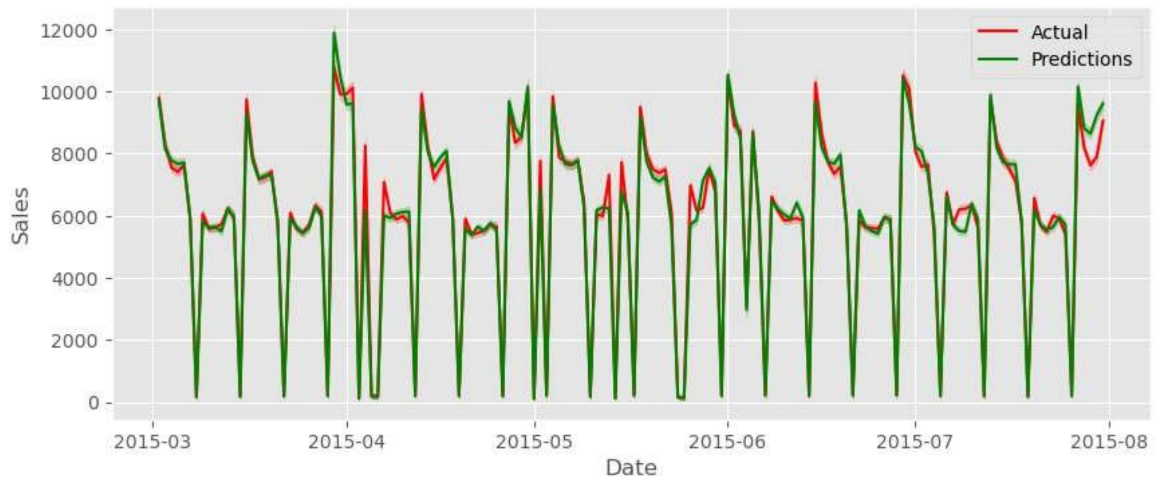
We used FB Prophet model to forecast sales for 6 months data i.e., from Aug 2015 to Jan 2016. And plotted a forecast chart for the same. The above image infers that the sales during the months of November, December and January is relatively high when compared to the rest of the months. The reason could be because of the Holidays & Promotions running during those months.

Comparison of Models:

Evaluation metric: We choose Root Mean Squared Error (RMSE) and Root Mean Squared Percentage error (RMSPE) as the evaluation metrics for our machine learning model. We aim to keep the error between the actual value and the predicted value of the target variable (Sales) minimum. We choose ‘RMSPE’ as it is most useful when large errors are particularly undesirable.

Models	MAE	RMSE	RMSPE
Linear Regression	813.58	1315.7	2.3
Decision Tree	651.09	1065.7	1.53
Random Forest	531.59	877.83	1.16
XGBoost	450.41	716.34	1.05
FB Prophet	2306.75	2978.56	57.55

The MAE and RMSPE error measurements were used to calculate the model's overall performance scores. The study found that the XGboost algorithm produces more accurate forecasts than the random forest, Decision tree and linear regression models. Because the XGboost algorithm learns from the error produced from the previous model. The XGboost technique is easier to use than the random forest model and is less time-consuming.



Above graph shows the comparison of actual sales and predicted sales on test data for the XGBoost model. We can see that there is very minimal error.

Limitations:

- Handling large amounts of sales data (10,17,209 observations on 13 variables) was challenging due to high latency in model execution.
- Around 180 stores were closed for 6 months. Unable to fill the gap of sales for those stores.
- Prediction of sales for individual stores(out of 1115) and most stores have different patterns of sales. A single model cannot justify all stores.

Conclusion:

To conclude, the XGBoost technique was the most accurate because of its ability to master real-time dynamics and improve pattern identification and prediction from the current window. It is pretty computationally tricky; however, with the right amount of training data, it will be much more accurate and an excellent forecast to estimate demand and plan for resources in advance.

References:

- https://www.kaggle.com/c/rossmann-store-sales/data
- https://towardsdatascience.com/tagged/rossmann
- https://www.analyticsvidhya.com/blog/tag/rossmann-store-sales/
- https://www.datacamp.com/courses/sales-forecasting-in-python
- https://youtu.be/NWONeJKn6kc