Regression Model on Retail Sales Prediction

Submitted By:
Lakshmi Narayana
Data Science Trainee
At Alma Better

Default Prediction

- Problem Statement
- Exploratory analysis
- Feature selection
- Data preparation
- Implementation of model
- Evaluation of Model
- Conclusion

Problem Statement

Rossmann operates over 3,000 drug stores in 7 European countries. Currently, Rossmann store managers are tasked with predicting their daily sales for up to six weeks in advance. Store sales are influenced by many factors, including promotions, competition, school and state holidays, seasonality, and locality. With thousands of individual managers predicting sales based on their unique circumstances, the accuracy of results can be quite varied.

Data Pipeline

- **Data processing-:** At first phase checked for null values and changed the datetime containing column in sales dataset.
- **EDA:** Exploratory analysis was done on the features selected in the first phase.
- **Data processing-2:** Checked for null values in stores dataset and dropped features having more null values.
- **EDA:** Combined the both datasets and made eda on the above selected features.
- **Feature Selection:** Used VIF to check the correlation among the variables.
- **Create a model:** Finally in this part created models, trained and tested it on the available dataset.

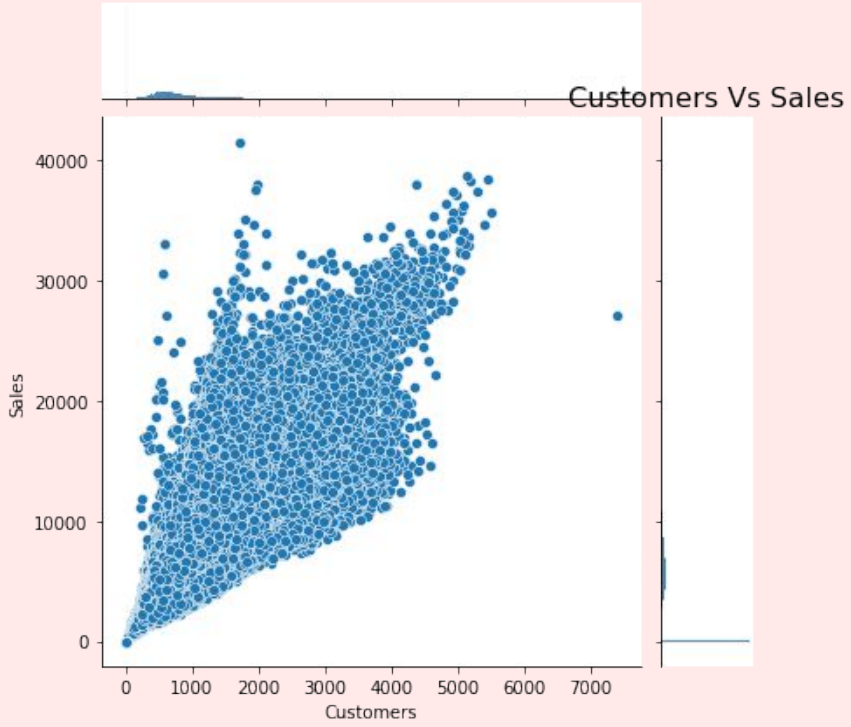
Data Description

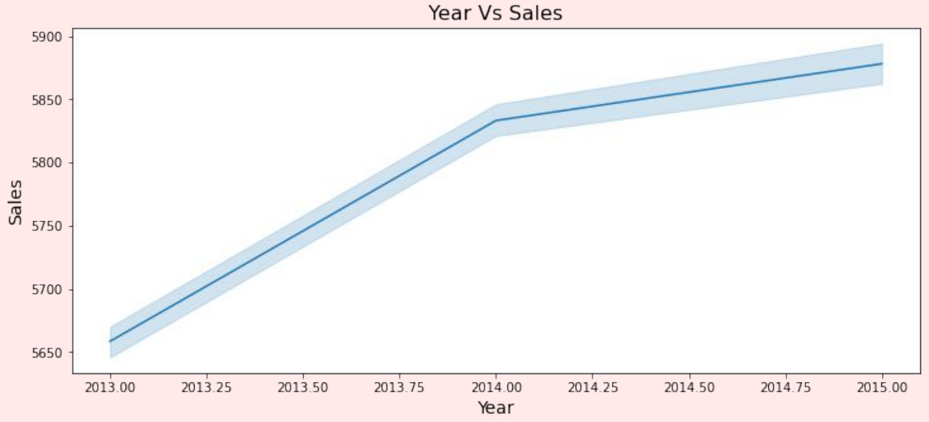
Most of the fields are self-explanatory. The following are descriptions for those that aren't.

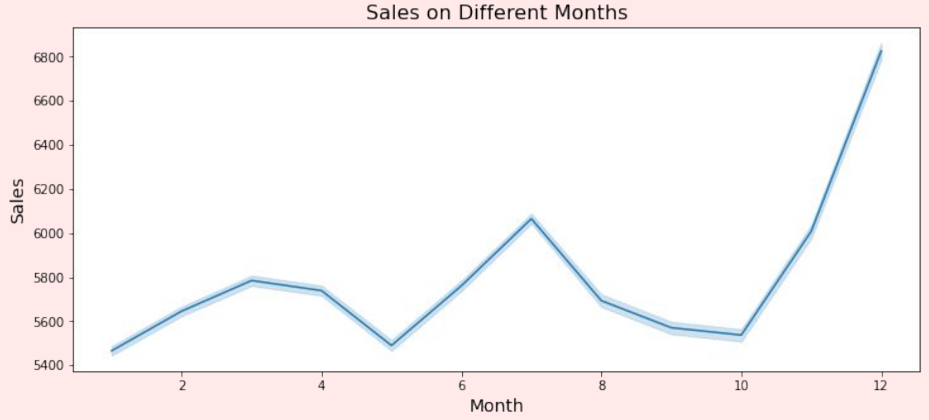
- Id - an Id that represents a (Store, Date) duple within the test set
- Store - a unique Id for each store
- Sales - the turnover for any given day
- Customers - the number of customers on a given day
- Open - an indicator for whether the store was open: 0 = closed, 1 = open
- State Holiday - indicates 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
- School Holiday - 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
- Competition Distance - distance in meters to the nearest competitor store
- Competition OpenSince[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
- Promo2 Since[Year/Week] - describes the year and calendar week when the store started participating in Promo2
- Promo Interval - 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

Defining the Dependent Variable

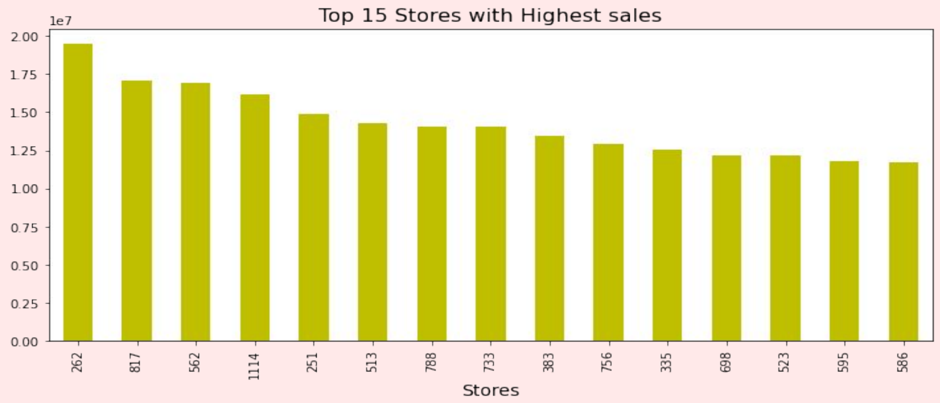
Sales - the turnover for any given day

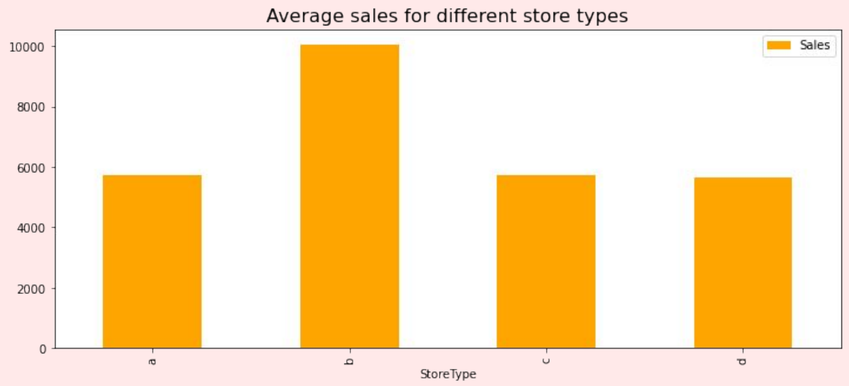


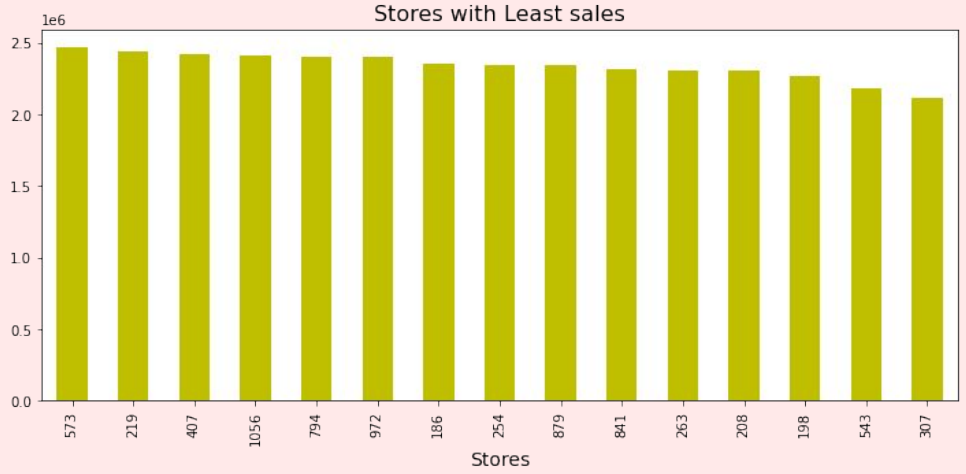


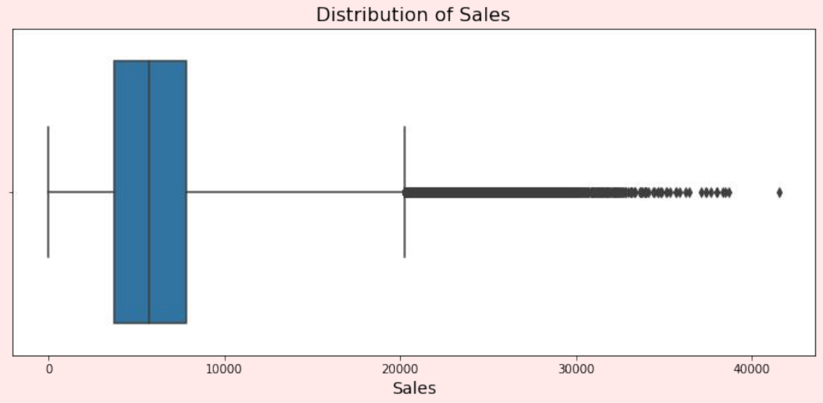


Defining the Dependent Variable



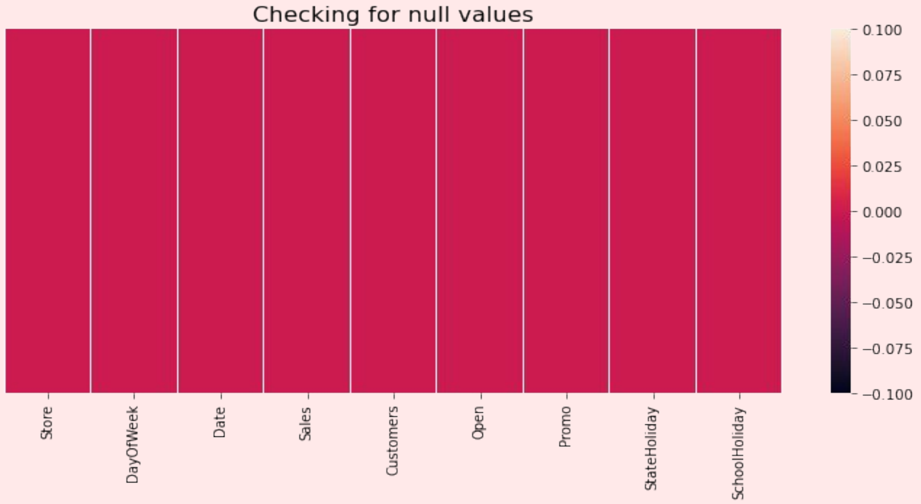


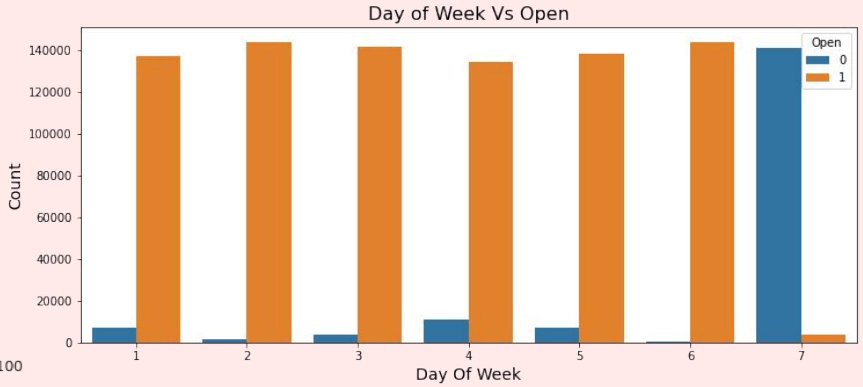


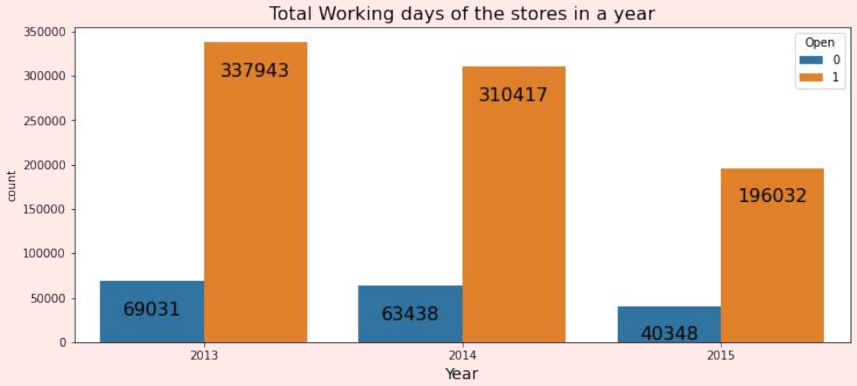


Exploratory Data Analysis

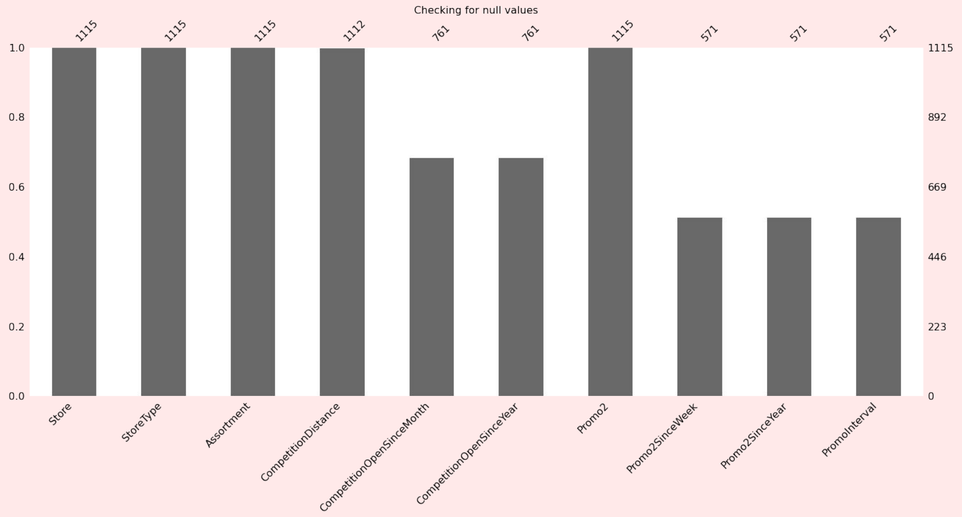
- Cleaning the null values.
- Changing the datetime format.
- Merged the datasets.
- Dropped some features.
- One hot encoding.
- Checking for Multicollinearity.

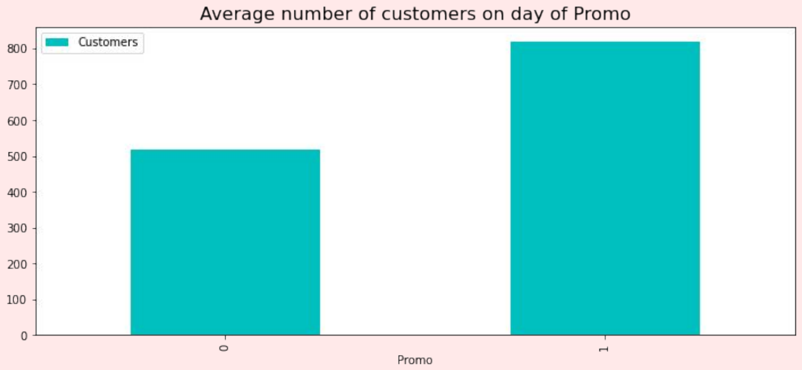


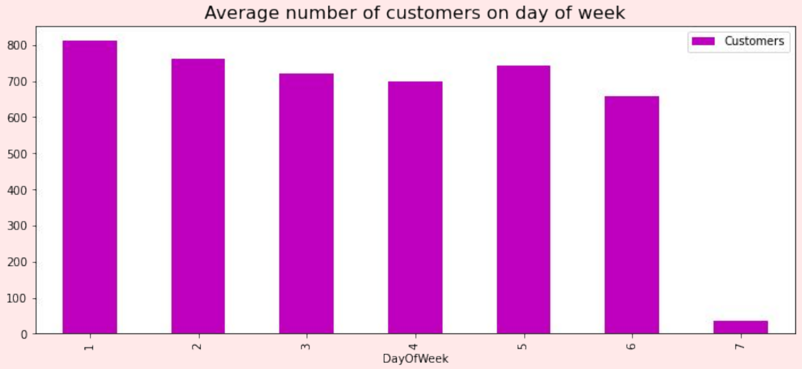


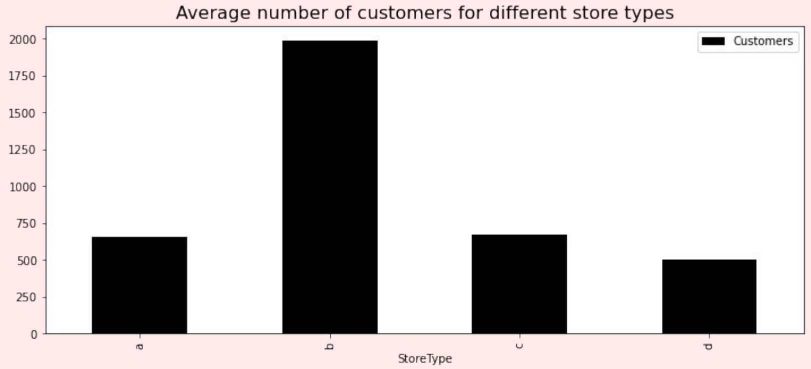


Exploratory Data Analysis

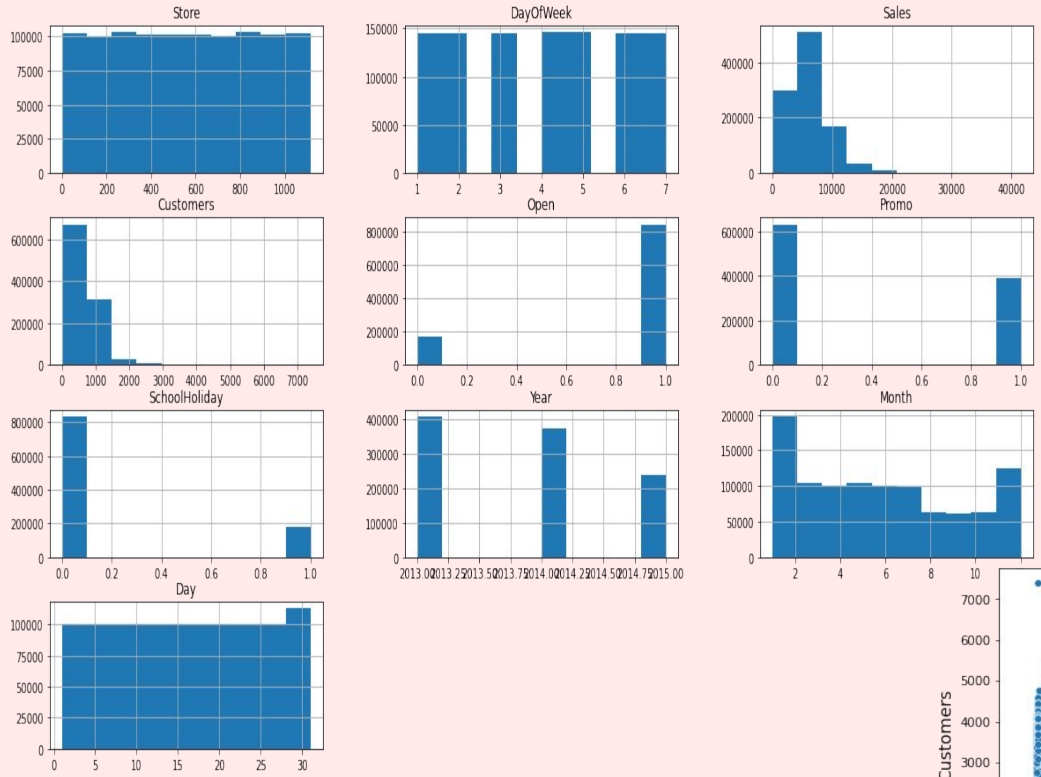


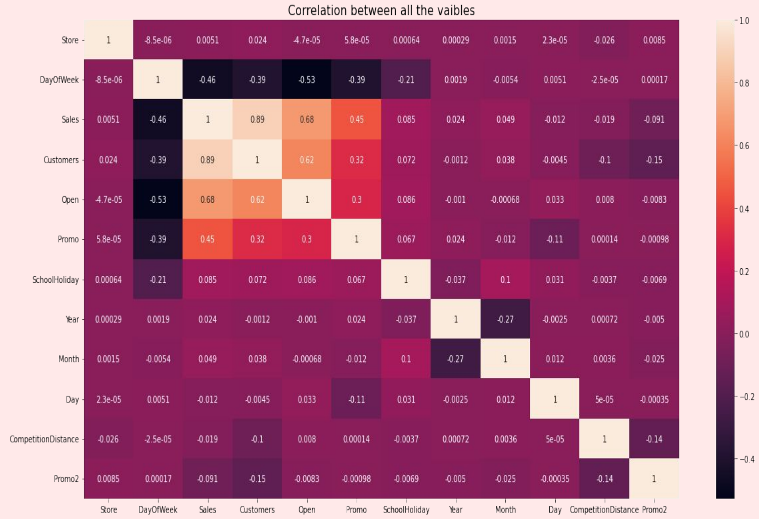




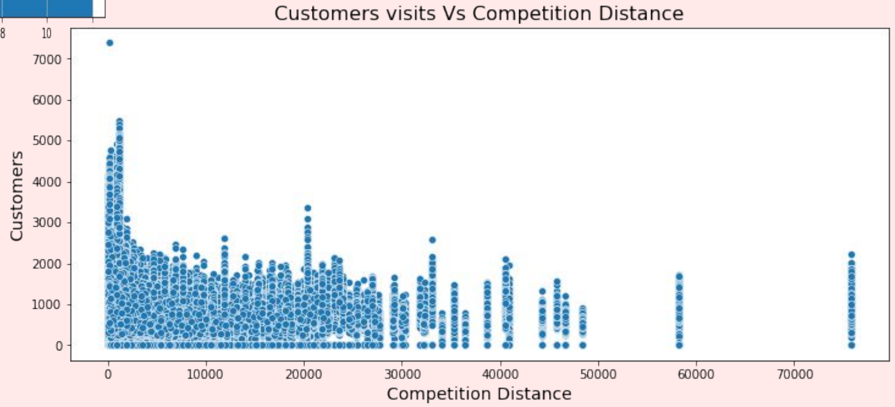


Exploratory Data Analysis



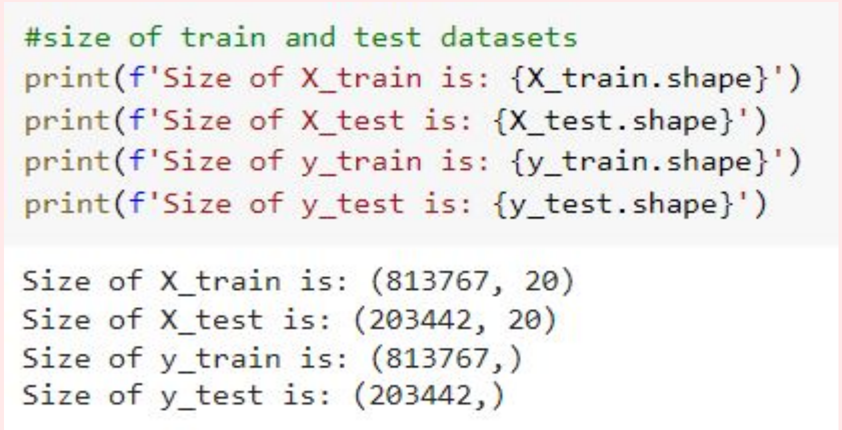


	Store	DayOfWeek	Sales	Customers	Open	Promo	SchoolHoliday	Year	Month	Day	CompetitionDistance	Promo2
Store	1	-8.5e-06	0.0051	0.024	-4.7e-05	5.8e-05	0.00064	0.00029	0.0015	2.3e-05	-0.026	0.0085
DayOfWeek	-8.5e-06	1	-0.46	-0.39	-0.53	-0.39	-0.21	0.0019	-0.0054	0.0051	-2.5e-05	0.00017
Sales	0.0051	-0.46	1	0.89	0.68	0.45	0.085	0.024	0.049	-0.012	-0.019	-0.091
Customers	0.024	-0.39	0.89	1	0.62	0.32	0.072	-0.0012	0.038	-0.0045	-0.1	-0.15
Open	-4.7e-05	-0.53	0.68	0.62	1	0.3	0.086	-0.001	-0.00068	0.033	0.008	-0.0083
Promo	5.8e-05	-0.39	0.45	0.32	0.3	1	0.067	0.024	-0.012	-0.11	0.00014	-0.00098
SchoolHoliday	0.00064	-0.21	0.085	0.072	0.086	0.067	1	-0.037	0.1	0.031	-0.0037	-0.0069
Year	0.00029	0.0019	0.024	-0.0012	-0.001	0.024	-0.037	1	-0.27	-0.0025	0.00072	-0.005
Month	0.0015	-0.0054	0.049	0.038	-0.00068	-0.012	0.1	-0.27	1	0.012	0.0036	-0.025
Day	2.3e-05	0.0051	-0.012	-0.0045	0.033	-0.11	0.031	-0.0025	0.012	1	5e-05	-0.00035
CompetitionDistance	-0.026	-2.5e-05	-0.019	-0.1	0.008	0.00014	-0.0037	0.00072	0.0036	5e-05	1	-0.14
Promo2	0.0085	0.00017	-0.091	-0.15	-0.0083	-0.00098	-0.0069	-0.005	-0.025	-0.00035	-0.14	1



Data Preparation

- The 2 datasets are merged and made a new dataframe.
- One hot encoding was done for categorical feature.
- Variance inflation factor was used to check the correlation among the variables.
- Few features are dropped.
- Divided the dataset into train and test set in the ratio of 80:20
- StandardScaler was used to scale the data.



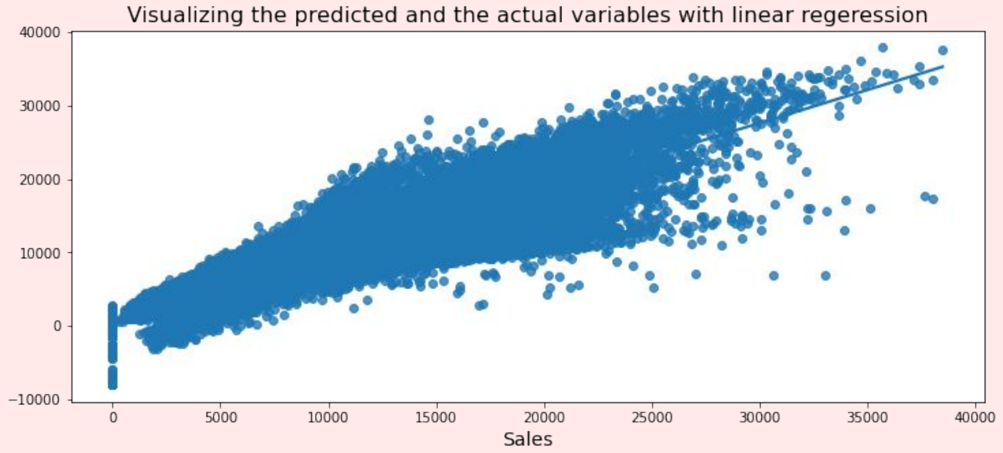
```
#size of train and test datasets
print(f'Size of X_train is: {X_train.shape}')
print(f'Size of X_test is: {X_test.shape}')
print(f'Size of y_train is: {y_train.shape}')
print(f'Size of y_test is: {y_test.shape}')
```

```
Size of X_train is: (813767, 20)
Size of X_test is: (203442, 20)
Size of y_train is: (813767,)
Size of y_test is: (203442,)
```

Model Implementation

Linear Regression:

- The Scaled data was used for the model implementation.
- Fit the training dataset to the model.
- The regressor score of the model is 0.90067
- Defining the predicted values form the model.



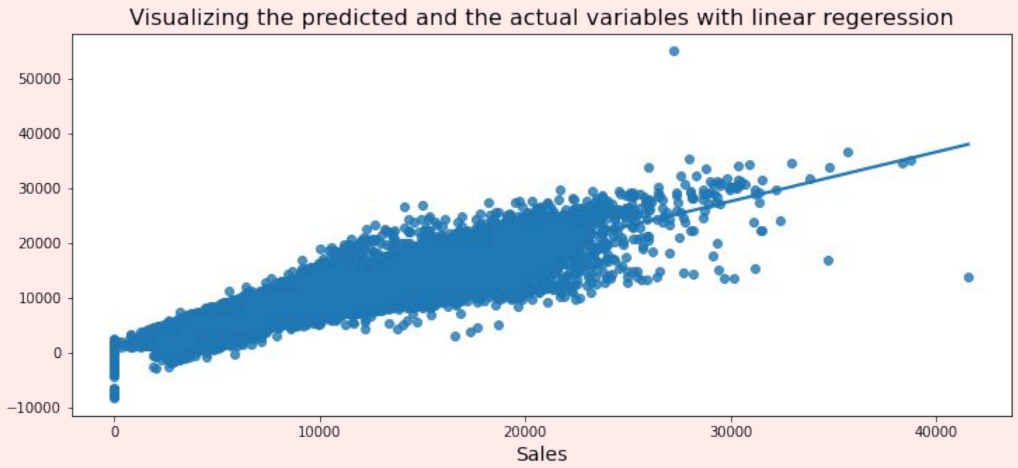
Linear Regression

```
#Evaluating the training dataset
#Mean Squared Error
MSE_train = mean_squared_error(y_train, pred_train)
print(f'MSE= {MSE_train}')

#Root Mean Squared Error
RMSE_train = np.sqrt(MSE_train)
print(f'RMSE= {RMSE_train}')

#R2_Score
R2_Score_train = r2_score(y_train, pred_train)
print(f'R2_Score= {R2_Score_train}')
```

```
MSE= 1473080.6552463644
RMSE= 1213.7053411954503
R2_Score= 0.9006701662380696
```



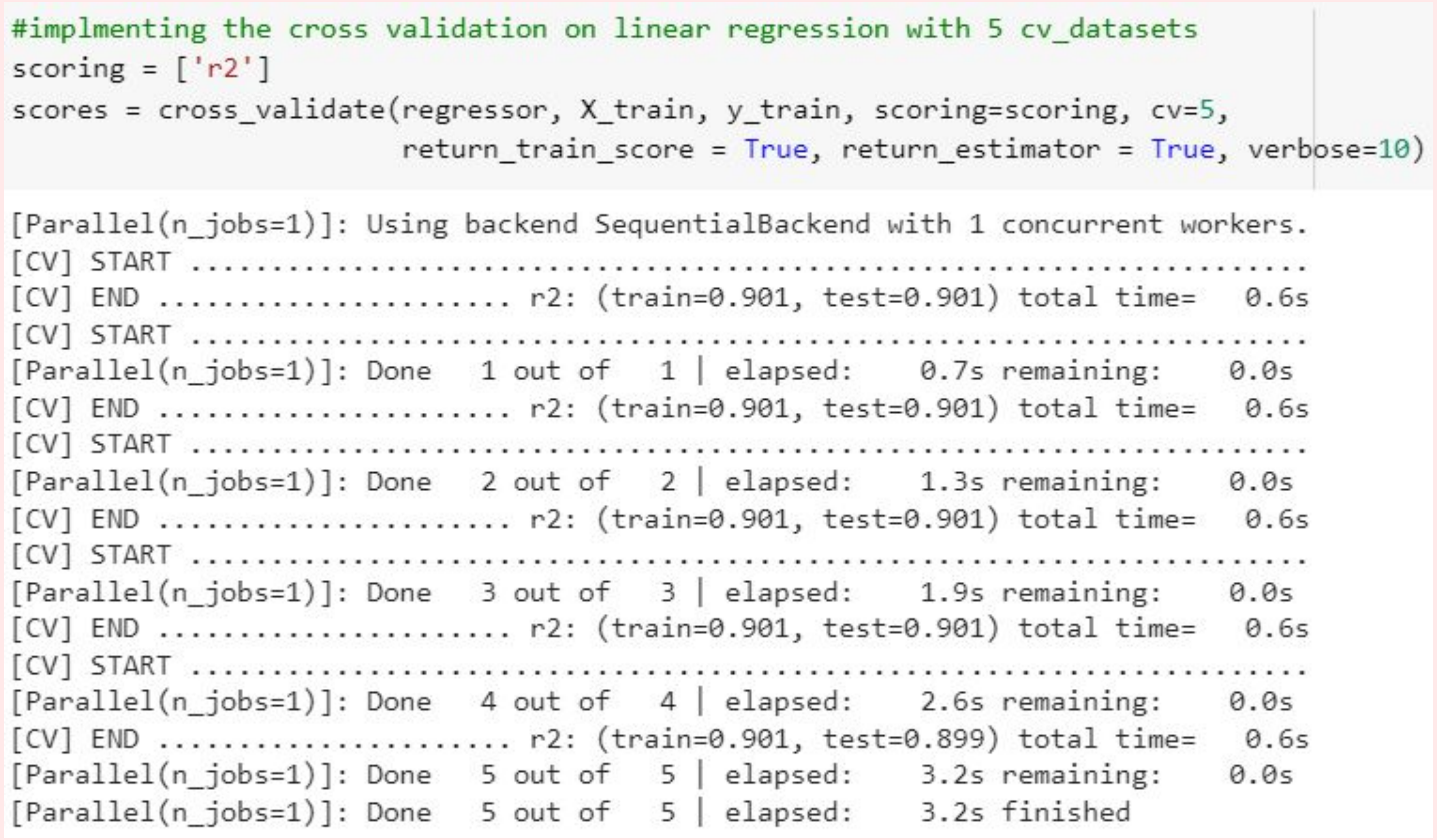
```
#Evaluating the test dataset
#Mean Squared Error
MSE_test = mean_squared_error(y_test, pred_test)
print(f'MSE= {MSE_test}')

#Root Mean Squared Error
RMSE_test = np.sqrt(MSE_test)
print(f'RMSE= {RMSE_test}')

#R2_Score
R2_Score_test = r2_score(y_test, pred_test)
print(f'R2_Score= {R2_Score_test}')
```

```
MSE= 1488637.608160531
RMSE= 1220.0973765075191
R2_Score= 0.8993401767895278
```

Cross Validation on Linear Regression



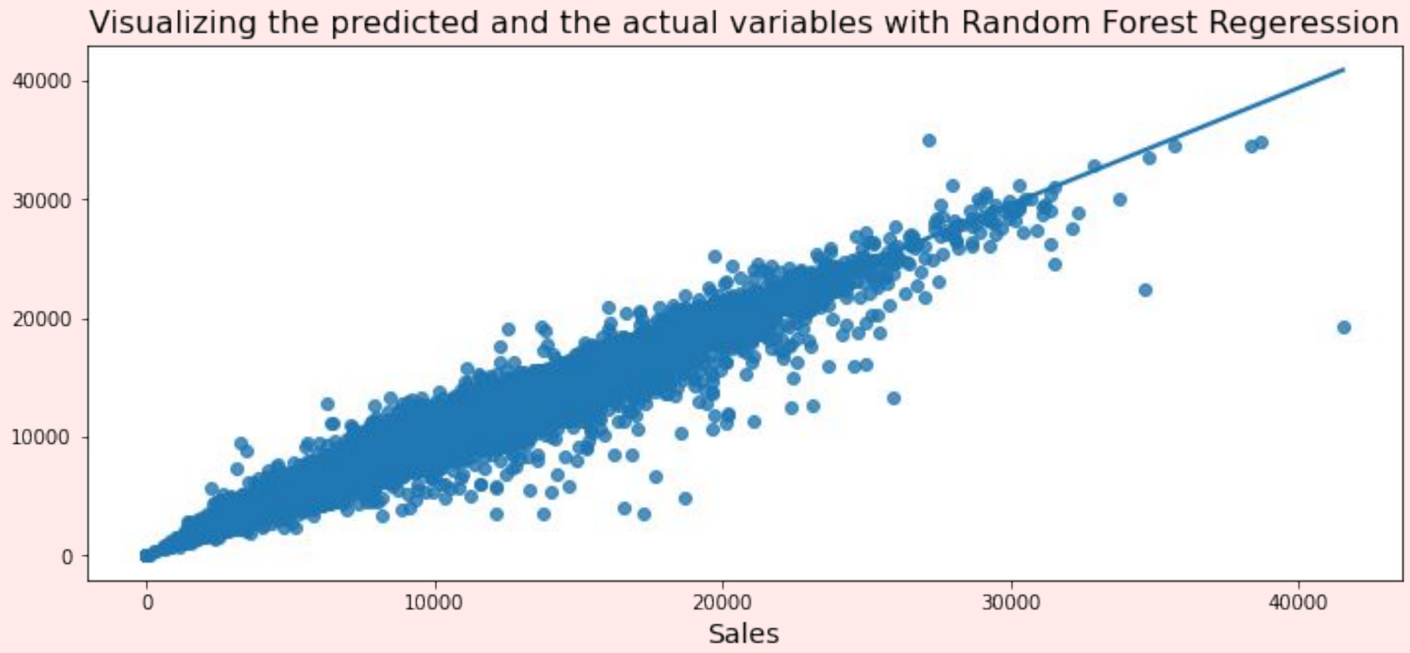
```python
#implmenting the cross validation on linear regression with 5 cv_datasets
scoring = ['r2']
scores = cross_validate(regressor, X_train, y_train, scoring=scoring, cv=5,
                        return_train_score = True, return_estimator = True, verbose=10)
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[CV] START .....................................................................
[CV] END ...................... r2: (train=0.901, test=0.901) total time=   0.6s
[CV] START .....................................................................
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.7s remaining:    0.0s
[CV] END ...................... r2: (train=0.901, test=0.901) total time=   0.6s
[CV] START .....................................................................
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    1.3s remaining:    0.0s
[CV] END ...................... r2: (train=0.901, test=0.901) total time=   0.6s
[CV] START .....................................................................
[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    1.9s remaining:    0.0s
[CV] END ...................... r2: (train=0.901, test=0.901) total time=   0.6s
[CV] START .....................................................................
[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:    2.6s remaining:    0.0s
[CV] END ...................... r2: (train=0.901, test=0.899) total time=   0.6s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    3.2s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    3.2s finished
```

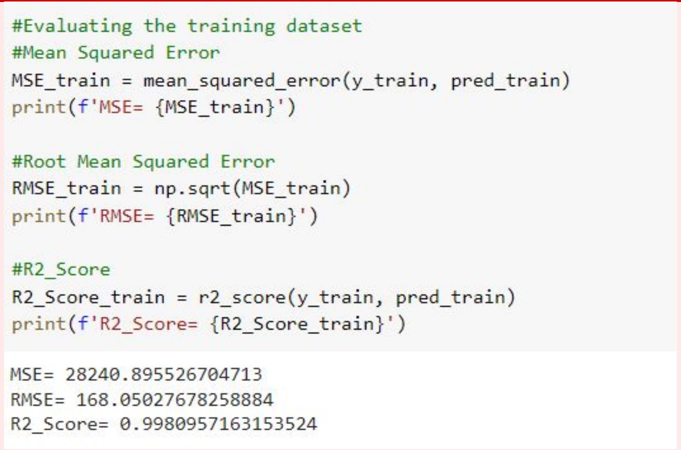
Model Implementation

Random Forest Regressor:

- The Scaled data was stored in the dataframe.
- Fit the training dataset to the model.
- Defining the predicted values form the model.



Random Forest Regressor

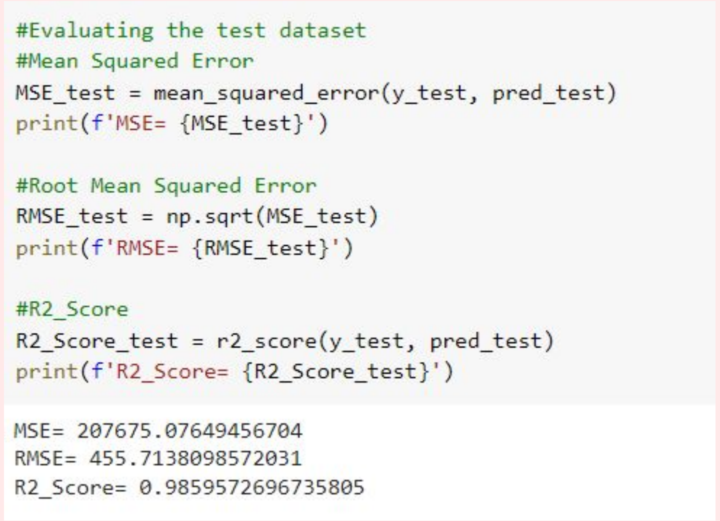


```
#Evaluating the training dataset
#Mean Squared Error
MSE_train = mean_squared_error(y_train, pred_train)
print(f'MSE= {MSE_train}')

#Root Mean Squared Error
RMSE_train = np.sqrt(MSE_train)
print(f'RMSE= {RMSE_train}')

#R2_Score
R2_Score_train = r2_score(y_train, pred_train)
print(f'R2_Score= {R2_Score_train}')
```

```
MSE= 28240.895526704713
RMSE= 168.05027678258884
R2_Score= 0.9980957163153524
```



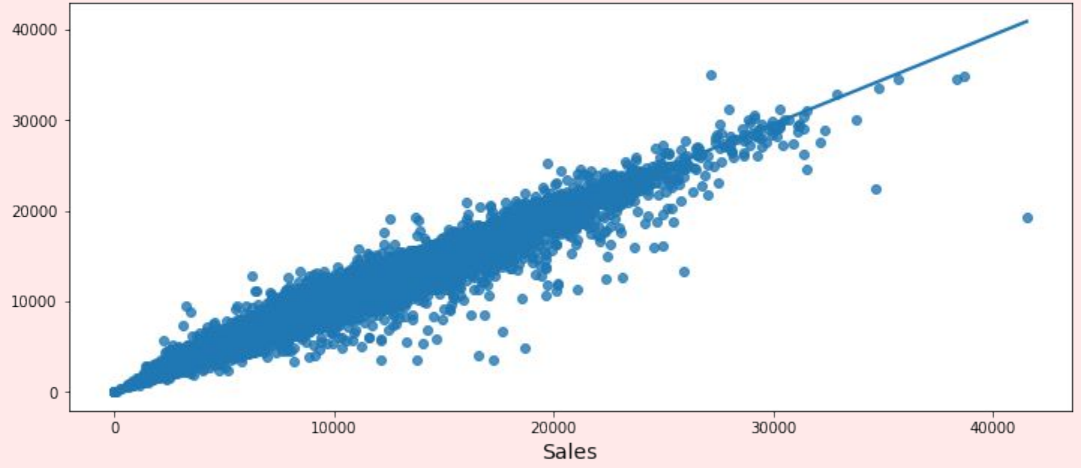
```
#Evaluating the test dataset
#Mean Squared Error
MSE_test = mean_squared_error(y_test, pred_test)
print(f'MSE= {MSE_test}')

#Root Mean Squared Error
RMSE_test = np.sqrt(MSE_test)
print(f'RMSE= {RMSE_test}')

#R2_Score
R2_Score_test = r2_score(y_test, pred_test)
print(f'R2_Score= {R2_Score_test}')
```

```
MSE= 207675.07649456704
RMSE= 455.7138098572031
R2_Score= 0.9859572696735805
```

Visualizing the predicted and the actual variables with Random Forest Regeression



Conclusion

- In the months of March, July and December we have high sales in those 4 month periods. In all over months we have maximum sales in the month of December.
- Most of the stores are closed during sunday.
- Sales are highly correlated with the customers.
- Customer’s purchasing index lies between 0 to 10000.
- Store number 733 has the highest number of customer visits and store number 543 has the least.
- Store number 262 has the highest sales and store number 307 has the least sales.
- Store type b is having the more number of customer visits and sales.
- Competitor stores are mostly in the range of 0 to 30000 meters of distance .
- Linear Regression and Random Forest Regression are used to train the model.
- Cross validation is also done on the linear regression model.
- When it comes to the accuracy the Random Forest Regression is performing well on the test dataset with the accuracy of 98.60%. As it can predict the daily sales of stores for up to six weeks in advance with the same accuracy.

Thank You