Supervised ML-Regression

Yes Bank Stock Closing Price Prediction

Team Members

Ankit Tiwari , Behara Sai Keerthi Mitesh Bhanushali , Abhishek Anand , Shivashish

Points for Discussion

- Problem Statement
- Introduction
- Exploratory Data Analysis
- FBProphet
- Regression models on Time Series dataset
- Regression models without lag columns
- Auto Arima
- Conclusion

Problem Statement

Yes Bank is a well-known bank in the Indian financial domain. Since 2018, it has been in the news because of the fraud case involving Rana Kapoor. Owing to this fact, it was interesting to see how that impacted the stock prices of the company and whether Time series models or any other predictive models can do justice to such situations. This dataset has monthly stock prices of the bank since its inception and includes closing, starting, highest, and lowest stock prices of every month. The main objective is to predict the stock's closing price of the month.

Introduction

The given dataset consist of 185 rows and 5 columns, the columns description is as follow:

1. The dataset contains multiple variables - date, open, high, low and close.
2. The column date contains the month and the year of the price of the share.
3. The columns Open and Close represent the starting and final price at which the stock is traded in a particular month.
4. High and Low represent the maximum and minimum price of the share for the month.
5. The profit or loss calculation is usually determined by the closing price of a stock for the month, hence we will consider the closing price as the target variable.

Sample of data

	Date	Open	High	Low	Close
0	Jul-05	13.00	14.00	11.25	12.46
1	Aug-05	12.58	14.88	12.55	13.42
2	Sep-05	13.48	14.87	12.27	13.30
3	Oct-05	13.20	14.47	12.40	12.99
4	Nov-05	13.35	13.88	12.88	13.41

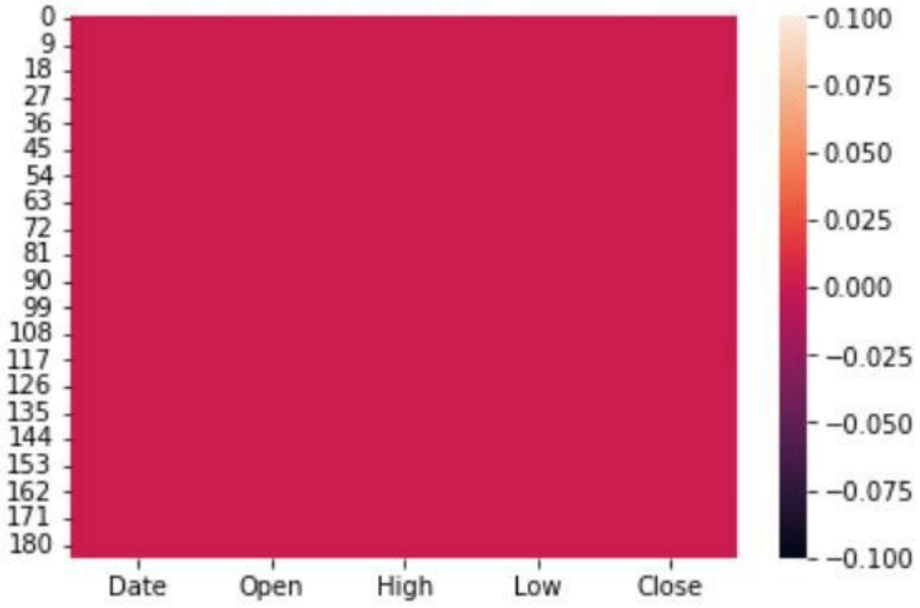
The stock prices in this dataset are from the year 2005 to 2021

Heatmap for showing the null value

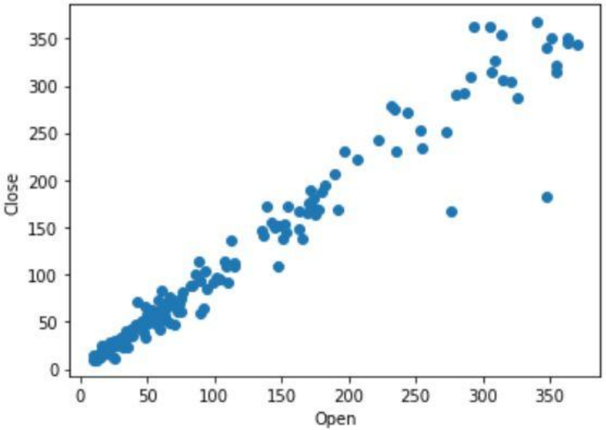
This graph shows that the given data does not contains any null values .

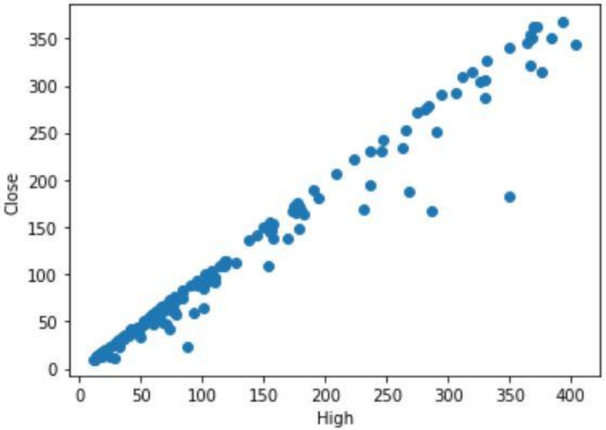
```
sns.heatmap(stock_data.isnull(),cbar=True)
```

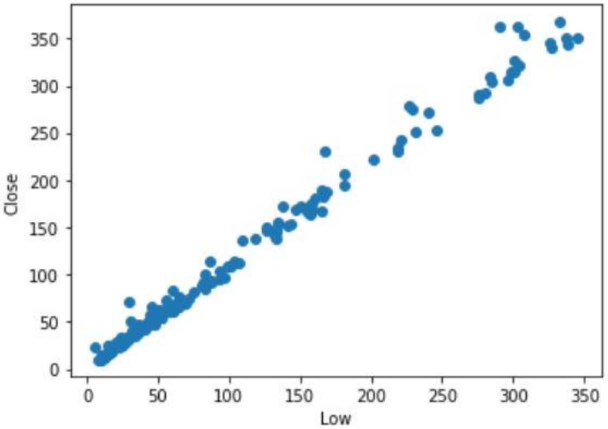
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f661ad42e10>
```

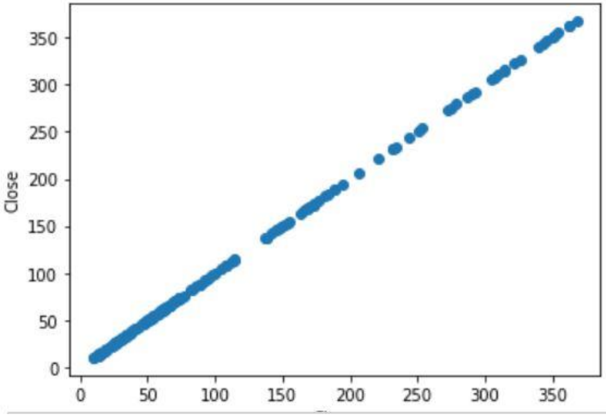


Checking the linear relationship of open, high and low with our target variable Close





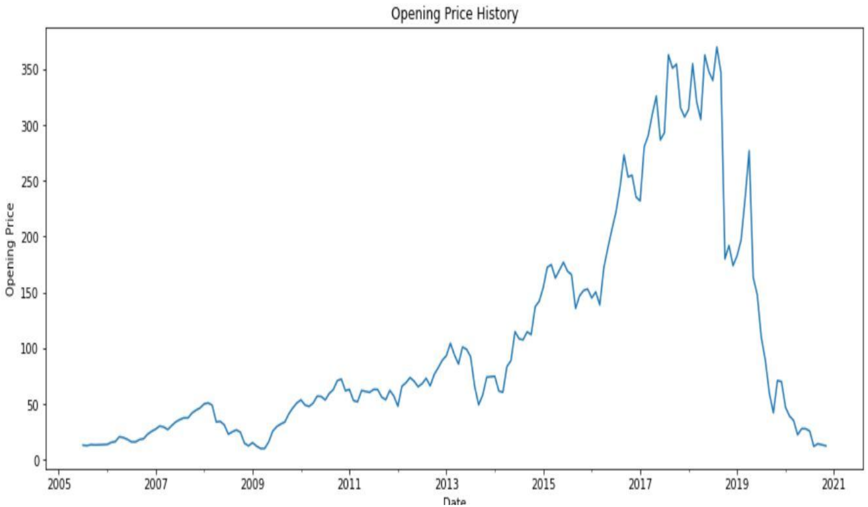




Above graph clearly shows that all features are linearly correlated.

Comparing Close price and Open price

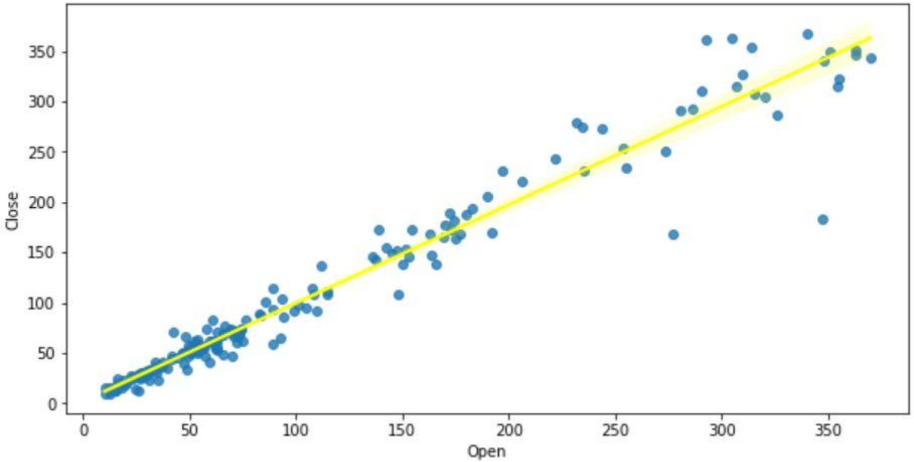
This line plot is on close price depending on date .

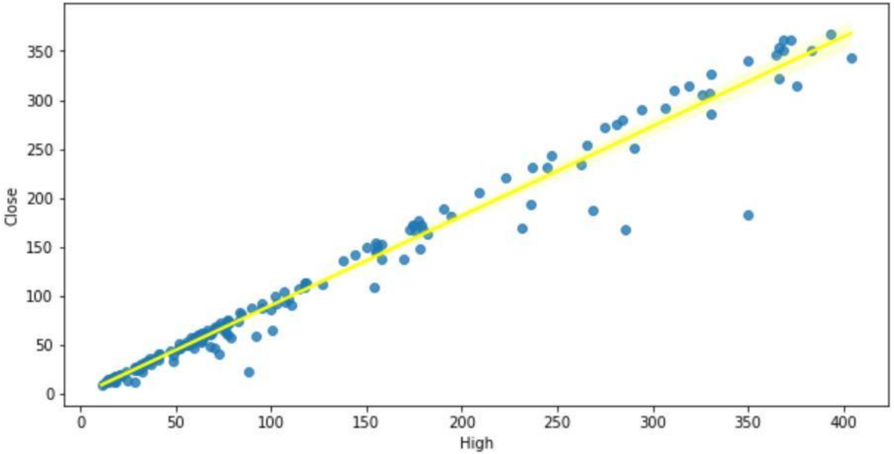


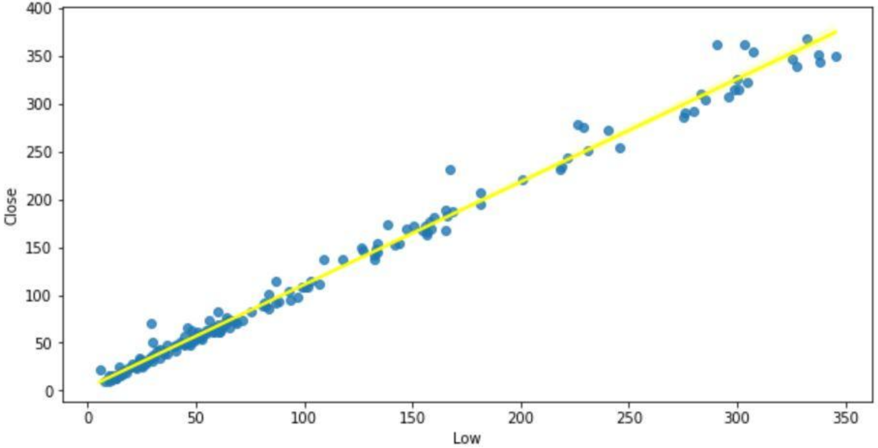
This line plot is on Open price depending on date .

As we can observe from the above plots that the opening and the closing prices of the shares are almost same.

Regression Plots on all the features with target value.

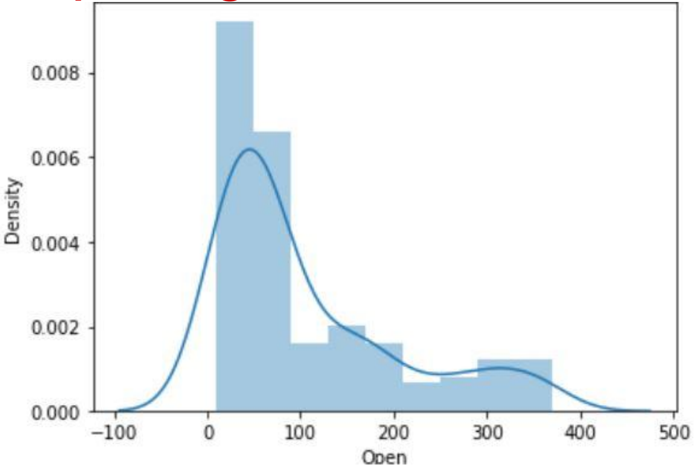


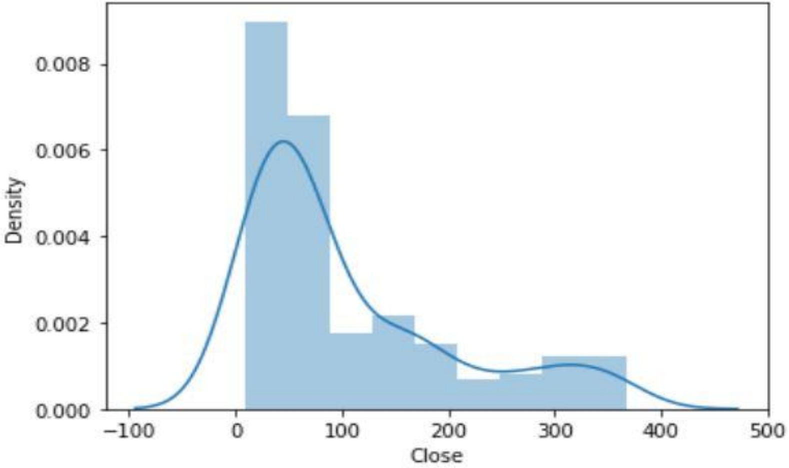




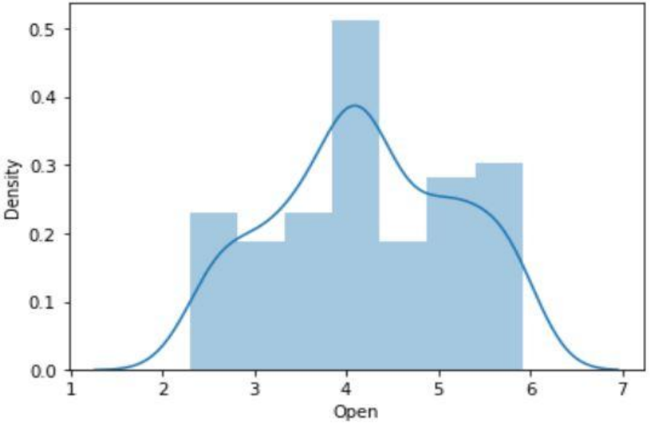
All the features are linearly correlated to each other . As we see the Best Fit Line.

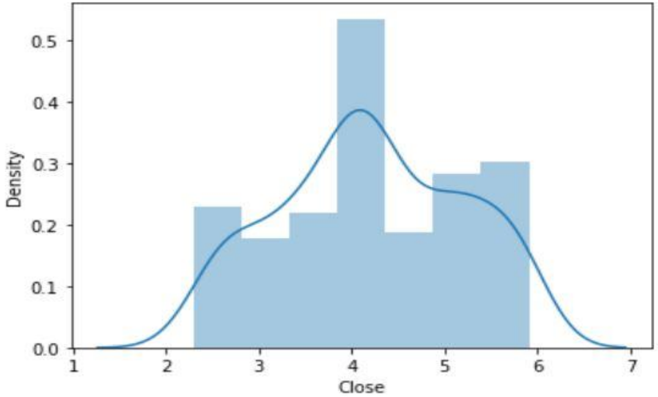
Comparing Data after and before normalising .



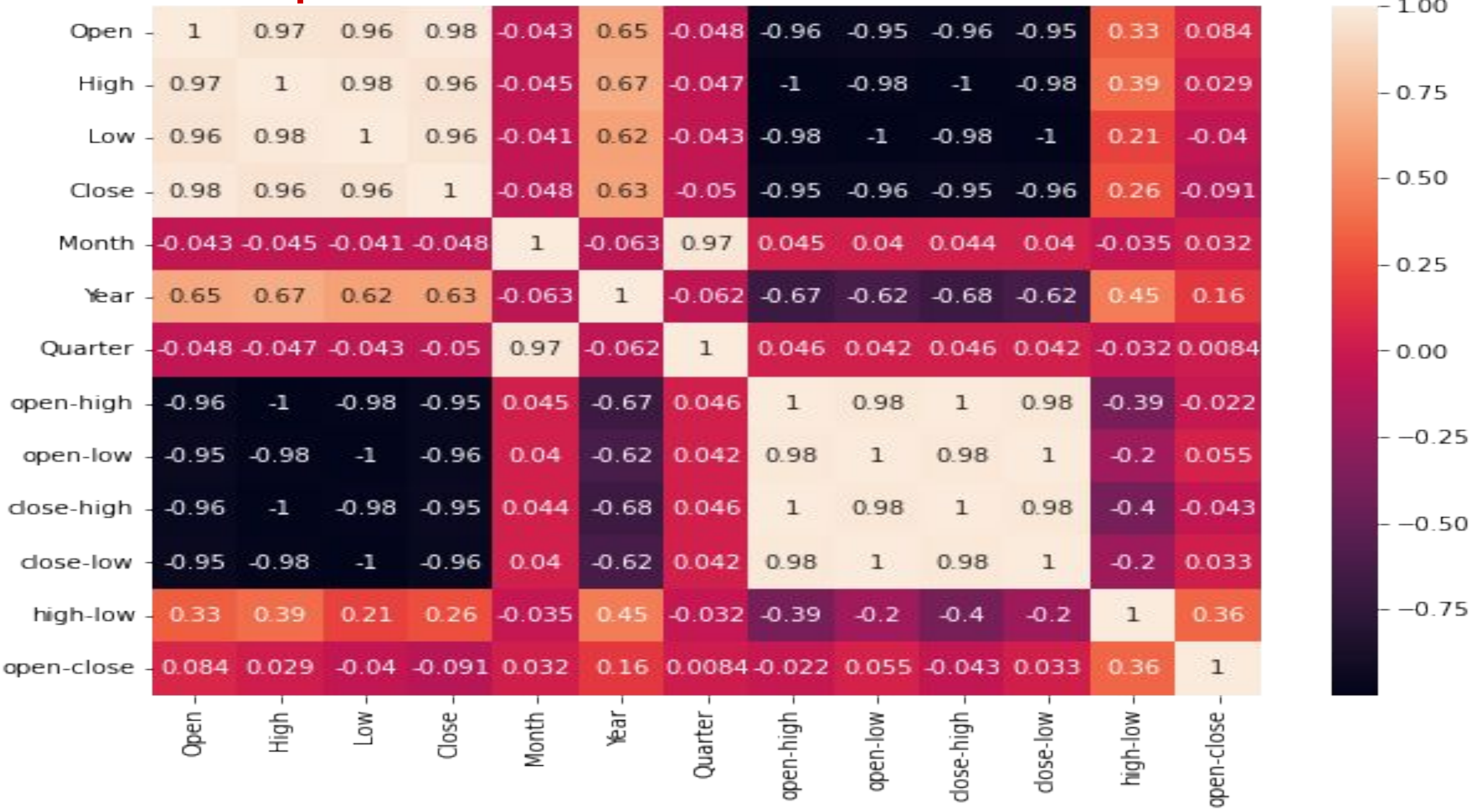


As we can see from above graph that the given Data is positively skewed . After normalising data we can see in below graph . For normalising data we have used boxcox function imported from scipy.stats .





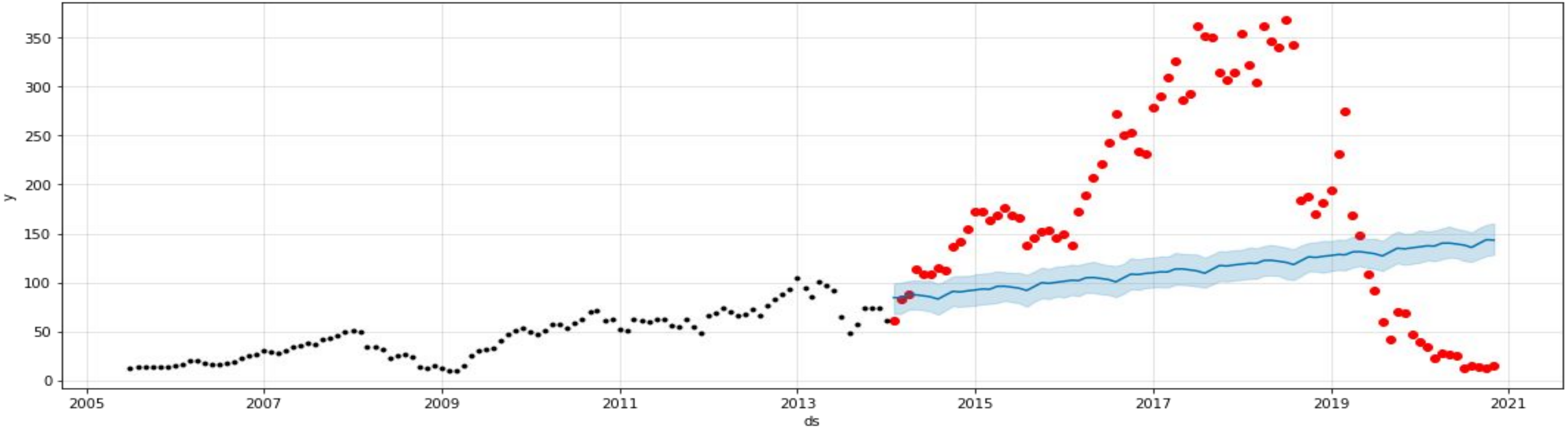
Correlation plot



FBProphet

- The Prophet library is an open-source library designed for making forecasts for time series datasets. It is easy to use and designed to automatically find a good set of hyperparameters for the model in an effort to make skillful forecasts for data with trends and seasonal structure by default.
- Prophet uses a decomposable time series model with three main model components: trend, seasonality, and holidays. They are combined in the following equation: **y(t) = g(t) + s(t) + h(t) + e(t)**
- g(t) is a trend function
- s(t) represents a periodic changes i.e weekly, monthly, yearly.
- h(t) is a function that represents the effect of holidays which occur on irregular schedules
- e(t) represents error changes that are not accommodated by the model.

FBProphet Models result



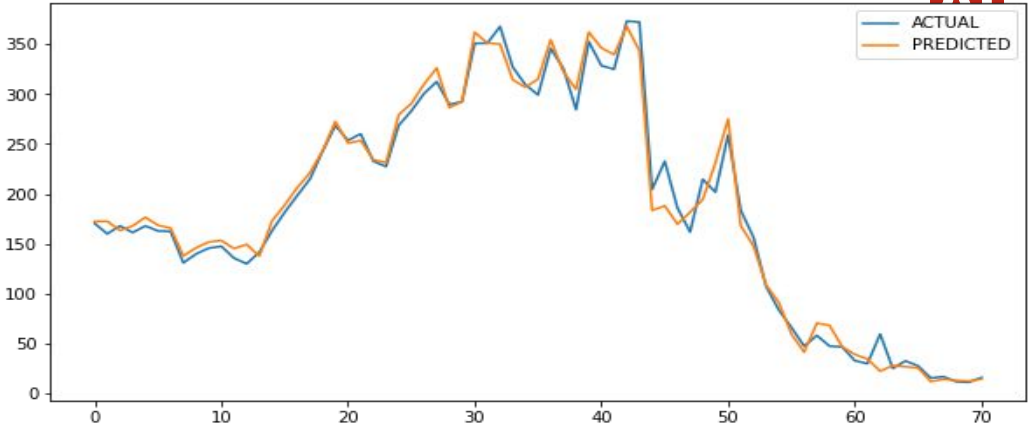
Prophet model does not work on the time series dataset, we can see in the above image where red dots are actual values where as blue line is predicted values.

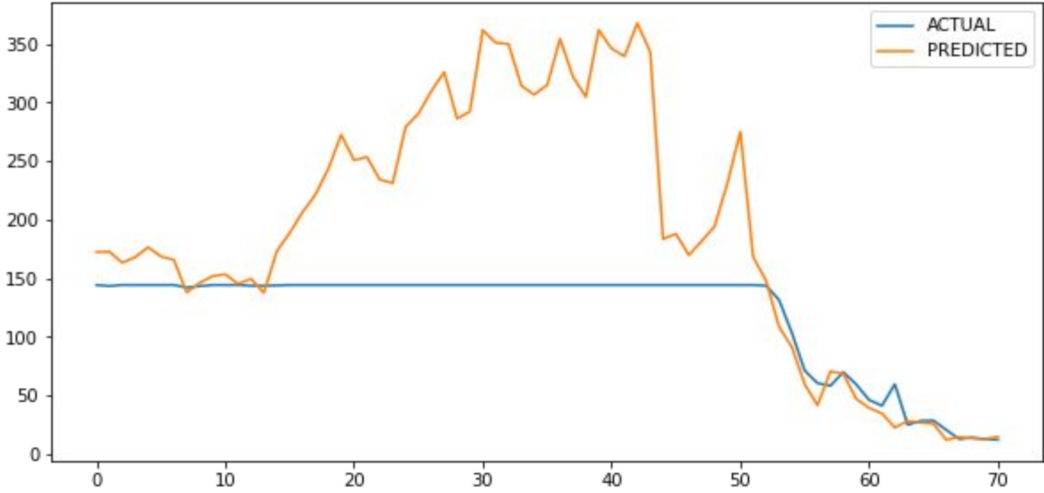
ML Regression models on Lag data set

After including lag columns we added original 3 columns in our dataset and performed multiple Regression model out of which Linear regression, Regularizing the linear models performed very well with r2score of 0.98 where as Random Forest didn’t perform well with r2score of 0.68.

Model Results

This graphs are of actual values vs. predicted values we can see that the linear model is performing very well on the test data set where as Random Forest is not able to predict the right values and have very high error





ML Models without using Lag data set

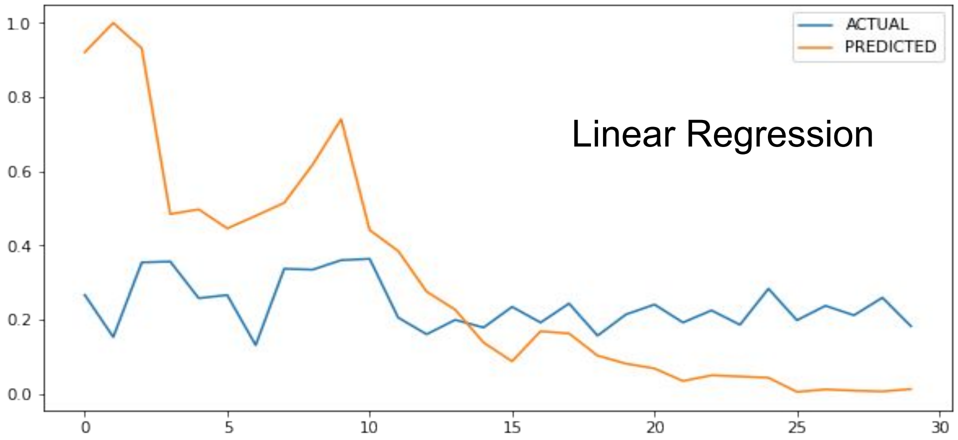
We have used 4 columns that have been created from the existing features. There are two methods used here to split the data into testing and training sets in order to find out which function gives us best results for our data.They are :

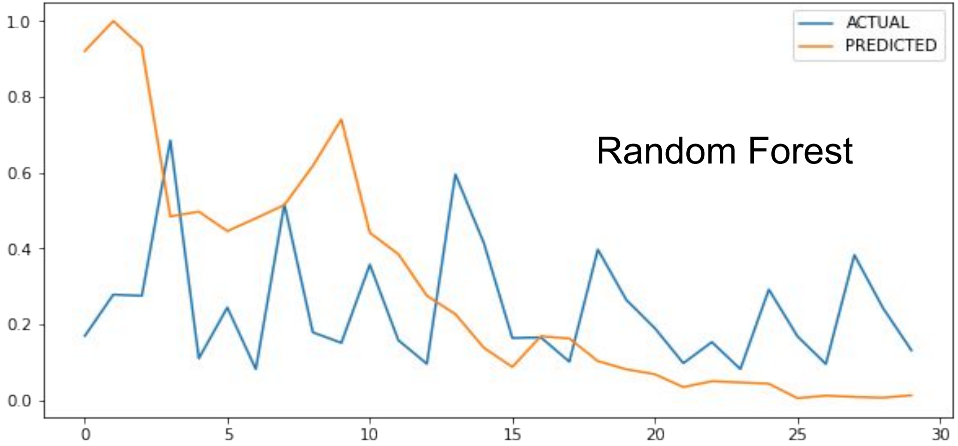
1. Split by using time series split
2. Split by using test_train_split

1.Split by time series function

The time series split function was used to split the data into training and testing sets. On this data two models, Linear Regression and Random Forest were used.Both the models did not perform well in this case. The models results were:

Model	R2 Score
Linear Regression	0.095
Random Forest	-0.186



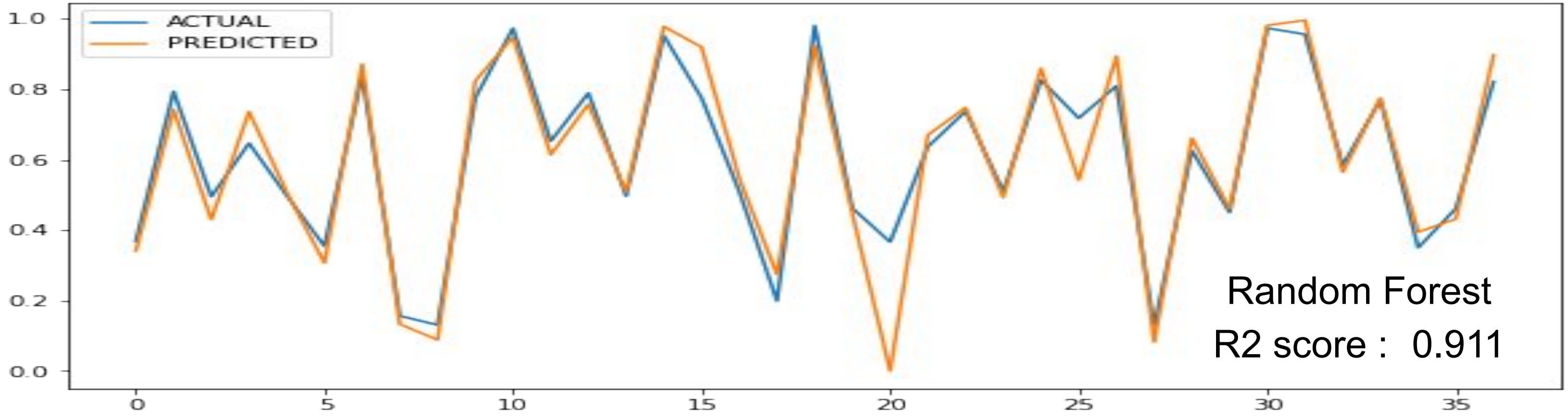


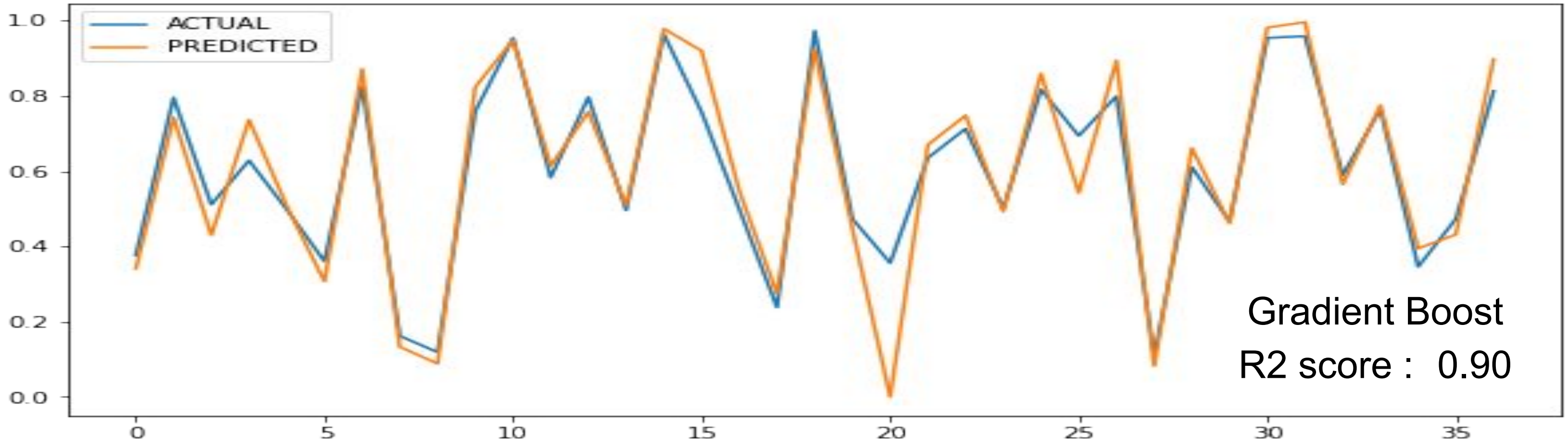
2.Split by test_train_split function

The training and testing sets contain 80% and 20% data respectively.We have used multiple models to fit the data out of which Random Forest and Gradient Boost have performed great with R2 score of 0.91 and 0.90 respectively. Here we can conclude that linear regression models could not perform better in case test_train_split.

Model	R2 Score
Random forest	0.911
Gradient Boost	0.900
XG Boost	0.901
KNN	0.785
SVM	0.687
Linear Regression	0.557
Ridge	0.546
Lasso	0.462

Model Results

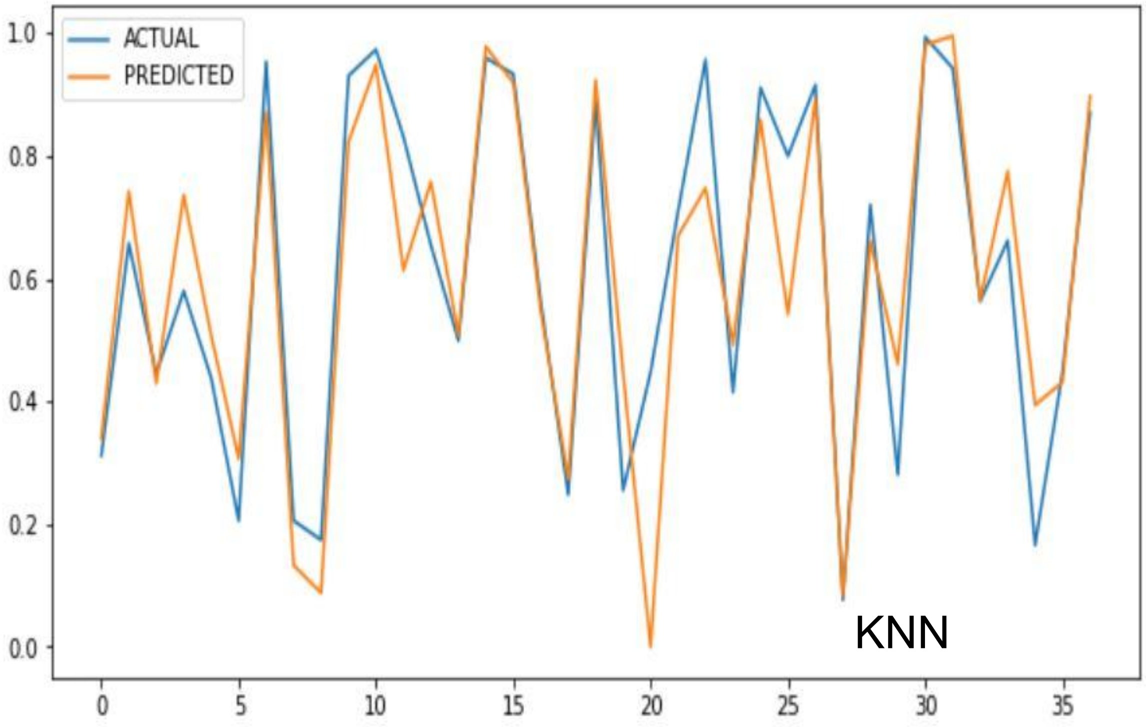




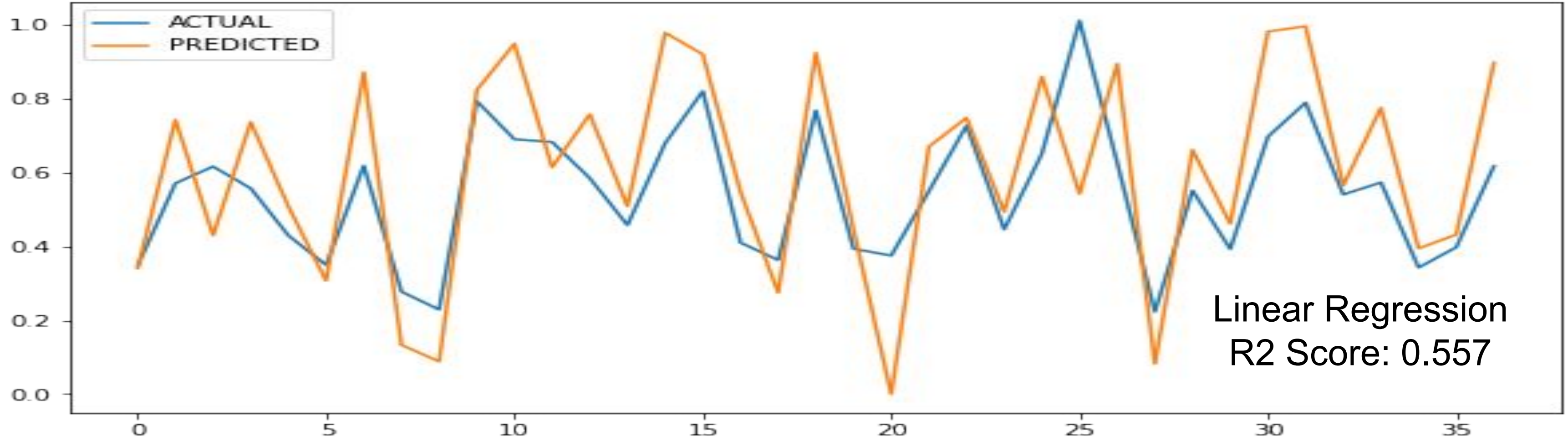
Model Results

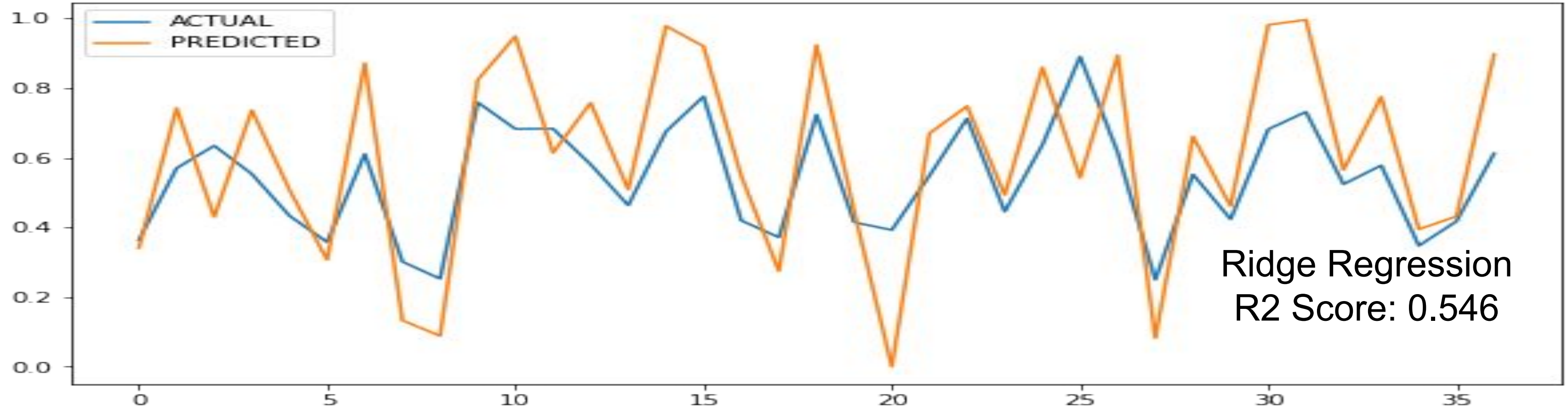
K Nearest Neighbours

As we can observe from the graph that KNN is working good in given data . As we can see in evaluated matrix that r2Score is good enough as well as adj_r2.

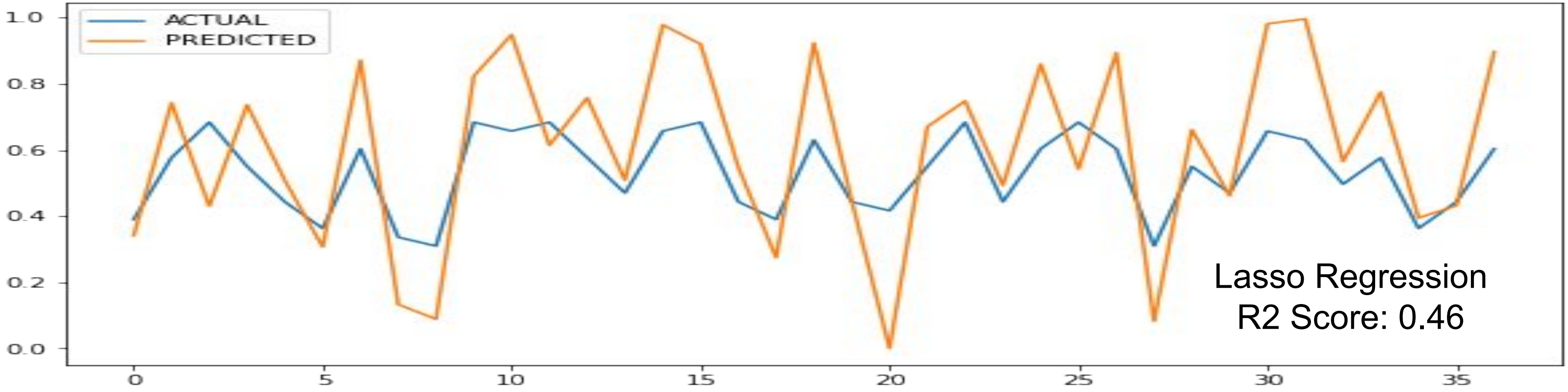


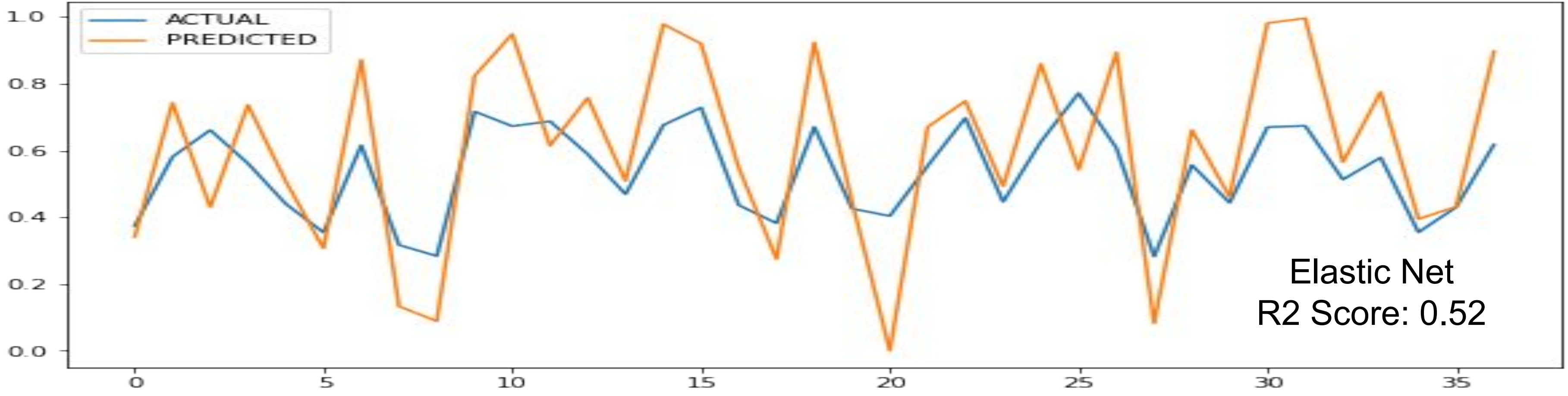
Model Results





Model Results





AUTO ARIMA

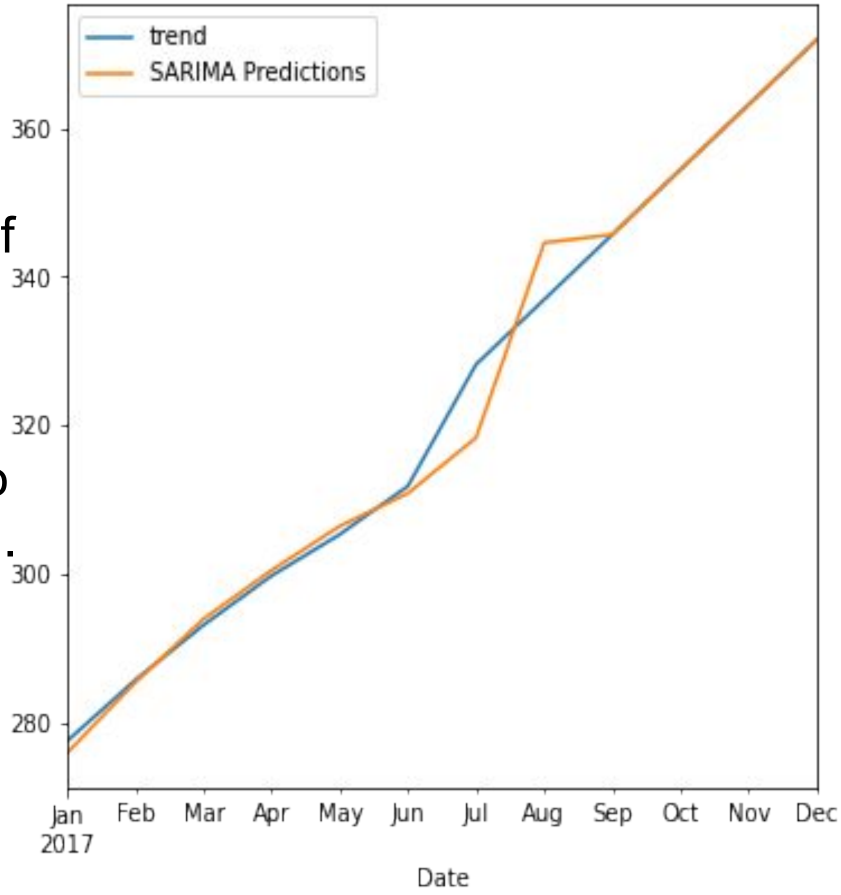
Auto ARIMA is like a grid search for time series models, it tries ARIMA, SARIMA, SARIMAX, all ARIMA related models depending on the parameters that are supplied to it. The auto_arima function seeks to identify the most optimal parameters for an ARIMA model, and returns a fitted ARIMA model. This function is based on the commonly-used R function, forecast::auto.arima

The auto_arima function works by conducting differencing tests (i.e., Kwiatkowski–Phillips–Schmidt–Shin, Augmented Dickey-Fuller or Phillips–Perron) to determine the order of differencing, d, and then fitting models within ranges of defined start_p, max_p, start_q, max_q ranges. If the seasonal optional is enabled, auto_arima also seeks to identify the optimal P and Q hyper- parameters after conducting the Canova-Hansen to determine the optimal order of seasonal differencing, D.

Here, the parameters which are supplied are: m= 12 indicating monthly range of Date seasonal True, and max iterations is set to 200 so that it analyses as many possible combinations of parameters before sticking to a local minima. Usually 200 works, However, higher the better, though that may take longer time.

AUTO ARIMA

It was achieved by plugging in time series data for the target variable. We have identified the appropriate number of lags or amount of differencing to be applied to the data and checked for stationarity. It then gave the output results, which are interpreted similarly to that of a multiple linear regression model. Auto ARIMA works after data from 2018-2020 was neglected for trial.
We can see in the image, the blue line is actual values whereas orange line is predicted values.



Conclusion

1. We have predicted the **stock prices** using different methods in this project.We have used simple machine learning linear regression, then have used FBProphet package and implemented linear regression, regularization techniques, bayesian ridge, xgboost, random forest and knn techniques.
2. We also have used two ways of splitting the data into train and test set one is train test split and other is **timeseriessplit**
3. Also we have implemented **Time series Analysis using ARIMA and SARIMA** (Checked **stationarity** of the model too).
4. Out of these we found **Linear regression, regularization technique, Random Forest and Auto Arima model** working very fine in predicting stock prices for the Yes Bank.

THANK YOU