Capstone Project 2
Yes Bank Stock Closing Price Prediction

Submitted by: Karan Tiwari

Content

Introduction

Yes Bank is a well-known bank in the Indian financial domain. It offers wide range of differentiated products for corporate and retail customers through retail banking and asset management services. According to the data shared by NPCI (National Payments Corporation of India), Yes Bank processed 25.94 million transactions amounting to INR 14811.73 crores through its own UPI app in July 2021. Since 2018, it has been in the news because of the fraud case involving Rana Kapoor.

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.

Data Description

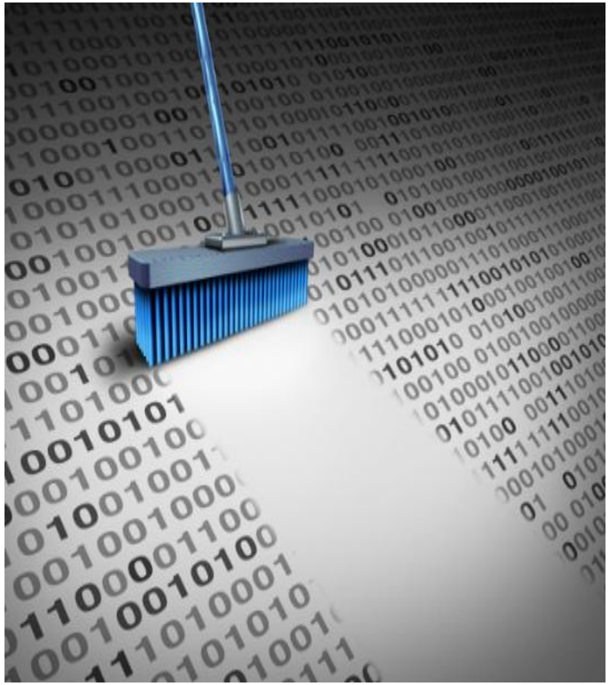
Dataset have 185 rows and 5 columns

- **Date_added: Date of record**
- **Open :Opening price of month**
- **High : Highest price of month**
- **Low : Lowest price of month**
- **Close : Closing Price of month**

AI

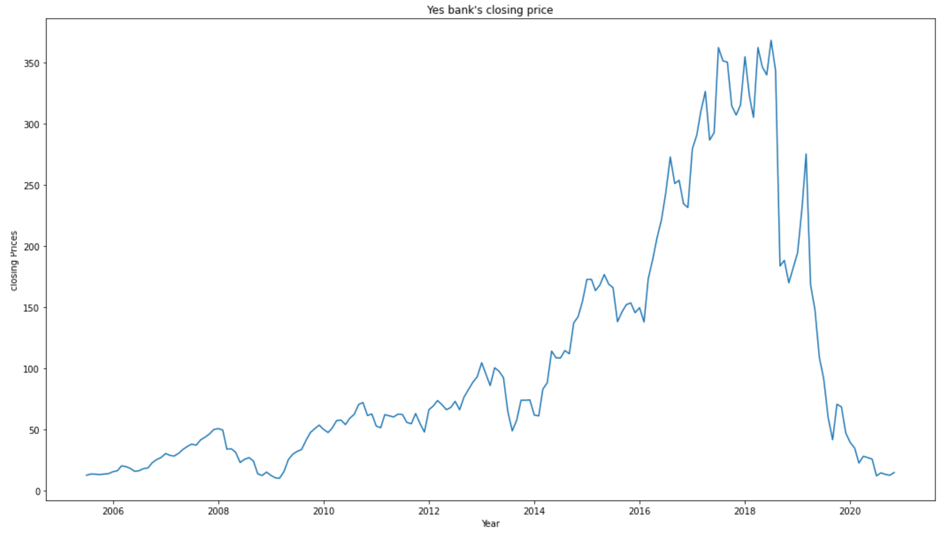
Data Cleaning

- Data set does not have null value.
- No duplicate rows
- We changed date format (from this Jul-05 to this 2005-07-01)



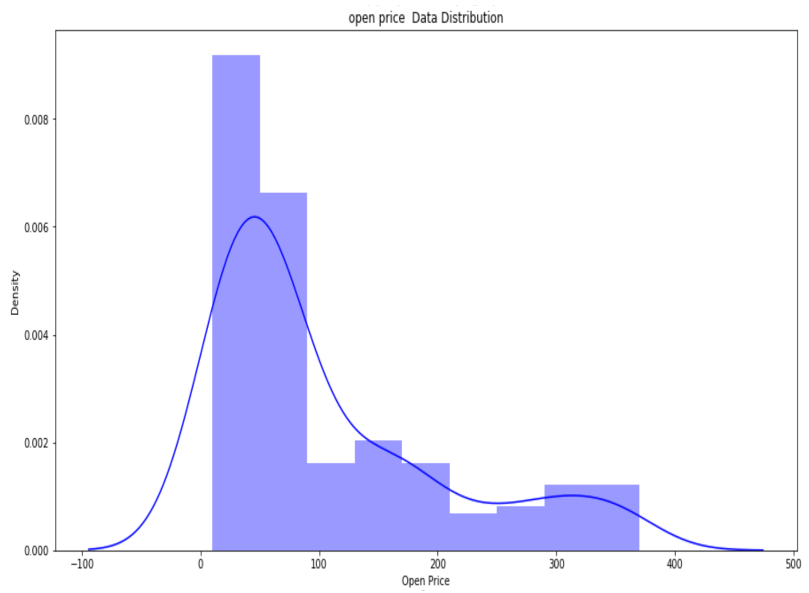
Exploratory Data Analysis

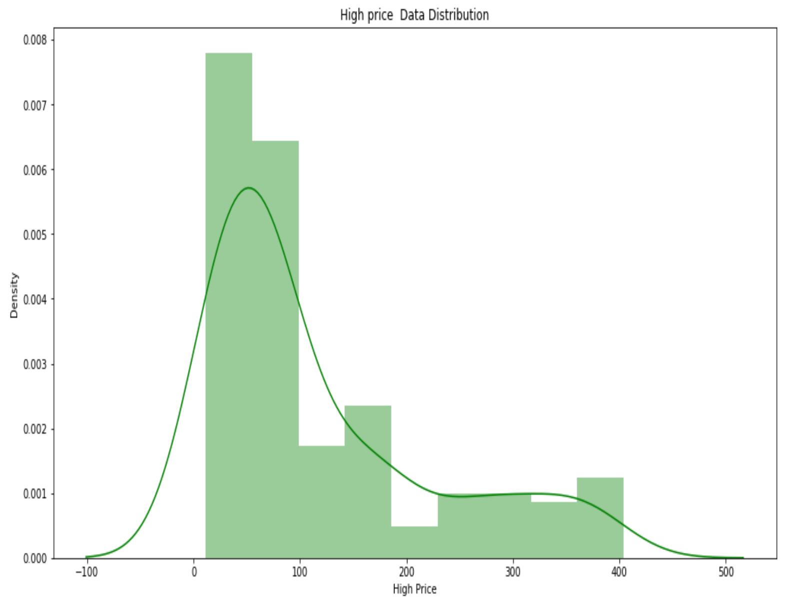
Yes Bank's closing price

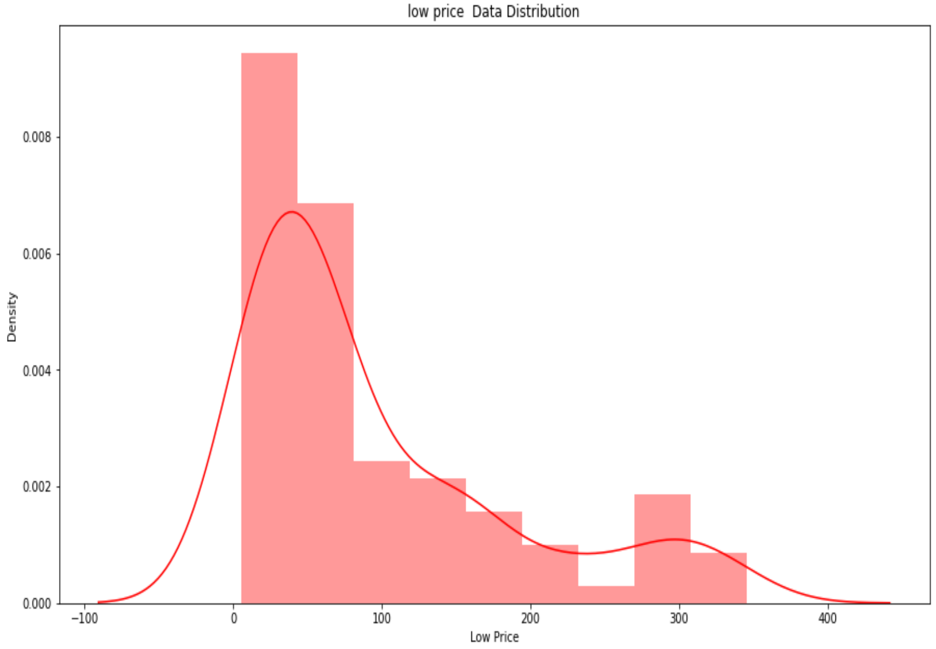


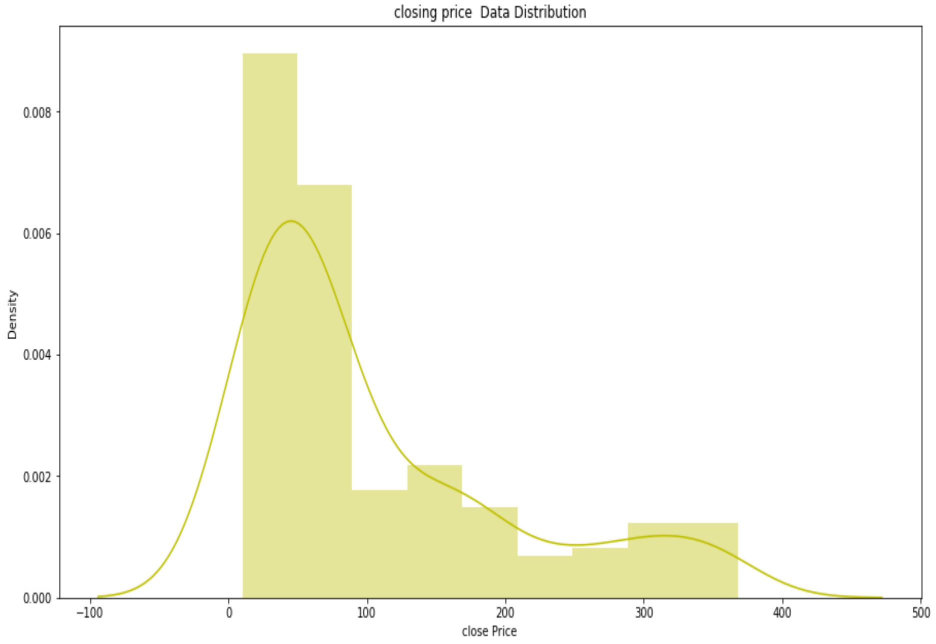
Here it's clear from the graph that stock was performing very good until 2018, but after 2018 there is a sudden fall we see in stock price of yes Bank. This might happen due to fraud case.

Data Distribution



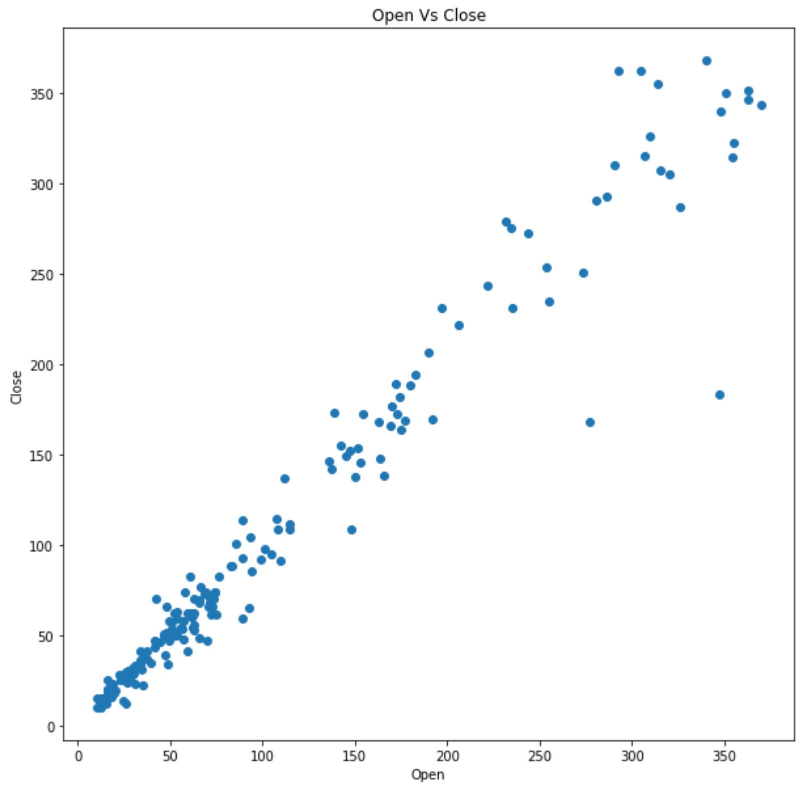


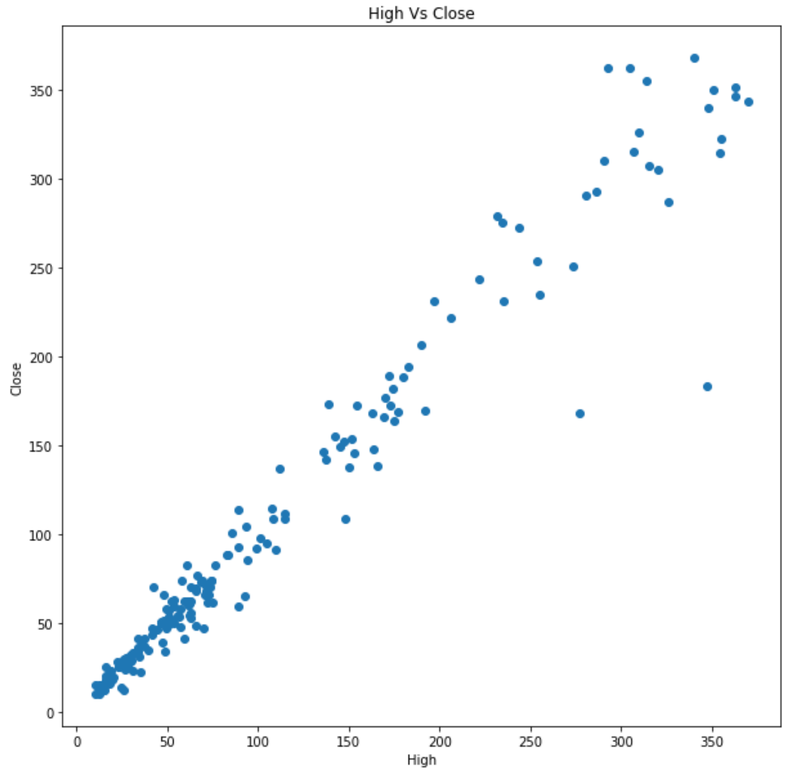


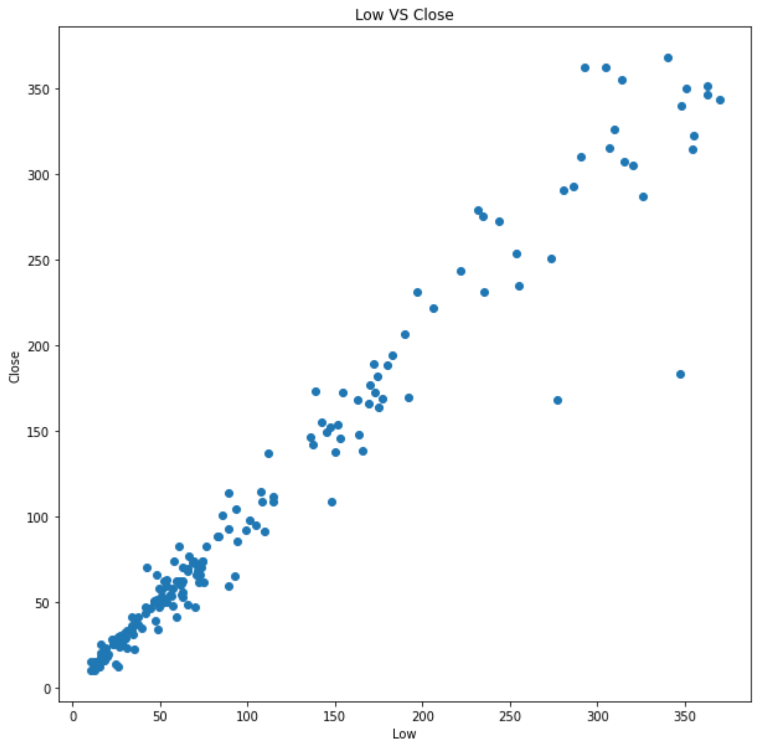


From the above four graph we can say that the data is rightly skewed and before applying Machine Learning model we have to change this to normal distribution.

Relation between dependent and independent variable.

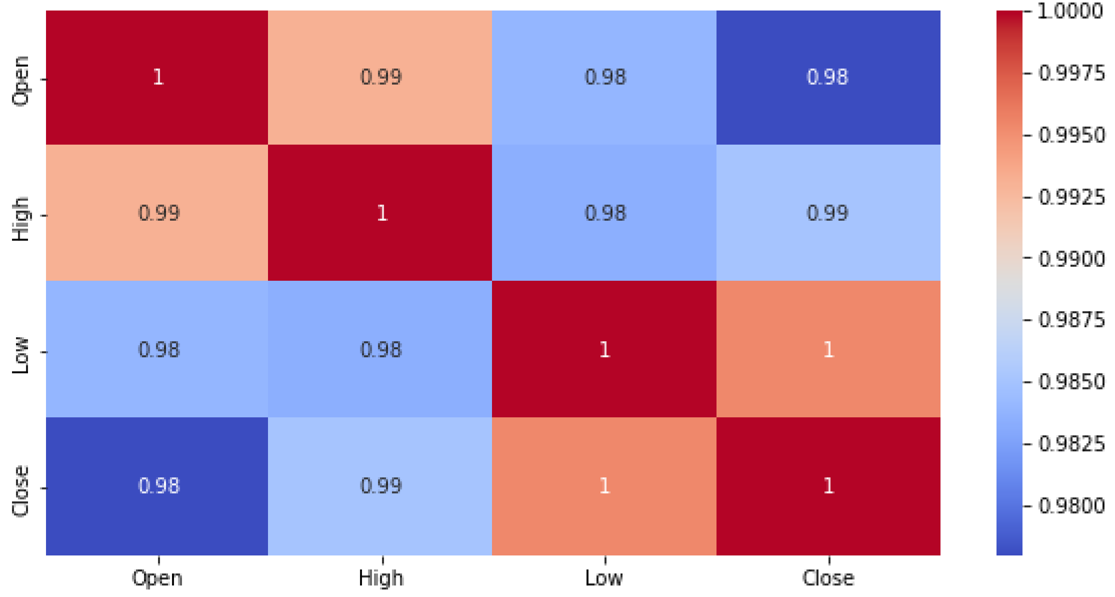






•In this independent variables are High,Low,Open features and dependent variable is close column.
•From the above graphs we can say that independent and depenent variable are linearly related to each other.

Correlation



- **The Heatmap is showing high correlation between the features.**
- **To handle multicollinearity we will use regularization technique(like L1 and L2)**

DATA MODELLING

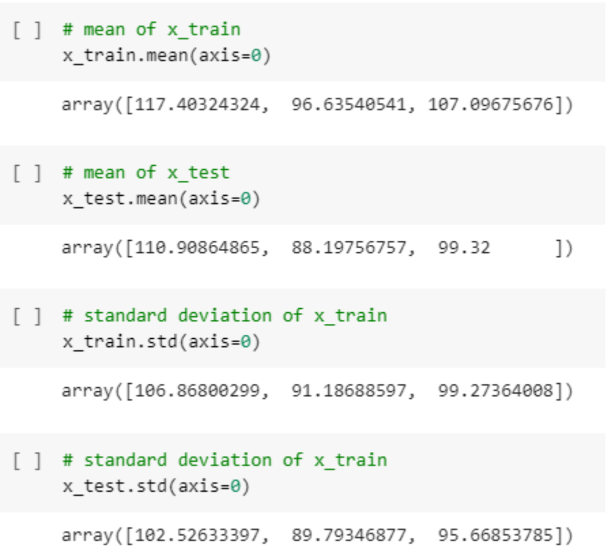
- First Assigning independent and dependent variable.

- Then do train test split in which 20% is test set and rest is for training set.

- x_train shape is 148 rows and 3 columns.

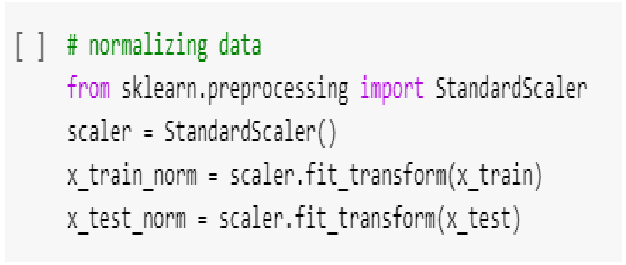
- x_test shape is 37 rows and 3 columns.

- Our Data was not normally distributed so now we will apply standard
 scaler to our x_train and x test.



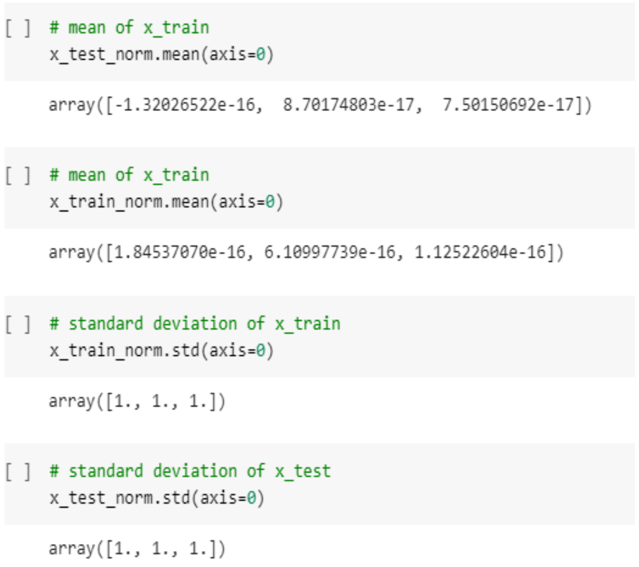
```
[ ] # mean of x_train
    x_train.mean(axis=0)

    array([117.40324324,  96.63540541, 107.09675676])

[ ] # mean of x_test
    x_test.mean(axis=0)

    array([110.90864865,  88.19756757,  99.32       ])

[ ] # standard deviation of x_train
    x_train.std(axis=0)

    array([106.86800299,  91.18688597,  99.27364008])

[ ] # standard deviation of x_train
    x_test.std(axis=0)

    array([102.52633397,  89.79346877,  95.66853785])
```

Mean and standard deviation of data before applying Standard scaler.



```
[ ] # normalizing data
    from sklearn.preprocessing import StandardScaler
    scaler = StandardScaler()
    x_train_norm = scaler.fit_transform(x_train)
    x_test_norm = scaler.fit_transform(x_test)
```

Applying standard scaler



```
[ ] # mean of x_train
    x_test_norm.mean(axis=0)

    array([-1.32026522e-16,  8.70174803e-17,  7.50150692e-17])

[ ] # mean of x_train
    x_train_norm.mean(axis=0)

    array([1.84537070e-16, 6.10997739e-16, 1.12522604e-16])

[ ] # standard deviation of x_train
    x_train_norm.std(axis=0)

    array([1., 1., 1.])

[ ] # standard deviation of x_test
    x_test_norm.std(axis=0)

    array([1., 1., 1.])
```

Mean and standard deviation of data after applying Standard scaler.

Implementing Machine learning Models

- Linear Regression
- Lasso Regression
- Ridge Regression
- Elastic Net Regression

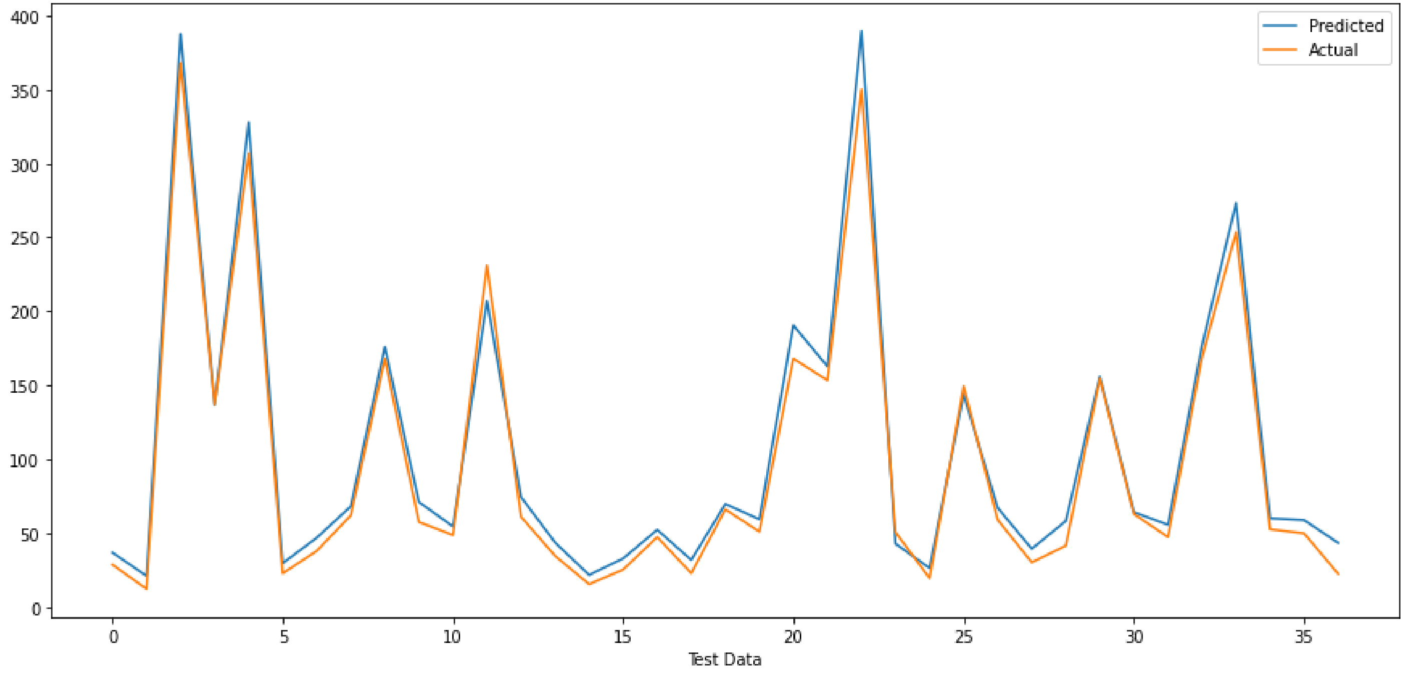
All these models are giving accuracy of 98 %

R squared

R-squared (R^2) is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model.

$$R^2 = 1 - \frac{RSS}{TSS}$$

R^2	= coefficient of determination
RSS	= sum of squares of residuals
TSS	= total sum of squares



•We plot the graph of actual value and predicted value and we got very good accuracy.

•Each models we used is giving a similar graph.

Conclusion

•This dataset has 185 rows and 5 columns.

• Histogram plot of all feaures are rightly skewed.

•The sudden crash we see in the graph after 2018 is might be due to the spread of Fraud case news among Stock Holder's which creates bearishness in the stock prices.

•In the Dataset Close Column is dependent Variable and Open, High, Low column's are independent features.

•Dependent and Independent features are linearly related with each other.

•The Heatmap is showing high correlation between the features.

• Then We implemented Linear regression ,Lasso regression,Ridge regression and Elasticnet regression and the accuracy score that we got is almost 98%,which shows that we achieve the almost best fit model for our dataset.