Capstone Project

Yes Bank Stock Closing Price Prediction

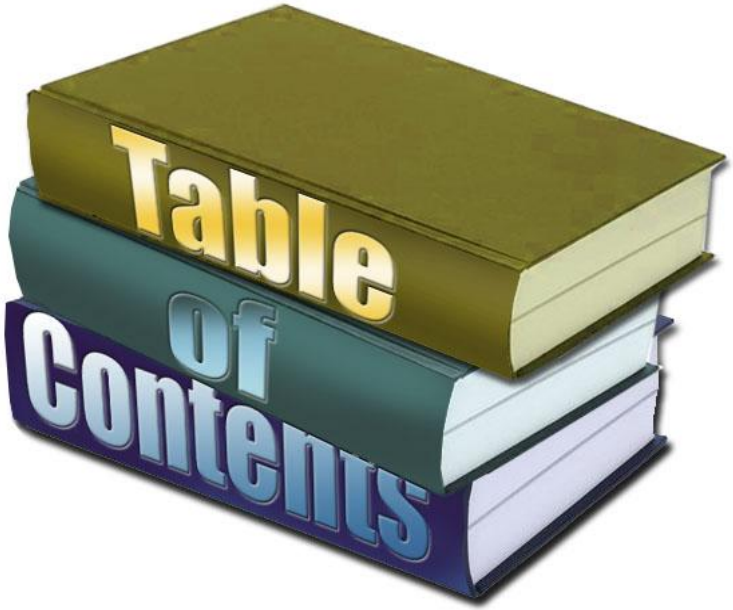
Team Members

i) Debaprasad Mohapatra

ii) Saurabh Yadav

Content

1. Problem Statement
2. Data Summary
3. Analysis of Data
4. Null value Imputation/ Data Cleaning
5. Data Preprocessing
6. Model Training
7. Forecasting
8. Evaluation Metrics
9. Challenges
10. Conclusion



Problem Statement:

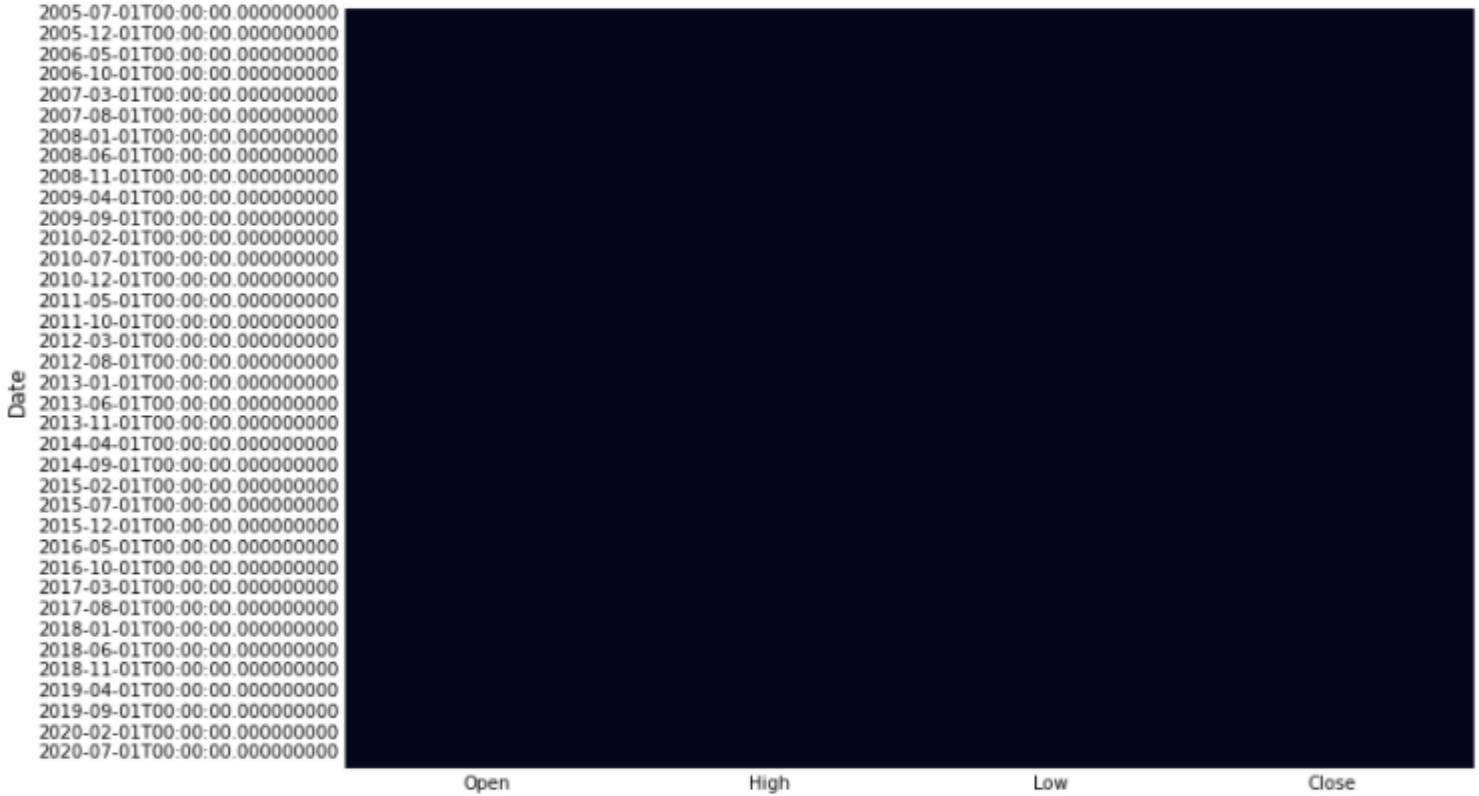
1. The objective of the project is to come up with the time series model or predictive model to predict the stock closing price of the month using the Yes Bank dataset of monthly stock prices of the month.

Data Summary:

We've YES BANK stock price dataset which has the monthly stock prices of the bank since it's inception.
It includes over **185** records and **5** attributes.
Important Attributes:

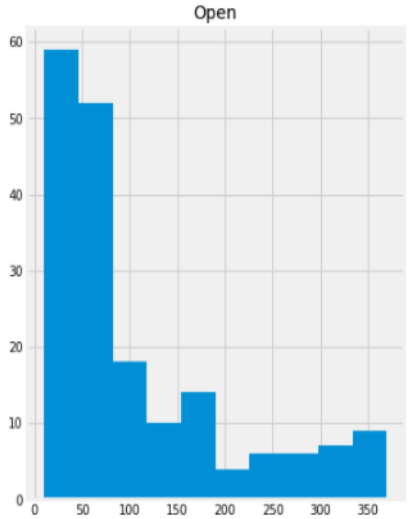
1. **Date:** It denotes date of investment done (in our case we have month and year).
2. **Open:** Open means the price at which a stock started trading when the opening bell rang.
3. **High:** High refer to the maximum prices in a given time period.
4. **Low:** Low refer to the minimum prices in a given time period.
5. **Close:** Close refers to the price of an individual stock when the stock exchange closed for the day.

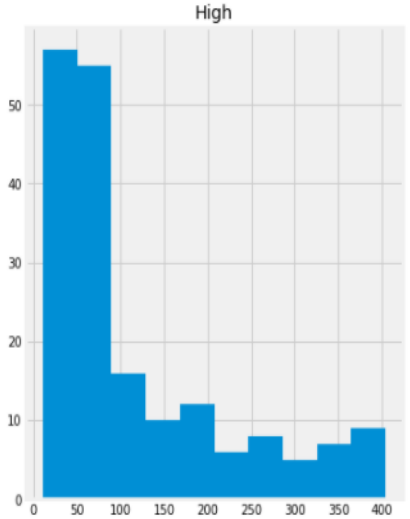
Where are the missing values in our data?

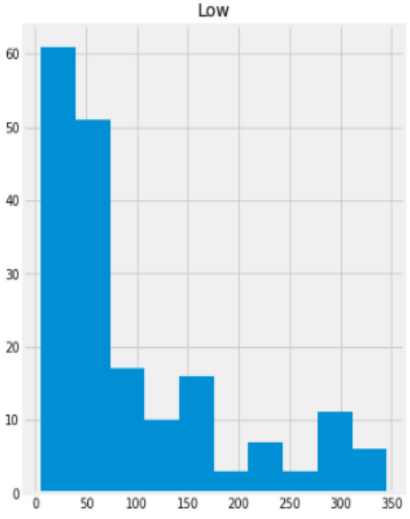


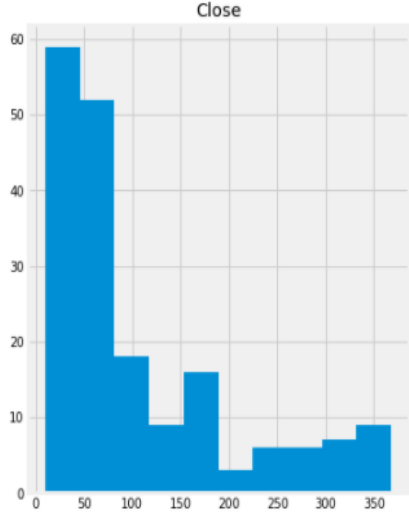
Visualizing the data

Distribution of the feature:

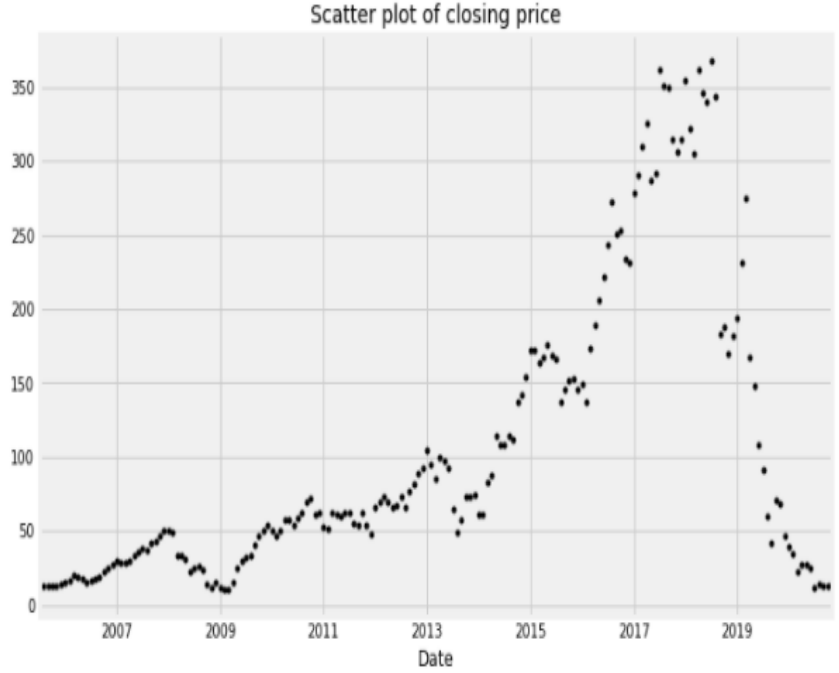




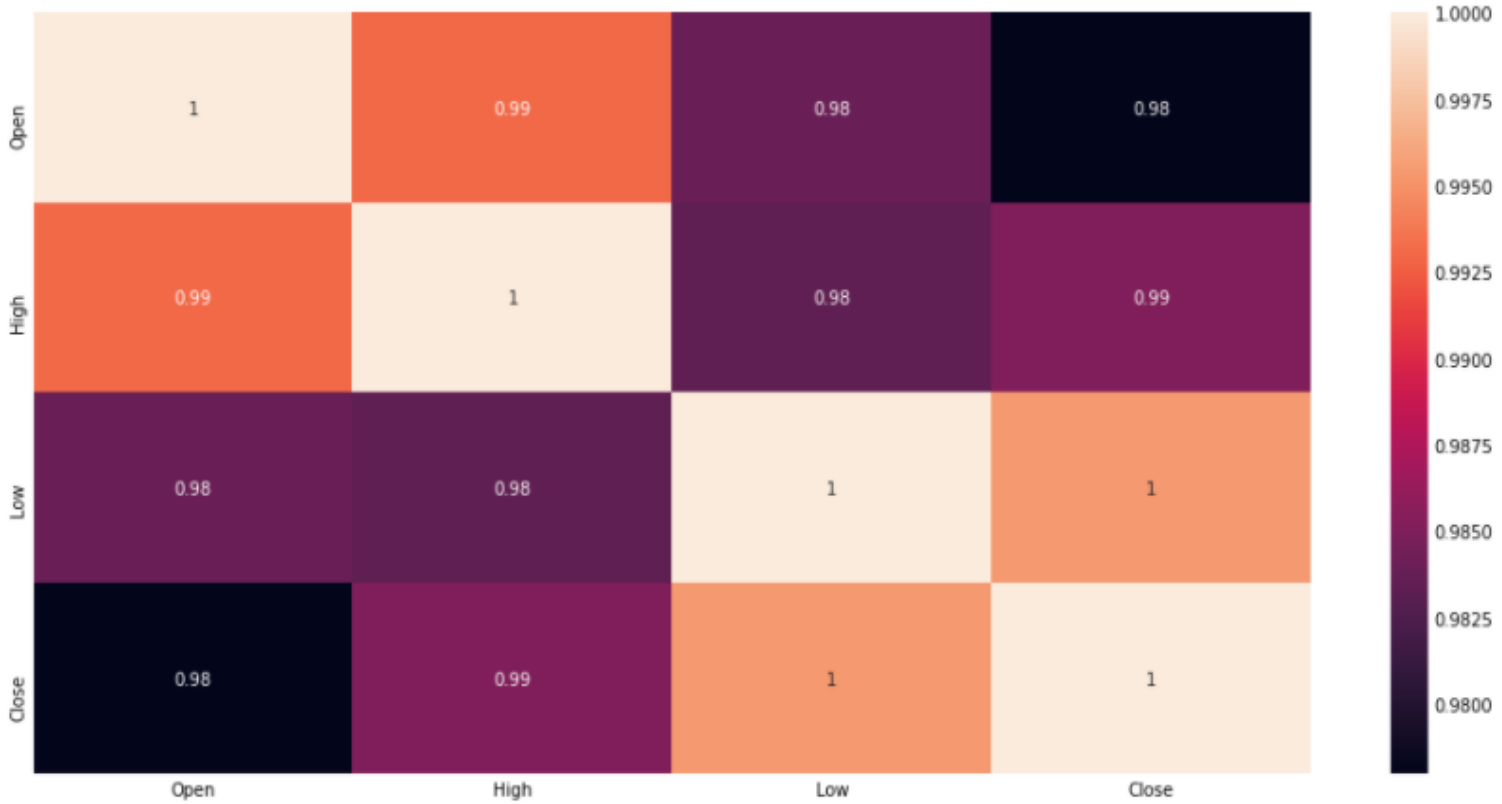




Analysis of target variable:



Correlation between the variables:



Data Preprocessing/ Feature Selection:

- The Given Date in data is of format MMM-YY is converted to proper date of YYYY-MM-DD. Since, ‘Date’ column has dtype as object we've converted it into date-time format. Then setting date column as an index.

	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

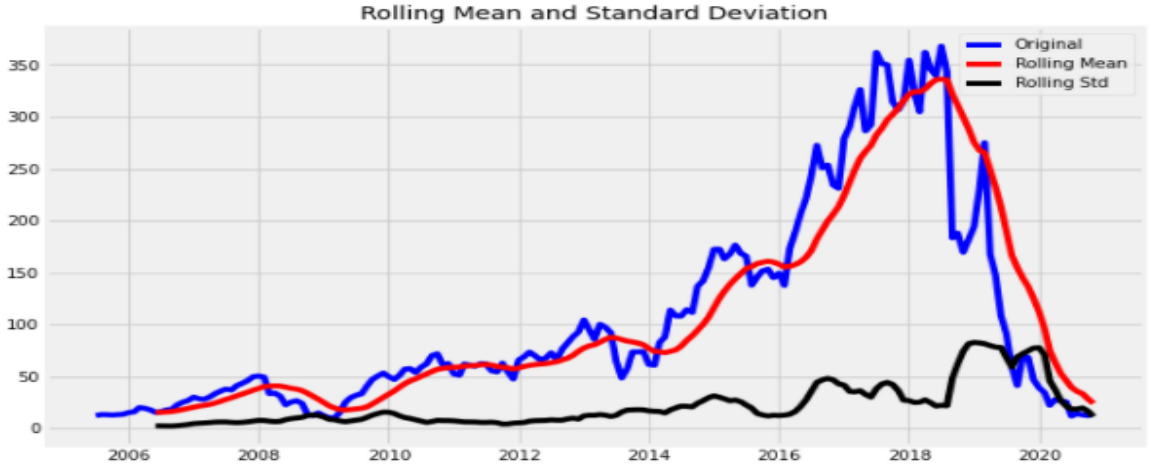
Converted

Date	Open	High	Low	Close
2005-07-01	13.00	14.00	11.25	12.46
2005-08-01	12.58	14.88	12.55	13.42
2005-09-01	13.48	14.87	12.27	13.30
2005-10-01	13.20	14.47	12.40	12.99
2005-11-01	13.35	13.88	12.88	13.41

- For time series forecasting we select only closing price feature of stock.
- Then we’ve checked series is stationary or not because time series analysis only works with stationary data.

Contd..

- To check series is stationary or not we've used Rolling window and ADF (Augmented Dickey-Fuller) Test.

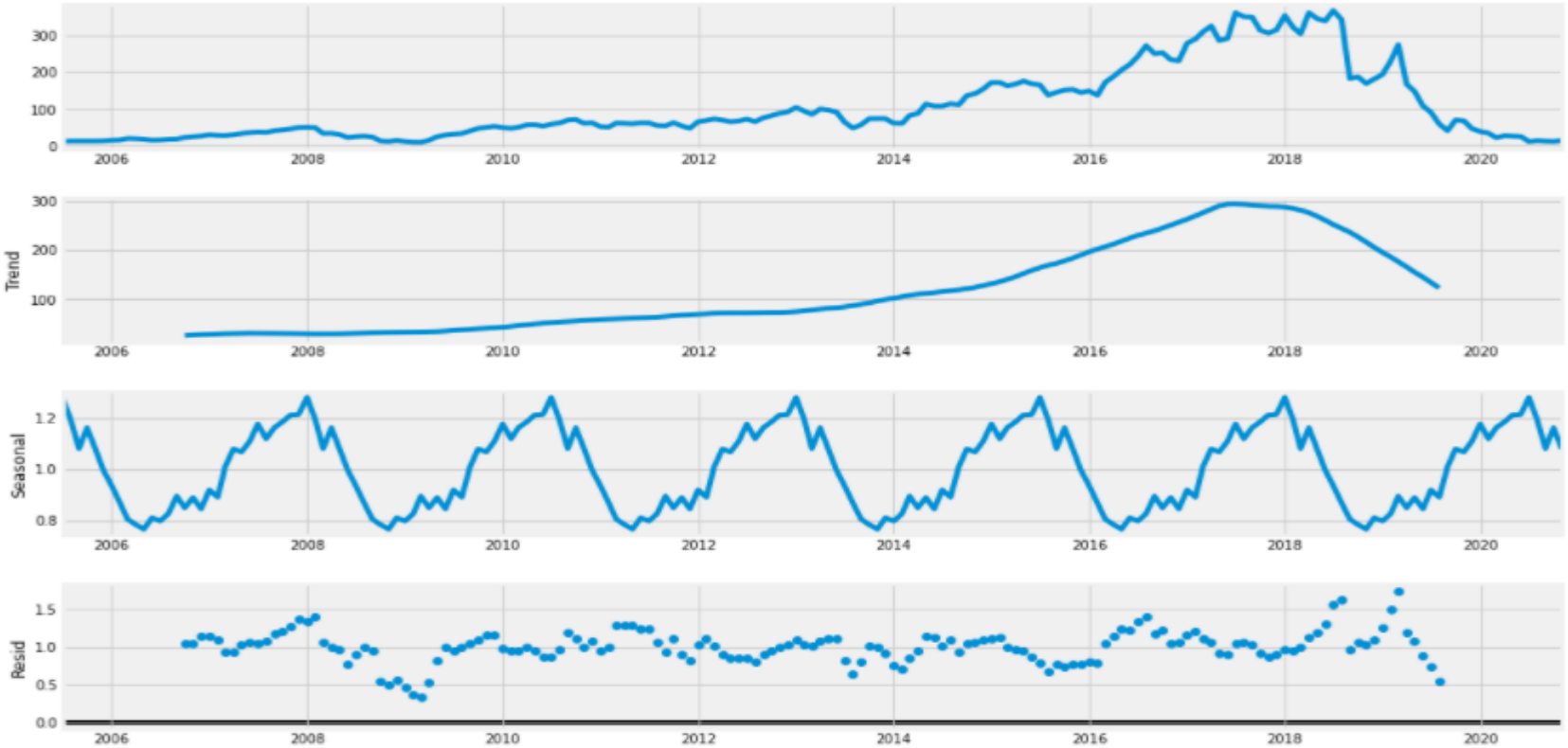



```
Results of dickey fuller test
Test Statistics                  -1.906409
p-value                           0.329052
No. of lags used                 14.000000
Number of observations used     170.000000
critical value (1%)              -3.469413
critical value (5%)              -2.878696
critical value (10%)             -2.575917
dtype: float64
weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary.
```

Contd..

- We see that the p-value is greater than 0.05 so we cannot reject the Null hypothesis. Also, the test statistics is greater than the critical values. so the data is non-stationary.
- In order to perform a time series analysis, we may need to separate seasonality and trend from our series. The resultant series will become stationary through this process.
- So let us separate Trend and Seasonality from the time series.

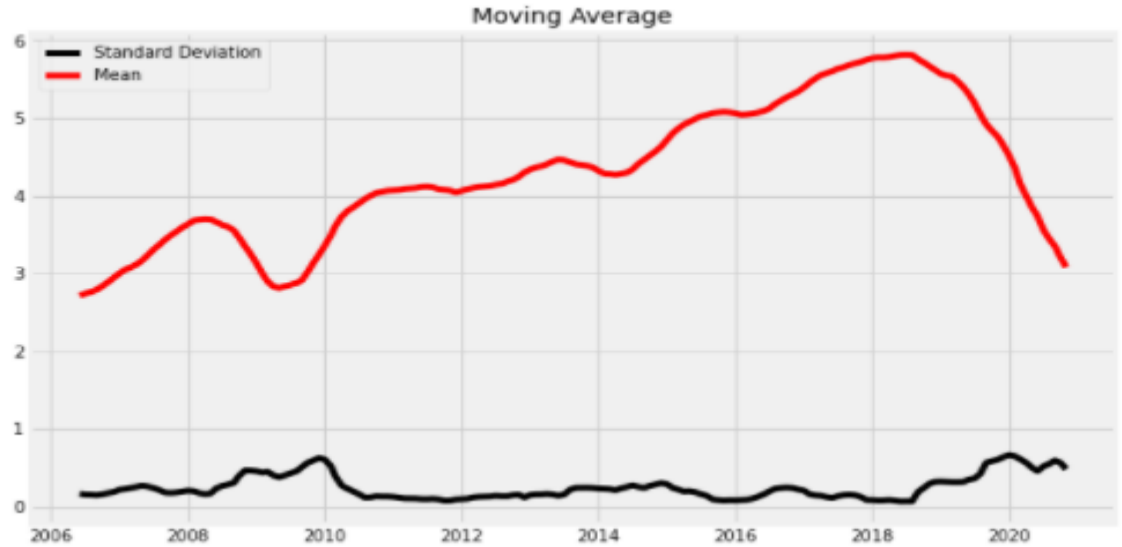
Contd..



Seasonal Decompose

Contd..

- Now, we start by taking a log of the series to reduce the magnitude of the values and reduce the rising trend in the series. Then after getting the log of the series, we find the rolling average of the series.
- A rolling average is calculated by taking input for the past 12 months and giving a mean consumption value at every point further ahead in series.



As we can see in plot the standard deviation follows the straight line.

Contd..

- For the **fbprophet** (Facebook's library for time series forecasting) the input to prophet is always a data-frame with two columns: **'ds' and 'y‘.**
- The **ds (datestamp)** column should be of a format expected by Pandas, ideally YYYY-MM-DD for a date or YYYY-MM-DD HH:MM:SS for a timestamp. The **y** column must be numeric, and represents the measurement we wish to forecast.

	Close
Date	
2005-07-01	12.46
2005-08-01	13.42
2005-09-01	13.30
2005-10-01	12.99
2005-11-01	13.41

	ds	y
0	2005-07-01	12.46
1	2005-08-01	13.42
2	2005-09-01	13.30
3	2005-10-01	12.99
4	2005-11-01	13.41

Model building, Predictions and Forecasting

Algorithms Used:

(Regression Approach)

- Linear Regression
- Random Forest
- XG-boost
- knn (K-Nearest Neigbors)

(Time series Approach)

- Auto-ARIMA
- Fbprophet

Combining all regression model

	Model_Name	MAE	MSE	RMSE	MAPE
0	LinearRegression	3.05	19.99	4.47	5.40
1	RandomForestRegressor	3.87	29.89	5.47	6.17
3	KNeighborsRegressor	4.70	50.70	7.12	6.45
2	XGBRegressor	4.30	33.32	5.77	6.86

- Observation from above table:
- **Linear Regression** gives lowest MAE, MSE, RMSE, MAPE.
- Here, we can say that Linear Regression is the best model that can be used for the stock price prediction.

Building ARIMA Model:

Split data into train and test :

Contd..

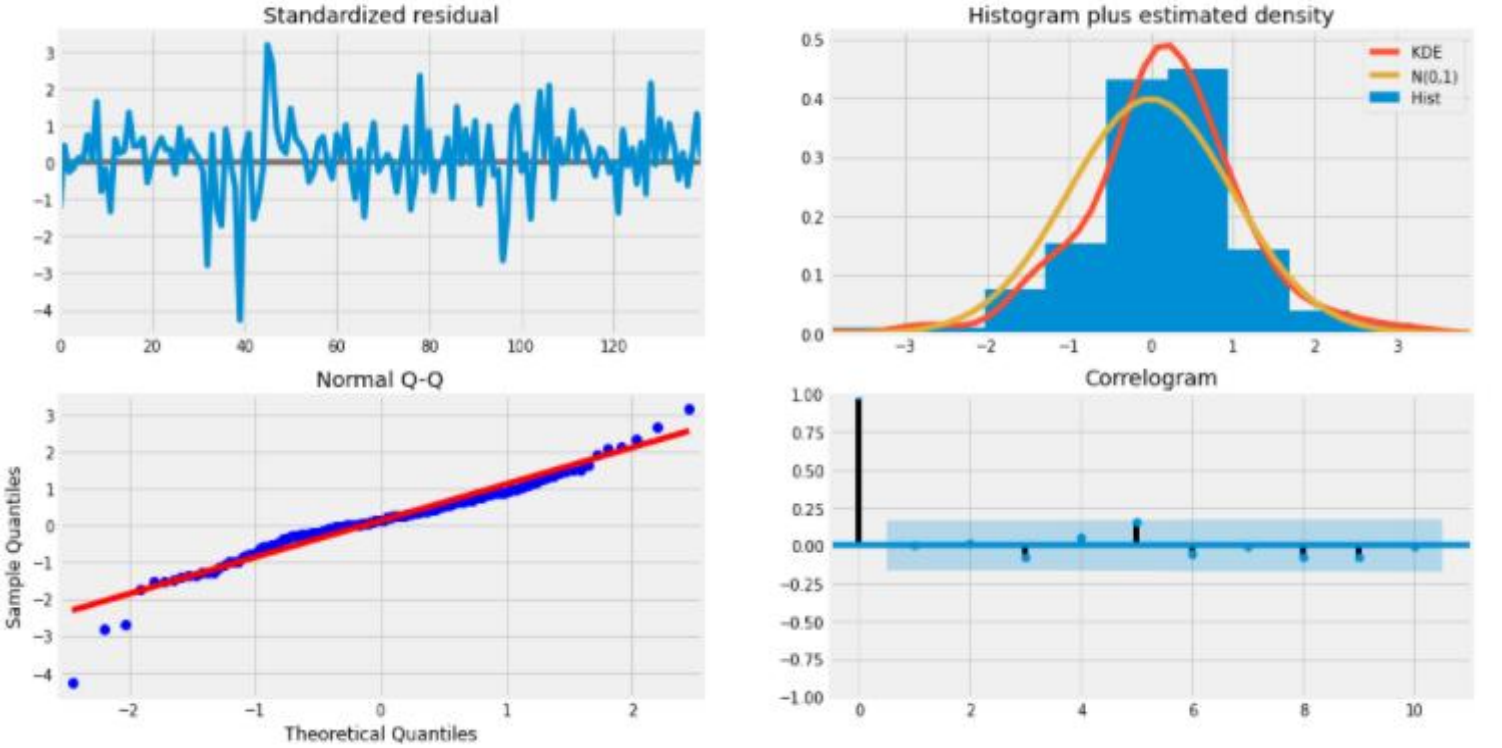
- Here, we use **Auto ARIMA** to get the best parameters without even plotting ACF and PACF graphs.

```
Best model:  ARIMA(2,0,1)(0,0,0)[0] intercept
Total fit time: 4.949 seconds
                               SARIMAX Results
==========================================================================================
Dep. Variable:                           y   No. Observations:                  140
Model:                 SARIMAX(2, 0, 1)   Log Likelihood                81.589
Date:                  Thu, 08 Jul 2021   AIC                           -153.179
Time:                          19:37:40   BIC                           -138.471
Sample:                               0   HQIC                          -147.202
                                  - 140
Covariance Type:                    opg
==========================================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------------------
intercept      0.0499      0.095      0.527      0.598      -0.136       0.236
ar.L1          0.4876      0.160      3.043      0.002       0.174       0.802
ar.L2          0.5001      0.161      3.110      0.002       0.185       0.815
ma.L1          0.7828      0.129      6.090      0.000       0.531       1.035
sigma2         0.0177      0.001     12.022      0.000       0.015       0.021
==========================================================================================
Ljung-Box (L1) (Q):                   0.01   Jarque-Bera (JB):                 60.14
Prob(Q):                              0.93   Prob(JB):                          0.00
Heteroskedasticity (H):               0.65   Skew:                             -0.57
Prob(H) (two-sided):                  0.14   Kurtosis:                          6.00
==========================================================================================
```

So the Auto ARIMA model provided the value of **p, d, and q as 2,0 and 1** respectively.

Contd..

- **Residual plots from Auto-ARIMA:**



Contd..

Here,

- **Top left:** The residual errors seem to fluctuate around a mean of zero and have a uniform variance.
- **Top Right:** The density plot suggest normal distribution with mean zero.
- **Bottom left:** All the dots should fall perfectly in line with the red line. Any significant deviations would imply the distribution is skewed.
- **Bottom Right:** The Correlogram, aka, ACF plot shows the residual errors are not auto-correlated.

Overall, it seems to be a good fit. Let’s start forecasting the stock prices.

Contd..

Now building ARIMA model with provided optimal parameters p(2), d(0) and q(1):

ARMA Model Results

Dep. Variable:	Close	No. Observations:	140
Model:	ARMA(2, 1)	Log Likelihood	81.589
Method:	css-mle	S.D. of innovations	0.133
Date:	Thu, 08 Jul 2021	AIC	-153.179
Time:	19:39:11	BIC	-138.471
Sample:	07-01-2005 - 02-01-2017	HQIC	-147.202

	coef	std err	z	P>|z|	[0.025	0.975]
const	4.0716	0.985	4.132	0.000	2.140	6.003
ar.L1.Close	0.4875	0.173	2.821	0.005	0.149	0.826
ar.L2.Close	0.5002	0.173	2.894	0.004	0.161	0.839
ma.L1.Close	0.7828	0.128	6.117	0.000	0.532	1.034

Roots

	Real	Imaginary	Modulus	Frequency
AR.1	1.0082	+0.0000j	1.0082	0.0000
AR.2	-1.9827	+0.0000j	1.9827	0.5000
MA.1	-1.2774	+0.0000j	1.2774	0.5000

Forecasting the auto ARIMA model:

As we can see, an **auto ARIMA** model uses past data to understand the pattern in the time series. Using these values, the model captured an decreasing trend in the series.

Evaluation Metrics of ARIMA model:

Metrics	Errors
MSE(Mean Square Error)	1.476
MAE(Mean Absolute Error)	0.8569
RMSE(Root Mean Square Error)	1.216
MAPE(Mean Absolute Percentage Error)	0.255

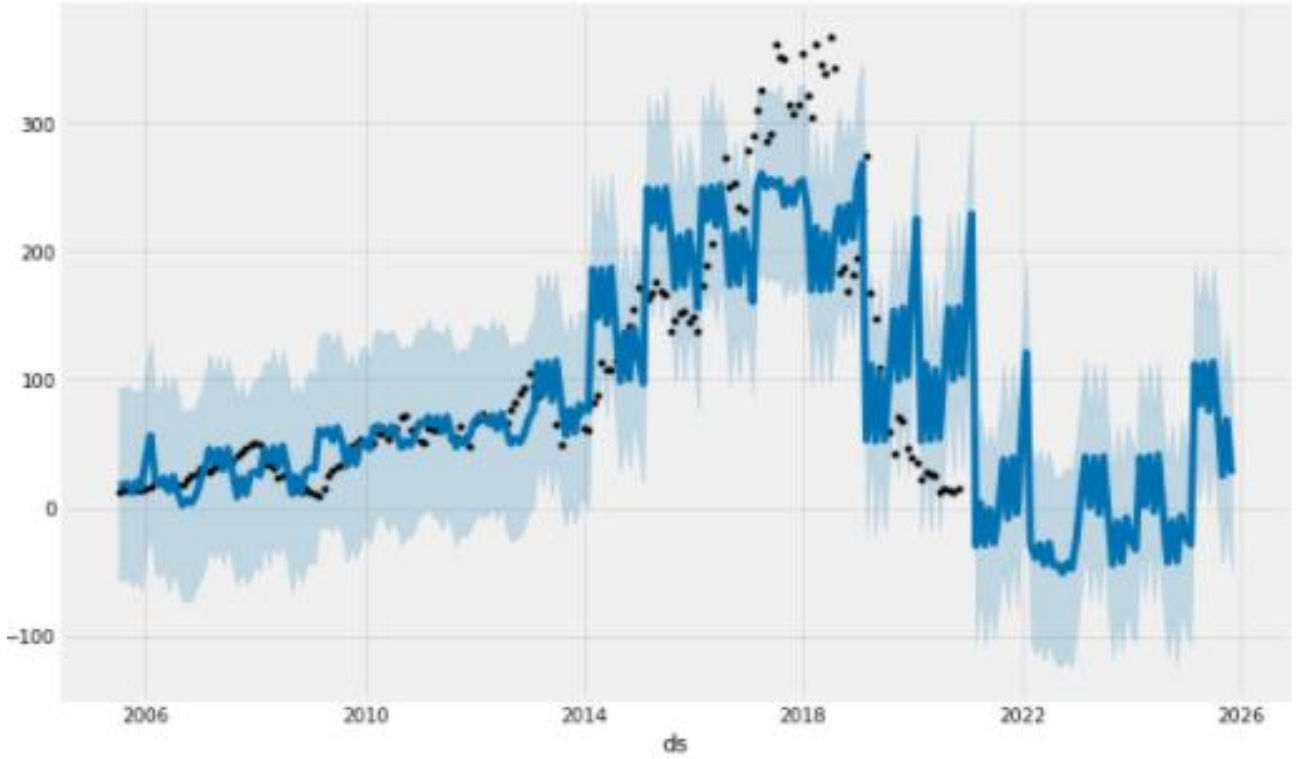
- Around 25.5% **MAPE** implies the model is about 74.5% accurate in predicting the test set observations.
- **Also** its evident from the plot, the model has captured a trend in the series, but does not focus on the seasonal part. In the next section, we will implement a time series model that takes both trend and seasonality of a series into account i.e. **fbprophet.**

Building Fbprophet Model:

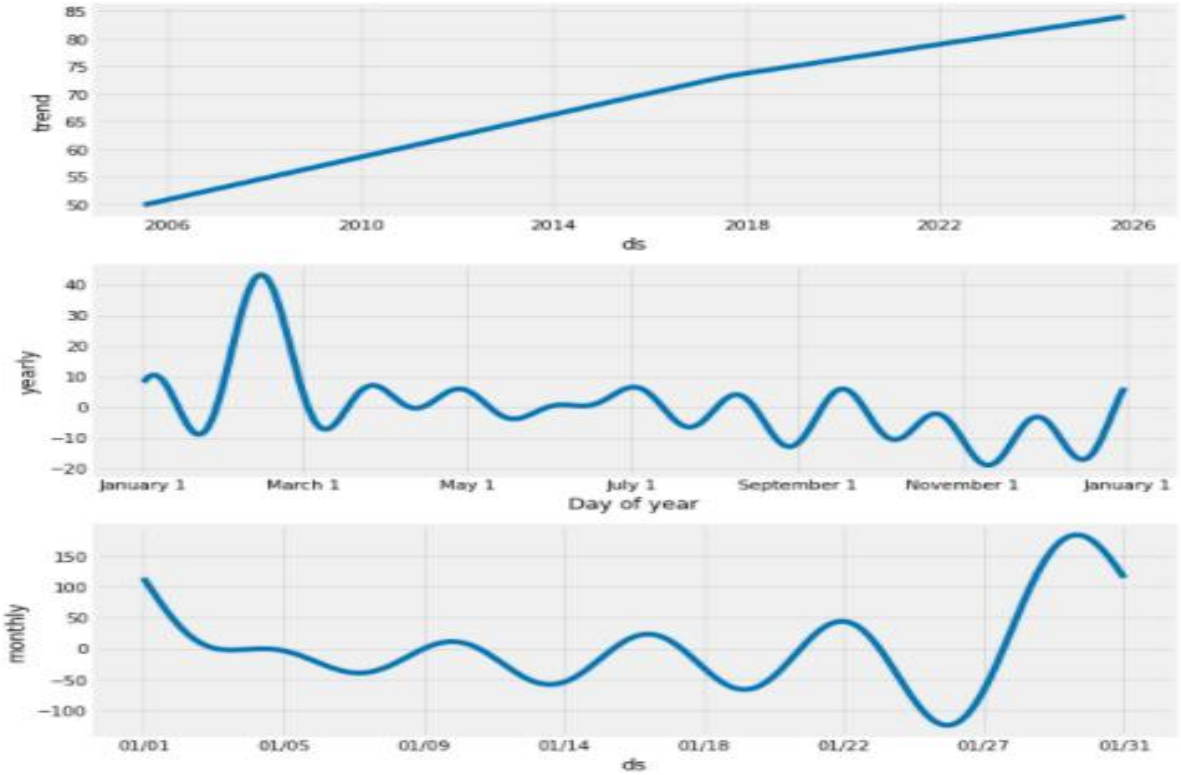
Components of our model:

- {'additive': ['monthly', 'yearly', 'additive_terms', 'extra_regressors_additive', 'holidays'],
- 'multiplicative': ['multiplicative_terms', 'extra_regressors_multiplicative']}

Forecasting model predictions



Plotting components of our model to see the trend, yearly seasonality, and weekly seasonality of the time series.



Trends:

- YES BANK's stock price is showing signs of upper trend yearly.

- YES BANK's stock price show upper trend signs during February and on December month it tend to give low stock price.

- YES BANK's stock price is showing signs of upper trend during end of the month.

Cross Validation for time series:

Prophet includes functionality for time series cross validation to measure forecast error using historical data. This is done by selecting cutoff points in the history, and for each of them fitting the model using data only up to that cutoff point. We can then compare the forecasted values to the actual values.

Parameters used:(model, horizon="365 days", period='180 days', initial='1095 days')

Where,

- **model**: Prophet class object. Fitted Prophet model.
- **horizon**: string with pd.Timedelta compatible style, e.g., '5 days', '3 hours', '10 seconds'.
- **period(spacing between cutoff dates)** : string with pd.Timedelta compatible style. Simulated forecast will be done at every this period. If not provided, 0.5 * horizon is used.
- **initial(the size of the initial training period)** : string with pd.Timedelta compatible style. The first training period will include at least this much data. If not provided, 3 * horizon is used.

Contd..

- Here, the output of **cross_validation** is a dataframe with the **true values y** and the out-of-sample **forecast values yhat**, at each simulated forecast date and for each cutoff date.

	ds	yhat	yhat_lower	yhat_upper	y	cutoff
0	2008-08-01	40.245191	38.961651	41.462829	26.83	2008-07-02
1	2008-09-01	41.067000	39.885731	42.314863	24.13	2008-07-02
2	2008-10-01	40.789432	39.487402	42.025925	13.58	2008-07-02
3	2008-11-01	42.973994	41.641828	44.234486	12.26	2008-07-02
4	2008-12-01	40.762470	39.520050	42.014857	15.03	2008-07-02

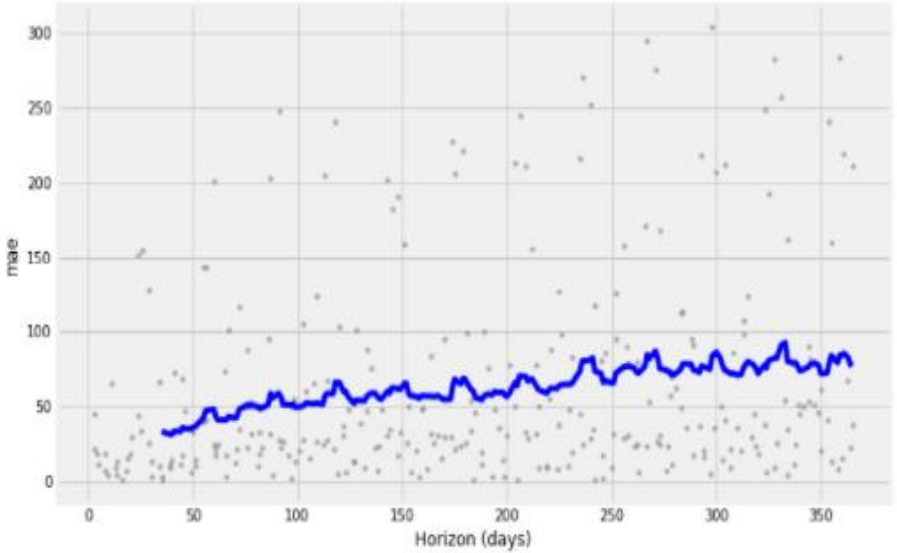
Evaluation Metrics:

- Here, we use the **performance_metrics** utility to compute the Mean Squared Error(MSE), Root Mean Squared Error(RMSE),Mean Absolute Error(MAE), Mean Absolute Percentage Error(MAPE) and the coverage of the the yhat_lower and yhat_upper estimates.

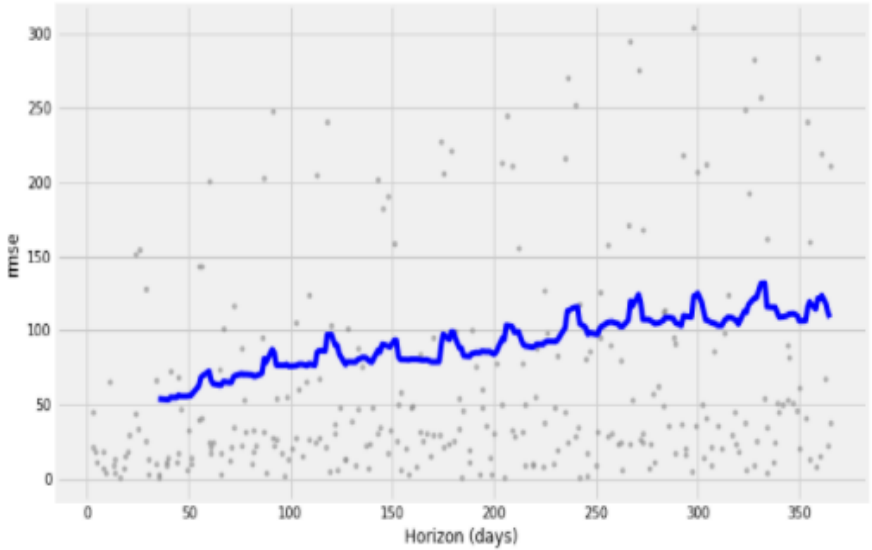
	horizon	mse	rmse	mae	mape	mdape	coverage
0	35 days	2909.646693	53.941141	32.995170	0.445647	0.209830	0.464286
1	39 days	2827.846854	53.177503	31.325250	0.381288	0.154778	0.500000
2	40 days	2823.260689	53.134364	31.181460	0.369197	0.154778	0.500000
3	41 days	3005.183478	54.819554	33.352355	0.377365	0.183549	0.464286
4	44 days	3002.675408	54.796673	33.301062	0.380515	0.207394	0.428571

Plotting MAE, RMSE and MAPE

Mean Absolute Error

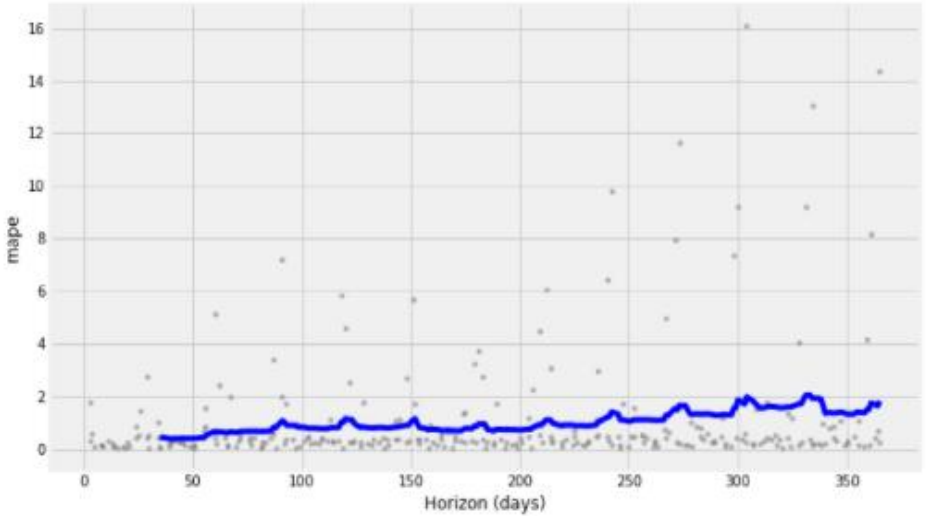


Root Mean Square Error



Contd..

Mean Absolute Percentage Error



- **Dots show** the absolute percent error for each prediction in data-frame.
- The **blue line** shows the MAPE, where the mean is taken over a rolling window of the dots.

Challenges

- Because of small dataset it’s difficult to get good model accuracy.
- Making the data stationary for time series model.
- Adding monthly trend to fbprophet.

Conclusion

- There is increase in trend of Yes Bank stock price till **2018** an then sudden decrease.
- Using regression approach we get the best performing model as Linear Regression model with accuracy of 94.60% which is excellent.
- On the other hand using time series approach for the FBprophet forecast the error of **3%** are typical for predictions one month into the future, and that errors increase up to around **18-19%** for predictions that are a year out.
- Fbprophet is the best performing time series model in terms of accuracy.
- February is the months where there is more upward trends for the stock.
- July also shows the upward trend but not as much February.
- November and December are months with the most downward pressure.
- On yearly basis there is decrease in trend from 2021 to 2024 then increase in trend from 2024 onwards.

Contributors Role

Debaprasad Mohapatra:

- Performed **EDA** (Exploratory Data Analysis)
- Build Regression Models such as, **Linear Regression, Random Forest, XG-boost, knn(K-Nearest neighbors).**

Saurabh Yadav:

- Performed Data preprocessing
- Build Time Series Models such as **Auto-ARIMA** and **Fbprophet**.