**STOCK PRICE PREDICTION**

**Team Members:**

Aditya Pawar ([pawar.ad@husky.neu.edu](mailto:pawar.ad@husky.neu.edu)) Dhairya Jaiswal ([jaiswal.d@husky.neu.edu](mailto:jaiswal.d@husky.neu.edu)) Ranga Chari ([vinjamuri.r@husky.neu.edu](mailto:vinjamuri.r@husky.neu.edu))

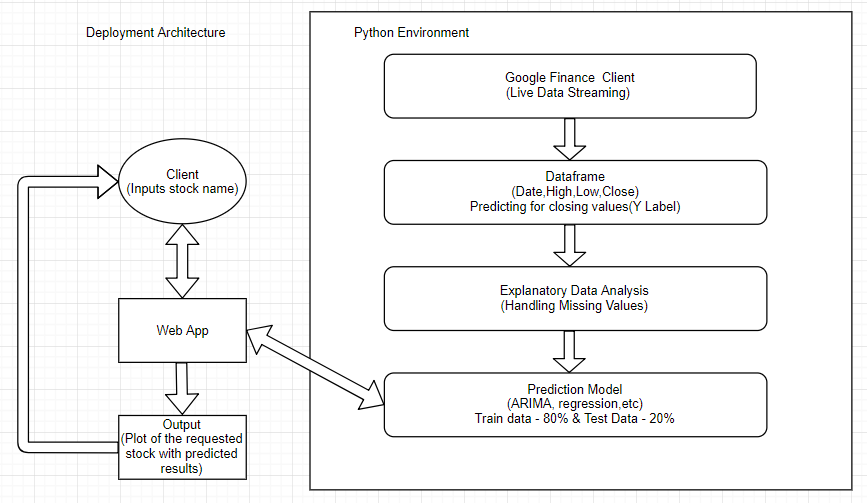
**Introduction:**

Predicting the stock price trend by interpreting the seemly chaotic market data has always been an attractive topic to both investors and researchers. Among those popular methods that have been employed, Machine Learning techniques are very popular due to the capacity of identifying stock trend from massive amounts of data that capture the underlying stock price dynamics.

**OBJECTIVE:**

The main objective of the dissertation is to examine the important and potential factors/predictors that could impact the stock market and develop a system for the investors and researchers in stock prediction.

**Work Flow:**

****

**Dataset:**

Firstly, we are using google finance package which is a Python module to get stock data from Google Finance API. This module provides **no delay**, **real time** stock data in NYSE & NASDAQ.

Here are the highlights and word definitions of the column names so as any reader can take as reference to understand how to read stocks:

### 1. Open: The price at the beginning of the trading day

### 2. High: The highest price the stock reached during the day

### 3. Low: The lowest price the stock reached during the day

### 4. Close: The final closing price of the stock for the day which becomes the opening price for the next day.

### 5. Volume: Volume is the number of shares that have been traded that is sold or brought throughout the day. It is important to understand Volume: A stock with low volume is thinly traded as it lacks liquidity, i.e., it would be difficult to buy or sell the stocks of that company easily. Similarly, if there is a stock that has a volume higher or more than the average trade-off dealing in this case can be volatile as the price fluctuation would be drastic and quick.

### 6. Adjust Closing Price: An adjusted closing price is a stock’s closing price on a given day of trading that has been amended to include any distribution and corporate actions that occur before the next day opening. The adjusted closing price is a useful tool when examining historical returns because it gives analyst an accurate representation of the firm’s equity value beyond the simple market price. It includes the corporate actions such as stock splits, dividends/distribution, and right offerings.

### 

### Explanatory Data Analysis and Handling Missing Values:

So, we are using Google finance module/package to update our dataset with new stock data on a daily basis but our original dataset had a time column attributing 4’O Clock closing time and another adjusted closing volume column which have been dropped because the new data being appended by the google finance package does not have these columns.

**Models Implemented:**

We are using the basic time series approach to predict stock trends, a popular and widely used statistical method for time series forecasting is the ARIMA model.

**ARIMA MODEL:**

ARIMA is an acronym that stands for AutoRegressive Integrated Moving Average. It is a class of model that captures a suite of different standard temporal structures in time series data.

An [ARIMA model](https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average) is a class of statistical models for analyzing and forecasting time series data.

It explicitly caters to a suite of standard structures in time series data, and as such provides a simple yet powerful method for making skillful time series forecasts.This acronym is descriptive, capturing the key aspects of the model itself. Briefly, they are:

* **AR**: *Autoregression*. A model that uses the dependent relationship between an observation and some number of lagged observations.
* **I**: *Integrated*. The use of differencing of raw observations (e.g. subtracting an observation from an observation at the previous time step) in order to make the time series stationary.
* **MA**: *Moving Average*. A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.

Each of these components are explicitly specified in the model as a parameter. A standard notation is used of ARIMA(p,d,q) where the parameters are substituted with integer values to quickly indicate the specific ARIMA model being used.

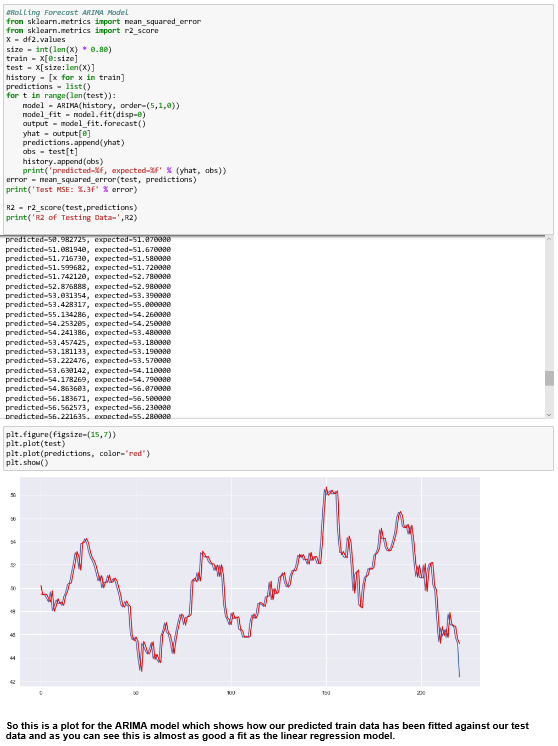
The parameters of the ARIMA model are defined as follows:

* **p**: The number of lag observations included in the model, also called the lag order.
* **d**: The number of times that the raw observations are differenced, also called the degree of differencing.
* **q**: The size of the moving average window, also called the order of moving average.

A linear regression model is constructed including the specified number and type of terms, and the data is prepared by a degree of differencing in order to make it stationary, i.e. to remove trend and seasonal structures that negatively affect the regression model.

A value of 0 can be used for a parameter, which indicates to not use that element of the model. This way, the ARIMA model can be configured to perform the function of an ARMA model, and even a simple AR, I, or MA model.

Adopting an ARIMA model for a time series assumes that the underlying process that generated the observations is an ARIMA process. This may seem obvious but helps to motivate the need to confirm the assumptions of the model in the raw observations and in the residual errors of forecasts from the model.



RED LINE – Prediction

BLUE LINE - Testing

**LINEAR REGRESSION MODEL:**

Linear regression is a statistical approach for modelling relationship between a dependent variable with a given set of independent variables. Linear regression models are a good starting point for regression tasks. Such models are popular because they can be fit very quickly and are very interpretable.



RED LINE – Prediction

BLUE LINE - Testing

As we can see from the plot this is a very good fit and when compared to the rest of our models our conclusion is that this is the best fit.

**RANDOM FOREST MODEL:**

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

****

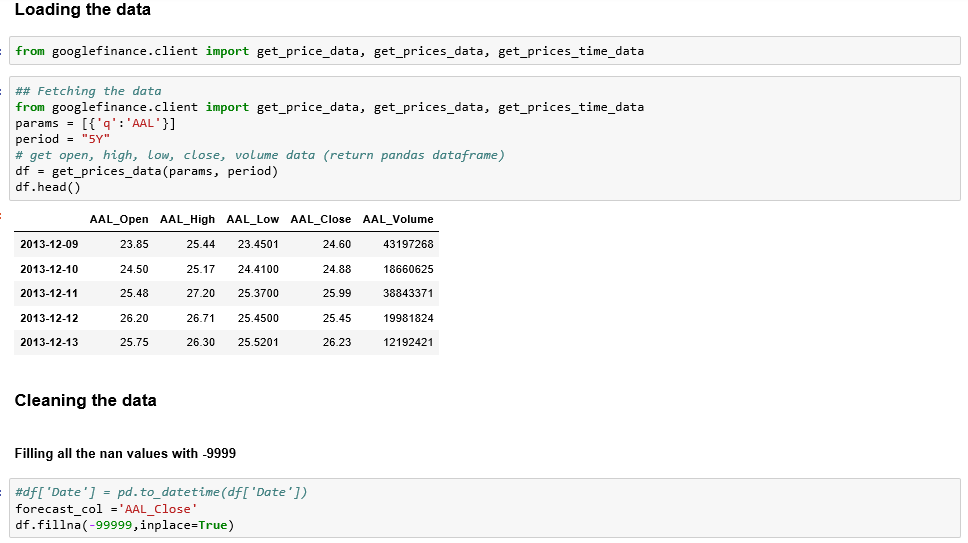
RED LINE – Prediction

BLUE LINE - Testing

So this is a plot for Random Forest model which shows how our predicted train data has been fitted against our test data and as you can see from the graph the prediction data which is represented by the red outline doesn't fit so well with the test data so we may conclude at a glance that Random forest is not such a good fit.

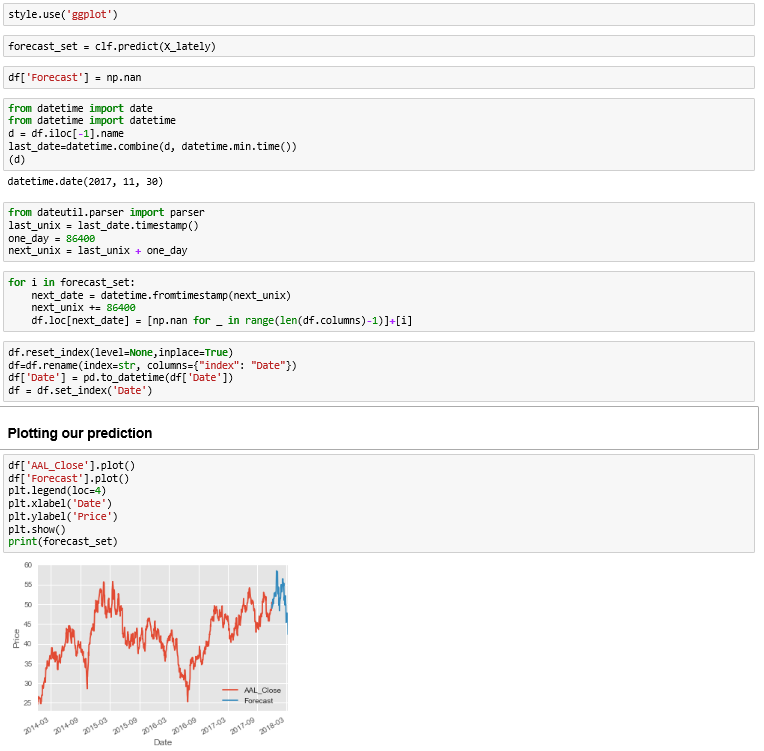
**Fitting Our Models and Creating Our Prediction Functions:**

1. **Linear Regression**

****

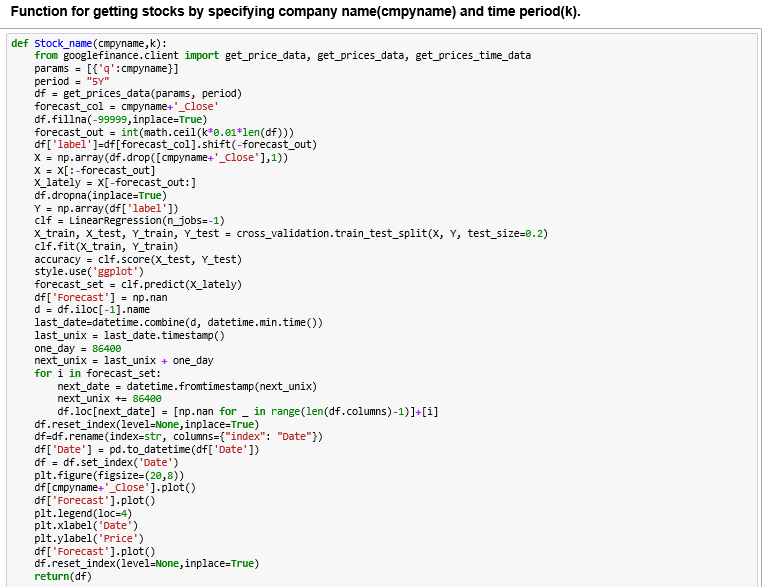
So, here we are just loading our data by specifying the company name and time period and we are replacing all nan values with -9999. After that we make an equation for weekly forecast where x\_lately is the column that we will be predicting.

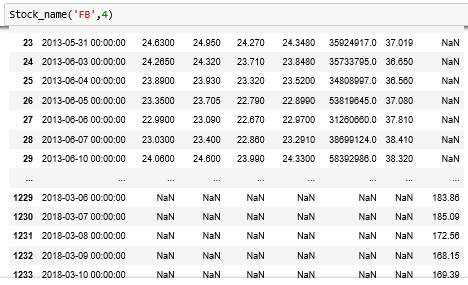


****

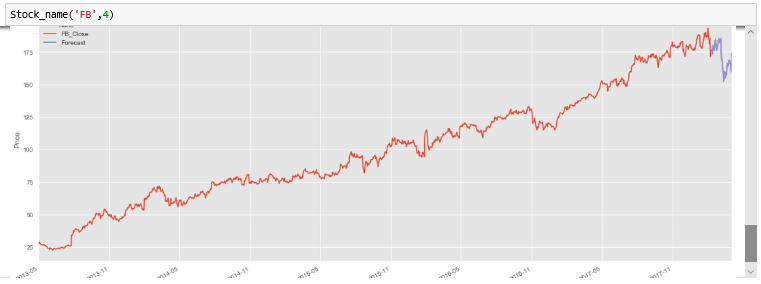
This is our prediction graph where the blue part specifies our predicted values.

So, next we create a function for all of this to serve to our web-app and our function takes in 2 parameters i.e. company name and the time period(k).



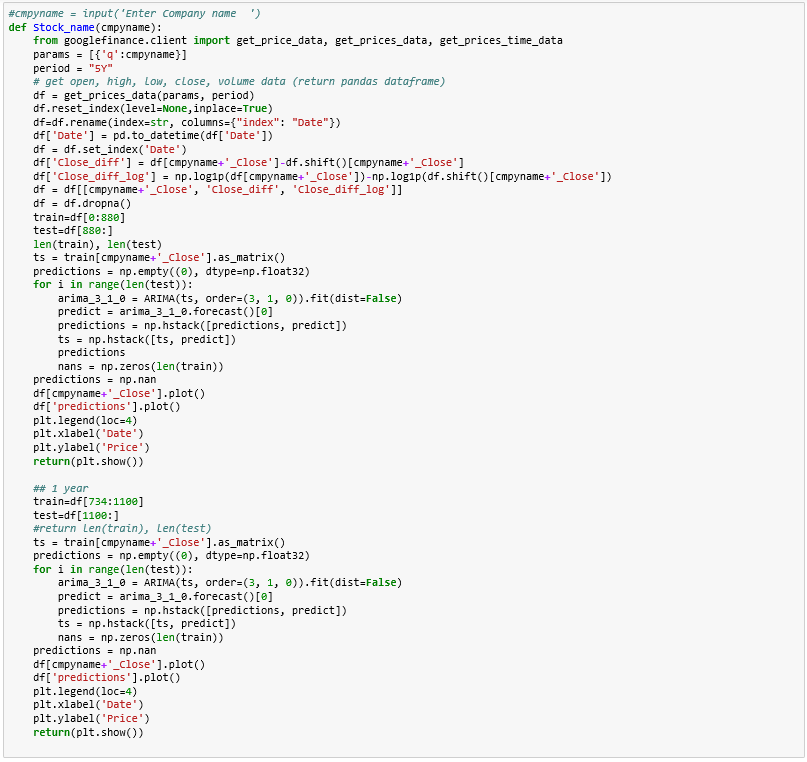


As you can see we get the predicted values and the graph with the predicted values on calling our function and giving it the parameters.



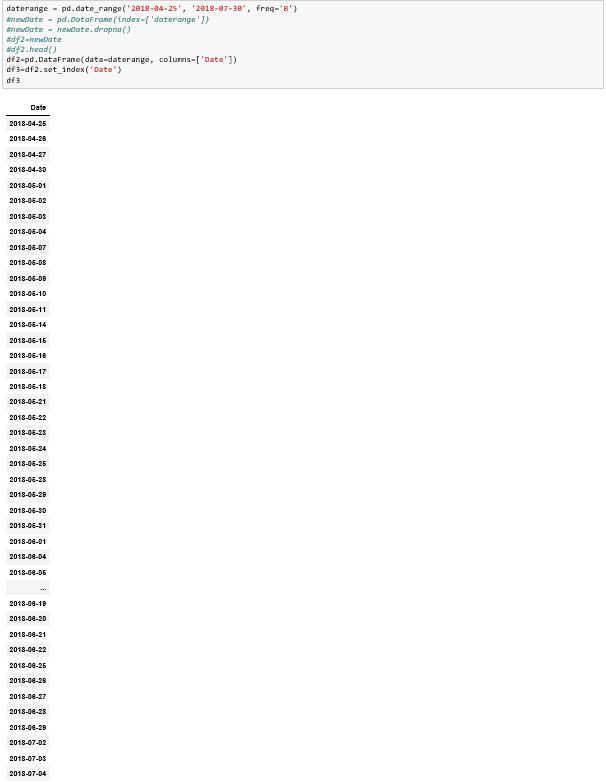
1. ARIMA MODEL

Just like we fit our model for linear regression we do the same for our ARIMA model by creating a parameterized function.



**TESTING:**

Next, we tried another method by appending the future dates we want to predict to our data frame. So, when we do a prediction, the prediction has no idea what date it is predicting for. In machine learning X and Y do not correspond to axis in a graph, in this case X is the feature and Y is the label, it just so happens the label is the price so Y is correct but X is not correct because date is not a feature, so that is why we are appending the date values and populating the data frame with the new dates and forecast values.



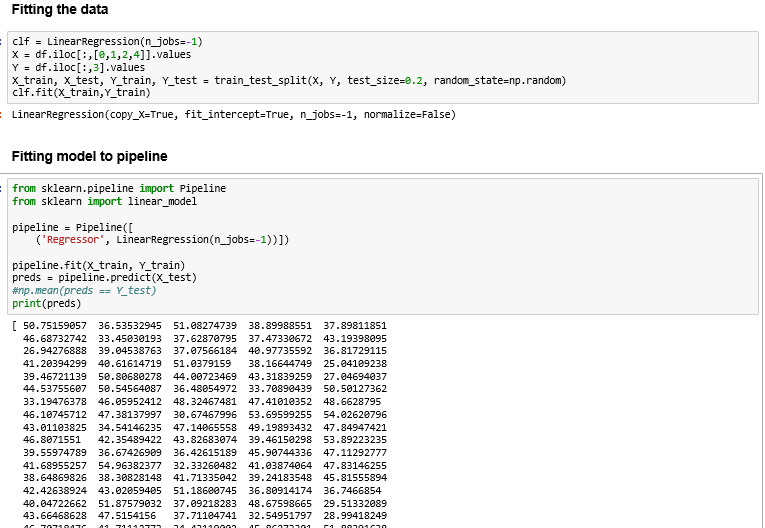


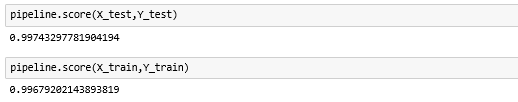
So, we are iterating through the forecast set taking each forecast and day and then setting those as the values in the data frame basically making the future features not a number.

The last line takes in all of the first columns, sets them to not a numbers and then the final column is whatever [i] is which is forecast in this case.

But as you can see by observing the predicted values they are all same and hence this method did not work.

**PIPELINING:**

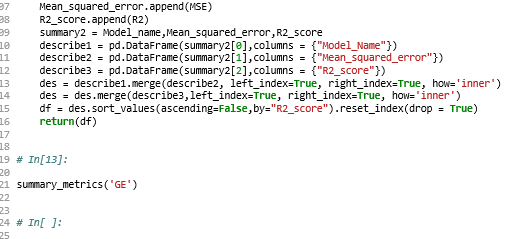
****

****

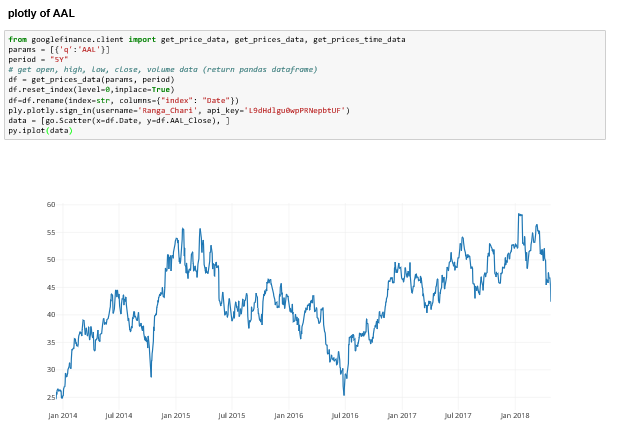
**SUMMARY METRICS:**

Different companies have different datasets and on running our models, they will train and test on each company’s dataset. So, we built a function where we pass the company name and our models run on that company’s dataset and we get the summary metrics ( R2 and RMSE) for the respective company.





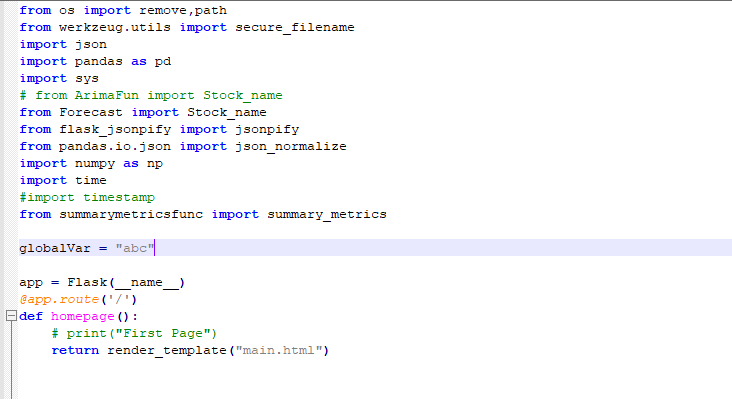
**PLOTLY:**

****

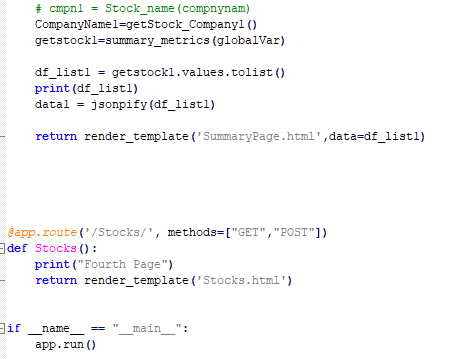
This graph represents the closing values of the company ‘AAL’

**FLASK And UI:**

**app.py** is our main flask application which calls all the other functions and html pages to run our app. A basic flask app requires a templates folder which contains our html pages.







So, we are making a flask app which is used for routing the various pages of our application. We have used a modular based design to call various processing ‘.py’ files in the app.py file.

Summary metrics: To show how our different models have worked on the selected company stock dataset and provide us with the R2 and RMSE values for the selected company’s dataset.

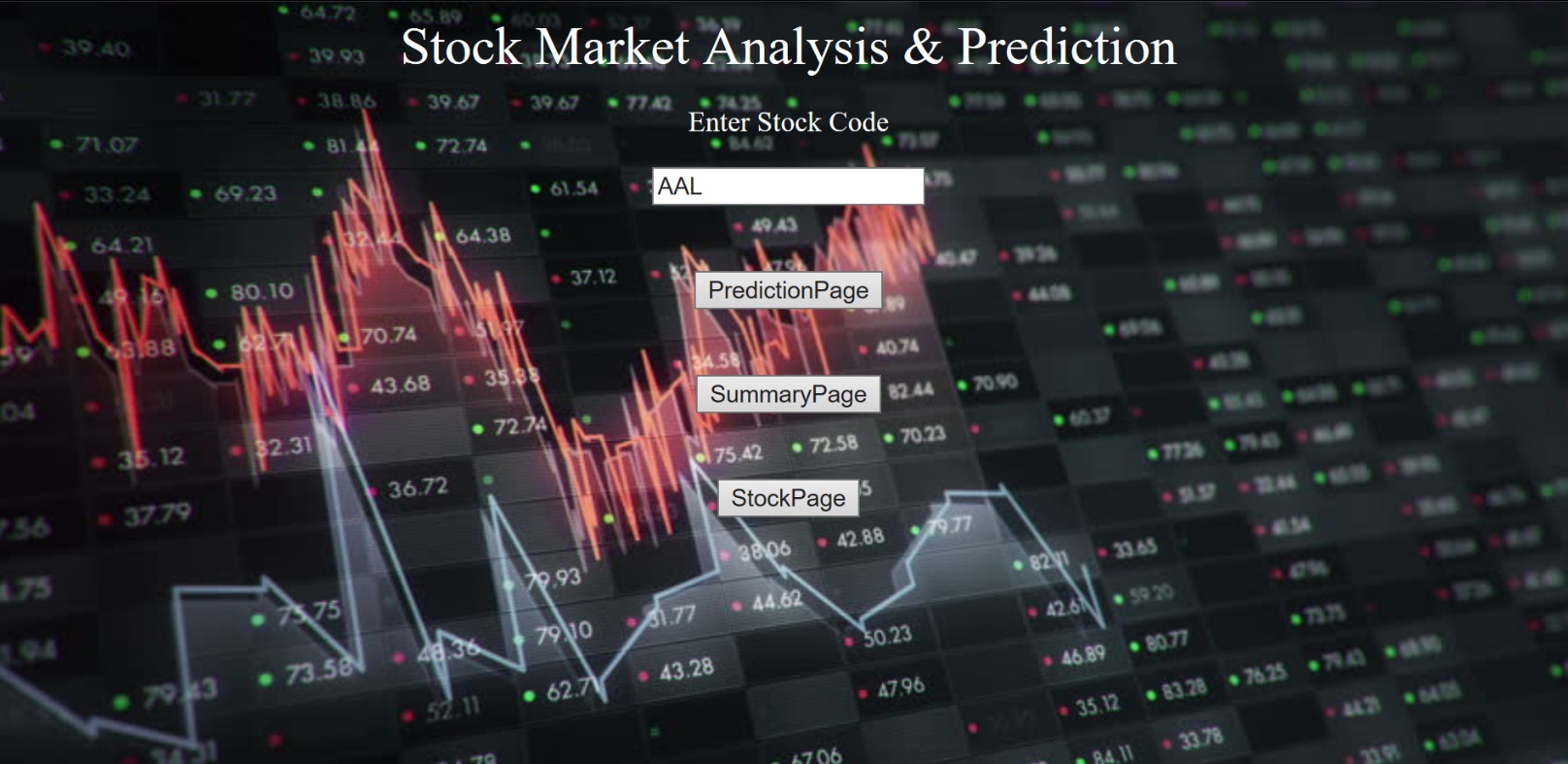
Plotly has also been integrated into our app to show an interactive graphical representation of how the graphs would show their trends.

We can check the summary metrics for our model by clicking the summary page button.

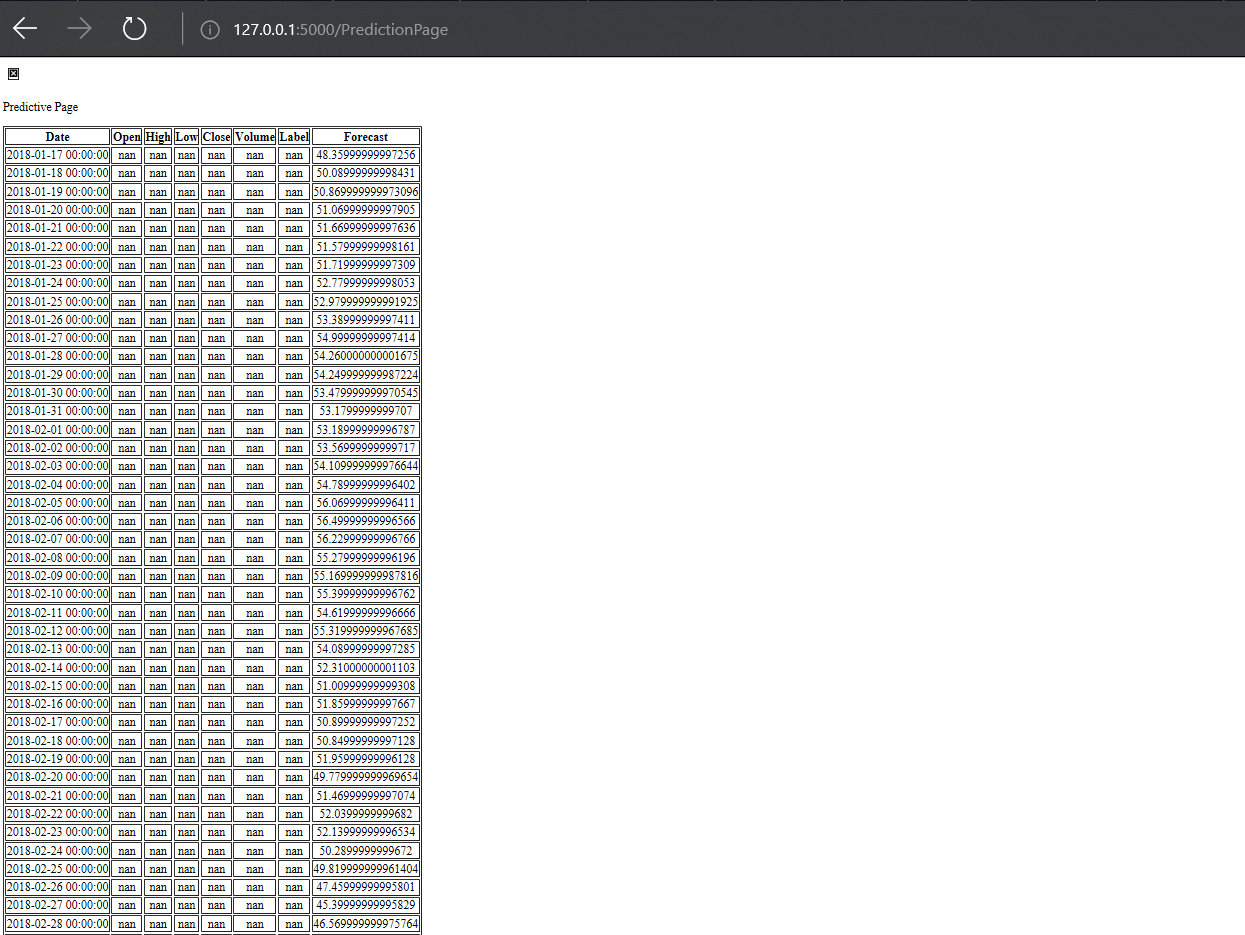
Lastly, we have integrated plotly graphs which can be accessed by clicking the Stock Page button.

This is our main.html page

We have the main page for our web app where we can input the company name and then check the forecasting values by clicking on the prediction page button.



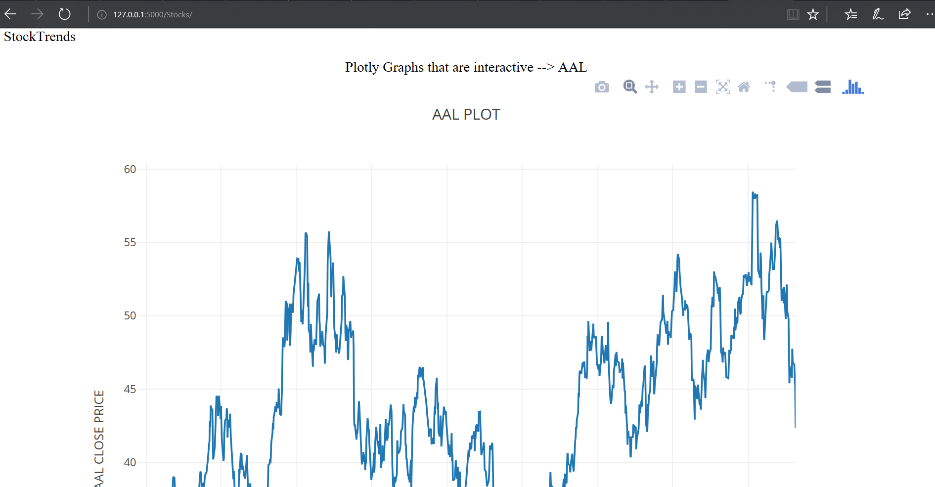
This is our prediction page which shows the predicted stock forecast values for the selected company.



This is our summary metrics page which is showing how the models performed on the basis of R2 and RMSE scores.



Finally, we have our stock page which has been integrated with plotly to show interactive graphs.



**CONCLUSION:**

By using this web application people can access any stock registered under NYSE and analyze the live stock price along with predictive forecasting with 99% accuracy which will eventually help them in making long term investments and avoid losses.

## EC2 Instance of the app[¶](http://localhost:8888/notebooks/Untitled4.ipynb?kernel_name=python3#EC2-Instance-of-the-app)

ec2-54-86-196-157.compute-1.amazonaws.com