Stock Market Prediction

Group - 4

S. Bhanoday

Rohith Surisetty

Bhavith Dulipalla

Professor - Niteesh Sahni

Table of Contents

ABSTRACT

Stock market is a place where people buy/sell shares of publicly listed companies. It offers a platform to facilitate seamless exchange of shares. In simple terms, if A wants to sell shares of XYZ Industries, the stock market will help him to meet the seller who is willing to buy XYZ Industries. Although experts in the field of Stock Market feel that no one can accurately predict the share prices of Stock Market since a lot of different parameters influence them. But a lot of individuals with the use of Machine Learning Algorithms have come close to solving the above problem in the past. A Machine Learning Model will be beneficial to all the stakeholders of the system if one can determine the flow of the market with a high degree of accuracy. Hence , a model which can extract the patterns out of a dataset and use that knowledge to predict the future behavior is essential. In this report we discuss and implement some of the Machine Learning Algorithms like Linear Regression , Ridge Regression , KNN, Random Forest to predict the future Adj.Close price of Facebook Stocks.

PROBLEM STATEMENT

The goal of our work is to collect the stock price of Facebook over a period of time and develop a model for forecasting the future stock price values. We hypothesize that it is possible for a machine learning model to learn from the features of the past movement patterns of a dataset and these learned features can be effectively exploited in accurately forecasting the future stock price values. We will be using algorithms such as Linear Regression , Ridge Regression , KNN, Random Forest to predict the future Adj.Close price of Facebook Stocks. To validate our hypothesis we will be comparing the R2 obtained from the Machine Learning Algorithms.

INTRODUCTION

DATASET DESCRIPTION

In this work, we have used a dataset of 1259 instances. This data set contains the details of the stock of Facebook Inc. This data set has 7 columns with all the necessary values such as opening price of the stock, the closing price of it, its highest in the day and much more. It has date wise data of the stock starting from 2015 to 2020(August). These Daily values were retrieved from Yahoo! Finance website.

<u>Description of Attributes</u>

Date - Gives the date of that particular day

Open - Opening price of the stock day

High - Max price of the stock for the day

Low - Min price of the stock for the day

Close - Closing price of stock for the day

Adj Close - Data is adjusted using appropriate split and dividend multipliers for the closing price for the day.

Volume - Volume are the physical number of shares traded of that stock on a particular day

Date	Open	High	Low	Close	Adj Close	Volume
02-03-2015	79	79.86	78.52	79.75	79.75	21662500
03-03-2015	79.61	79.7	78.52	79.6	79.6	18635000
04-03-2015	79.3	81.15	78.85	80.9	80.9	28126700
05-03-2015	81.23	81.99	81.05	81.21	81.21	27825700
06-03-2015	80.9	81.33	79.83	80.01	80.01	24488600
09-03-2015	79.68	79.91	78.63	79.44	79.44	18925100
10-03-2015	78.5	79.26	77.55	77.55	77.55	23067100
11-03-2015	77.8	78.43	77.26	77.57	77.57	20215700
12-03-2015	78.1	79.05	77.91	78.93	78.93	16093300

MATHEMATICS BEHIND MODELS USED

LINEAR REGRESSION

The simple linear regression is used to predict a quantitative outcome y on the basis of one single predictor variable x. The goal is to build a mathematical model (or formula) that defines y as a function of the x variable.

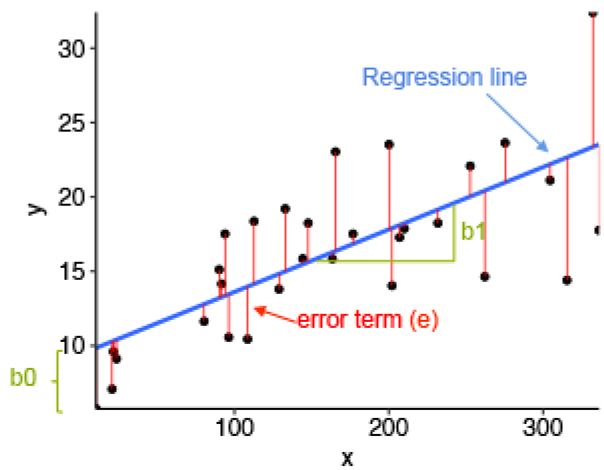
Once we build a statistically significant model, it's possible to use it for predicting future outcomes on the basis of new x values.

The mathematical formula of the linear regression can be written as y = b0 + b1*x + e, where:

- b0 and b1 are known as the regression *beta coefficients* or *parameters*:
    - b0 is the *intercept* of the regression line; that is the predicted value when x = 0.
    - b1 is the *slope* of the regression line.
- e is the *error term* (also known as the *residual errors*), the part of y that can be explained by the regression model

The figure below illustrates the linear regression model, where:

- the best-fit regression line is in blue
- the intercept (b0) and the slope (b1) are shown in green
- the error terms (e) are represented by vertical red lines



RIDGE REGRESSION

Ridge regression is also known as L2 regularization and it is used for variable selection and at the same time to reduce variance in the model.Ridge regression proceeds by adding a small value **k** to the diagonal elements of the correlation matrix i.e ridge regression got its name since the diagonal of ones in the correlation matrix are thought to be a ridge.

It is very similar to linear regression in the sense that it minimizes whereas linear regression just minimizes RSS.

The Ridge regression obtains the parameters $\beta_0, \beta_1, \ldots, \beta_p$ by minimising:

$$\sum_{i=1}^{N}\left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j}\right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \left\|\hat{\beta}\right\|_2^2.$$

Here $\hat{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)$ and $\left\|\hat{\beta}\right\|_2 = \sqrt{\sum_{j=1}^{p} \beta_j^2}.$

The constant $\lambda \geq 0$ is called the tuning parameter (inpractice found via cross-validation) and the second term in the above equation is the penalty that is imposed on the error RSS for choosing a large λ. Very large λ, the minimization would force the coefficient of λ to shrink towards 0. So, if some parameters become 0, then the corresponding terms get dropped from the model.

K-NEAREST NEIGHBOURS

The K-nearest neighbor algorithm creates an imaginary boundary to classify the data. When new data points are added for prediction, the algorithm adds that point to the nearest of the boundary line. It follows the principle of “Birds of a feather flock together.” This algorithm can easily be implemented in the R language.

K-NN Algorithm

1. Select K, the number of neighbors.
2. Calculate the Euclidean distance of the K number of neighbors.
3. Take the K nearest neighbors as per the calculated Euclidean distance.
4. Count the number of data points in each category among these K neighbors.
5. The new data point is assigned to the category for which the number of the neighbor is maximum.

For distance metrics, we will use the Euclidean metric.

$$d(x, x') = \sqrt{(x_1 - x_1')^2 + \ldots + (x_n - x_n')^2}$$

Finally, the input x gets assigned to the class with the largest probability.

$$P(y = j | X = x) = \frac{1}{K} \sum_{i \in \mathcal{A}} I(y^{(i)} = j)$$

Random Forest

Random forest is an improvement to the process of bagging.(Bagging: Taking repeated samples from the (single) training data set of the same size. In this approach we generate B different training data sets. We then train our method on the bth training set in order to get a single prediction, and finally average all the predictions). As in bagging, we build a number of decision trees on training samples. But when building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors. A fresh sample of m predictors is taken at each split, and typically we choose m ≈ Öp i.e., the number of predictors considered at each split is approximately equal to the square root of the total number of predictors.

$$h(x) = \frac{1}{B} \sum_{j=1}^{B} h_j(x)$$

EXPERIMENTAL ANALYSIS

EXPLORATORY DATA ANALYSIS

The Data set has been loaded and been saved into fb2 using the read.csv function.

```
# Data loading and preprossesing
fb2 <- read.csv(file = "FB.csv")
```

Now let us look into the brief summary of our dataset

```
#Exploratory Data Analysis
summary(fb2)
```

```
      Date                Open             High             Low
 Min.   :2015-03-02   Min.   : 77.03   Min.   : 77.89   Min.   : 72.0
 1st Qu.:2016-05-29   1st Qu.:116.52   1st Qu.:117.55   1st Qu.:114.9
 Median :2017-08-28   Median :150.92   Median :152.25   Median :149.1
 Mean   :2017-08-28   Mean   :147.27   Mean   :148.73   Mean   :145.8
 3rd Qu.:2018-11-26   3rd Qu.:179.30   3rd Qu.:180.75   3rd Qu.:177.9
 Max.   :2020-02-28   Max.   :222.57   Max.   :224.20   Max.   :221.3
     Close          Adj.Close          Volume
 Min.   : 77.46   Min.   : 77.46   Min.   :  5913100
 1st Qu.:116.24   1st Qu.:116.24   1st Qu.: 13860950
 Median :150.64   Median :150.64   Median : 18697200
 Mean   :147.32   Mean   :147.32   Mean   : 22317643
 3rd Qu.:179.53   3rd Qu.:179.53   3rd Qu.: 25300550
 Max.   :223.23   Max.   :223.23   Max.   :169803700
```

The following function prints the internal structure for the features of the dataset

```
str(fb2)
```

```
'data.frame':	1259 obs. of  7 variables:
 $ Date     : Date, format: "2015-03-02" "2015-03-03" ...
 $ Open     : num  79 79.6 79.3 81.2 80.9 ...
 $ High     : num  79.9 79.7 81.2 82 81.3 ...
 $ Low      : num  78.5 78.5 78.8 81.1 79.8 ...
 $ Close    : num  79.8 79.6 80.9 81.2 80 ...
 $ Adj.Close: num  79.8 79.6 80.9 81.2 80 ...
 $ Volume   : int  21662500 18635000 28126700 27825700 24488600 18925100 23067100 20215700 16093300 18557300 ...
```

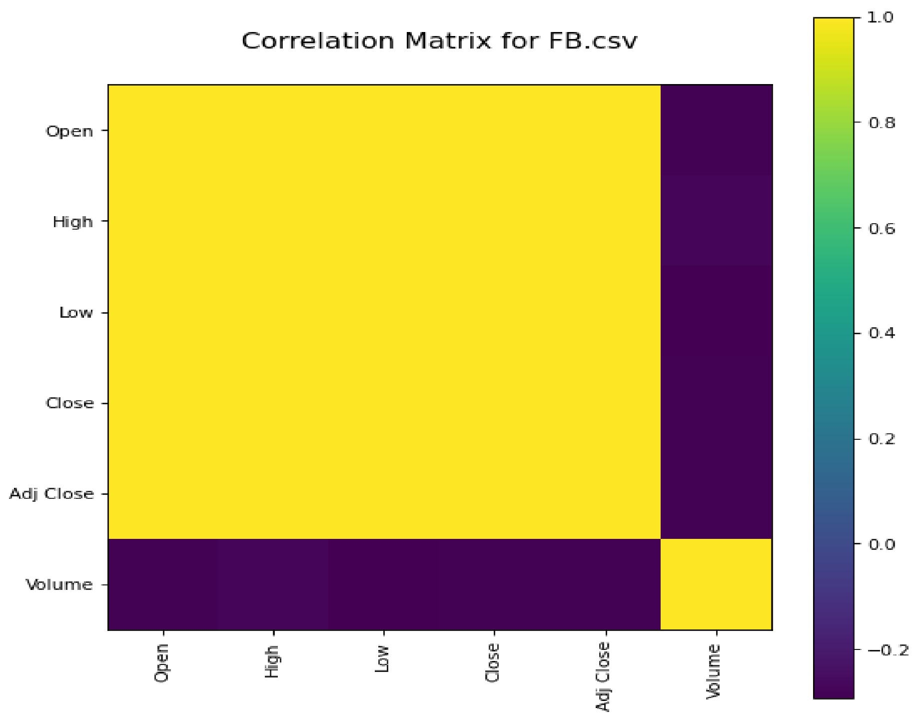
This function prints the sum of all the null values in the dataset . Since there are no null values in our dataset, we get the sum is 0

```
sum(is.na(fb2)) #prints the number of null values
0
```

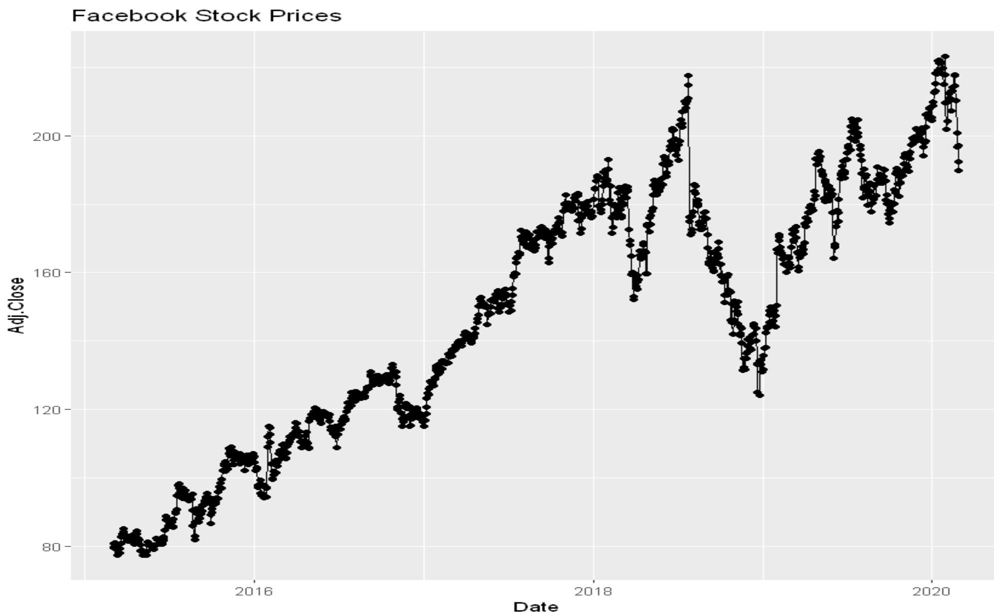
Correlation matrix -

A correlation matrix is a table showing correlation coefficients between sets of variables. Each random variable (Xi) in the table is correlated with each of the other values in the table (Xj). This allows you to see which pairs have the highest correlation .

Now let's look at the correlation matrix for our dataset



We will take Adj.Close as our target variable throughout the entire process . Lets see the plot of Adj.Close vs Date for our dataset



Data Split -

Since the dataset consists of a time series data , we will be dividing the dataset into training and test as shown in the code below

The training data consists of 989 values whereas the test data consists of 270 values which is approximately 72.7% of training data and 27.3% of testing data

```
#Split the data into train and test - use data before year 2019 as train data, and 2019 onwards as test data
fb2_split <-  split(fb2, fb2$Date < as.Date("2019-02-02"))
training <- as.data.frame(fb2_split$`TRUE`)
test <-  as.data.frame(fb2_split$`FALSE`)
dim(training)
dim(test)
```

```
989  7

270  7
```

<u>Algorithms</u> -

Linear regression:

We performed Linear Regression for Adj.Close vs Date

```r
#LinearRegression
#Train model
lm <- lm(Adj.Close ~ Date , data = training)
summary(lm)
coefficients(lm)
plot(fb2$Date,fb2$Adj.Close)
abline(lm, col="red")

#Predict on test data
lm_pred <- predict(lm, test)
```

Ridge Regression -

We performed Ridge Regression for Adj.Close vs all other features

```r
#RidgeRegression
# Custom Control Parameters
custom <- trainControl(method = "repeatedcv",
                       number = 10 ,
                       repeats = 5,
                       verboseIter = T)
#Train model
set.seed(1234)
ridge <- train(Adj.Close ~ .,training, method = "glmnet",
               tuneGrid = expand.grid(alpha = 0,lambda = seq(0.0001,1,length = 100)), trControl = custom)
#Predict on test data
rr_pred <- predict(ridge, newdata = test)
```

KNN -

We performed KNN algorithm for Adj.Close vs all other features

```r
# KNN Model
trControl <- trainControl(method = 'repeatedcv',number = 10,repeats = 3)
#Train model
set.seed(333)
knn <- train(Adj.Close ~.,data = training,tuneGrid = expand.grid(k=1:70),method = 'knn',metric = 'Rsquared',
             trControl = trControl,
             preProc = c('center', 'scale'))

# Model Performance
knn
plot(knn)
#Predict on test data
knn_pred <- predict(knn, newdata = test)
```

Random Forest -

We performed Random Forest for Adj.Close vs all other features

```
#Train model
#Using mtry=3
set.seed(1)
rf=randomForest(Adj.Close~.,data=fb2,subset=rf_train,mtry=3,importance =TRUE)

#Predict on test data
rf_pred = predict(rf ,newdata=fb2[-rf_train ,])
```

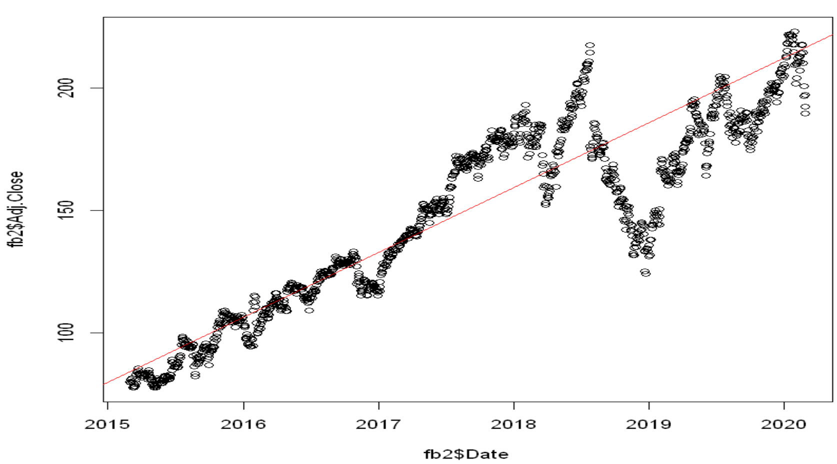
RESULT ANALYSIS

We have found the R squared value of all the above algorithms and compared them to get the best possible algorithm

Linear Regression -

```
R2_lm = R2(lm_pred,test$Adj.Close)
R2_lm
```

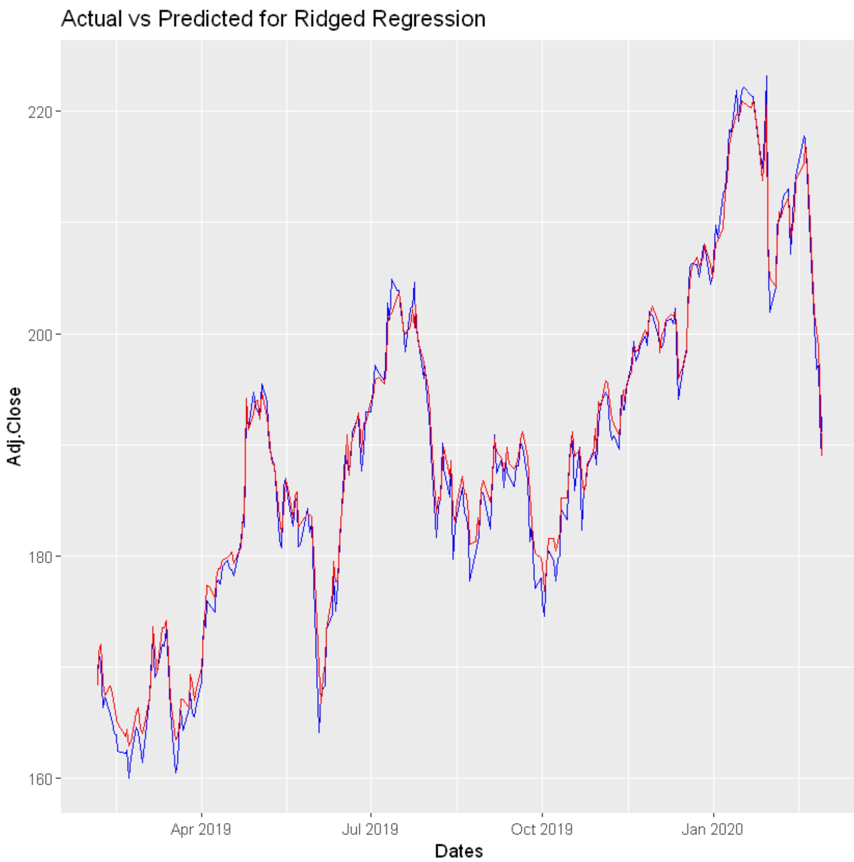
0.655347085615697



Ridge Regression -

```
R2_rr = R2(rr_pred,test$Adj.Close)
R2_rr
```

0.992785873119559



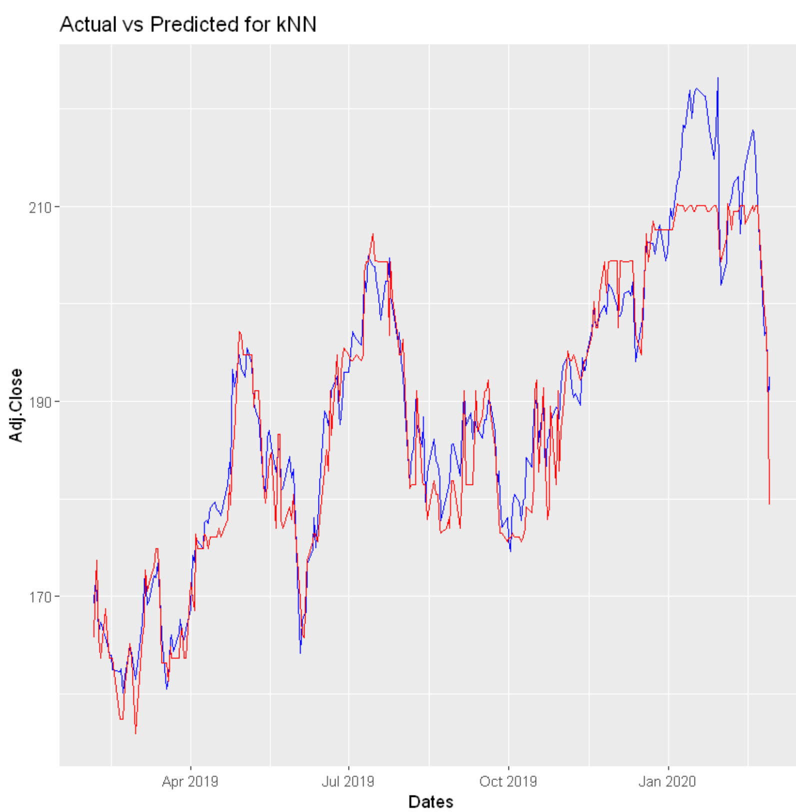
Blue color represents Actual values.

Red Color represents Predicted values.

KNN -

```
R2_knn = R2(knn_pred,test$Adj.Close)
R2_knn
```

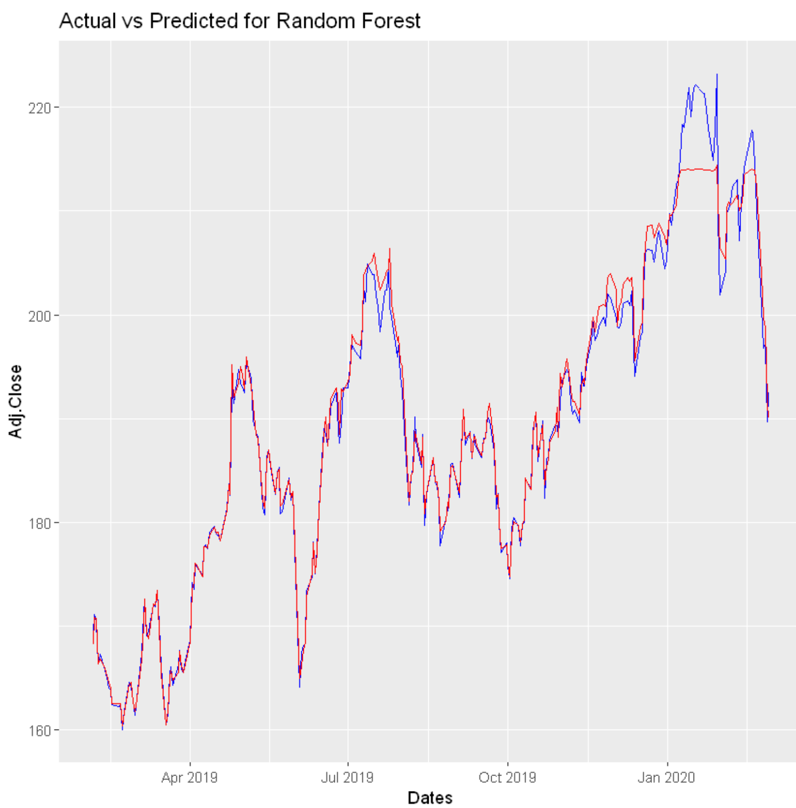
0.936939837271454



Random forest -

```
R2_rf = R2(rf_pred,rf_test)
R2_rf
```

0.984772403339923



CONCLUSION

We have used the above mentioned algorithms to predict the Adj.Close value of the stock market. We decide the best model by taking the R squared value into consideration. R squared value lies between 0 and 1. R squared value provides a measure of how well our model fits the data, it denotes the amount of variation explained in the response variable. A value closer to 1 is considered as a good R squared value

Algorithm Used	R Squared value
Linear Regression	0.655
Ridge Regression	0.993
KNN	0.937
Random Forest	0.985

From the above table , we can see that the R squared value of **Ridge Regression** is more close to 1 as compared to the other models , hence this model has better performance and it fits our data the best .

Actual Values	Predicted Values
169.25	168.3586
171.16	171.3419
170.49	172.0657
166.38	168.6720
167.33	167.4822

The above table shows Actual vs Predicted values of Ridge Regression algorithm for the 1st five inputs of test data

REFERENCES

https://finance.yahoo.com/quote/FB/history/?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAM-0M9MTTIdg_5EVkWJtaeXqsdpmdL89Ug_93qy7S9M-HUJzQ1cMQAKheQuREnRjobPTB3cFR6-MUhkdrrzxpILTxrbJq2GREyEG-jjZ4PCH7v_wZOYsilejZodl9jkqxk5Wv_zHnCNhUGRgvj0BOHUBwIBJEDRWajgIjj4sPOqy

KNN(K-Nearest Neighbour) algorithm, maths behind it and how to find the best value for K | by i-king-of-ml | Medium

Simple Linear Regression in R - Articles - STHDA

Stock Price Prediction Using Regression Analysis (ijser.org)

RPubs - Plotting two lines on one plot with ggplot2

Github Link - https://github.com/Rohith767/Stock-Market-Prediction