ASSIGNMENT-1

Name – Anshika Singh Section- 20BCS-23/B

UID – 21BCS8162 Subject - SMUR

MINI PROJECT

TITLE - Forecasting Stock Prices using historical data and market indicator

Introduction:

Stock market forecasting is a critical area of interest for investors, analysts, and financial professionals alike, aiming to predict future movements in stock prices based on historical data and market indicators. In this mini-project, we explore the application of R programming to forecast stock prices using historical data of Apple Inc. (AAPL) sourced from Yahoo Finance. Leveraging the power of quantitative analysis and time series forecasting techniques, we aim to develop a model that provides insights into potential future trends in AAPL stock prices.

Through this mini-project, we demonstrate a step-by-step approach to:

1.Retrieve historical stock price data using the quantmod package.

2.Visualize historical stock price trends and volatility using Bollinger Bands.

3.Conduct time series analysis to identify patterns and correlations in the data.

4.Calculate and analyze stock returns to understand the potential for future price movements.

5.Build and evaluate an AutoRegressive Integrated Moving Average (ARIMA) model for stock price forecasting.

6.Generate forecasts for future stock prices and assess the accuracy of the predictions.

By following this workflow, we aim to provide an introductory understanding of how R programming can be utilized for stock market forecasting, while highlighting the importance of data analysis and interpretation in making informed investment decisions.

Code for the given project:

Stock Market Price Prediction using R

##Importing Required Packages

```
install.packages("quantmod")
install.packages("tseries")
install.package("timeSeries")
install.packages("forecast")
install.packages("zoo")
install.packages("graphics")


library(quantmod)
library(tseries)
library(timeSeries)
library(forecast)
```

##Importing Dataset from Finance Websites...(Default yahoo)

```
getSymbols('AAPL', from = '2019-01-01', to = '2021-01-01')
View(AAPL)
#class(AAPL)
chartSeries(AAPL, subset = 'last 6 months', type = 'auto')
addBBands()
```

```
##Assigning columns of dataset
Open_prices = AAPL[,1]
High_prices = AAPL[,2]
Low_prices = AAPL[,3]
Close_prices = AAPL[, 4]
Volume_prices = AAPL[,5]
Adjusted_prices = AAPL[,6]
par(mfrow = c(2,3))
plot(Open_prices, main = 'Opening Price of Stocks (Over a given period)')
plot(High_prices, main = 'Highest Price of Stocks (Over a given period)')
plot(Low_prices, main = 'Lowest Price of Stocks (Over a given period)')
plot(Close_prices, main = 'Closing Price of Stocks (Over a given period)')
plot(Volume_prices, main = 'Volume of Stocks (Over a given period)')
plot(Adjusted_prices, main = 'Adjusted Price of Stocks (Over a given period)')
Predic_Price = Adjusted_prices
#class(Predic_Price)
```

Finding the Linear Relation between observations

```
par(mfrow = c(1,2))
Acf(Predic_Price, main = 'ACF for differenced Series')
Pacf(Predic_Price, main = 'PACF for differenced Series ', col = '#cc0000')
Auto_cf = Acf(Predic_Price, plot = FALSE)
Auto_cf
PAuto_cf = Pacf(Predic_Price, plot = FALSE)
PAuto_cf
print(adf.test(Predic_Price))
```

############ **Prediction of Return** ############

```
return_AAPL <- 100*diff(log(Predic_Price))

AAPL_return_train <- return_AAPL[1:(0.9*length(return_AAPL))]

AAPL_return_test <-
return_AAPL[(0.9*length(return_AAPL)+1):length(return_AAPL)]

auto.arima(AAPL_return_train, seasonal = FALSE)

fit <- Arima(AAPL_return_train, order = c(1,0,0))

preds <- predict(fit, n.ahead = (length(return_AAPL) -
(0.9*length(return_AAPL))))$pred

preds
```

########## **Forecasting Predicted Result** ########

```
test_forecast <- forecast(fit,h = 15)

test_forecast

par(mfrow = c(1,1))

plot(test_forecast, main = "Arima forecast for Apple Stock")


accuracy(preds, AAPL_return_test)
```

To build this mini project we have to followed the below steps:

Step 1: Importing Required Packages:

The code begins by importing necessary R packages for data manipulation, visualization, and time series forecasting. These packages include quantmod, tseries, forecast, and zoo.

Step 2: Importing Dataset from Finance Websites:

The getSymbols() function from the quantmod package is used to fetch historical stock price data for the specified symbol (in this case, "AAPL" for Apple Inc.) from a finance website (default is Yahoo Finance). The data is retrieved for the period from January 1, 2019, to January 1, 2021.

Step 3: Visualization of Stock Data:

After importing the data, the code visualizes the stock price data using the chartSeries() function from the quantmod package. Bollinger Bands are added to the chart using addBBands() to visualize volatility.

Step 4: Data Preparation:

The code assigns different columns of the dataset to separate variables, such as opening, high, low, closing, volume, and adjusted prices. These variables will be used for analysis and forecasting.

Step 5: Time Series Analysis:

The code then performs time series analysis to identify patterns and correlations in the data. It calculates the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots for the differenced series of adjusted prices. Additionally, it conducts the Augmented Dickey-Fuller (ADF) test to test for stationarity of the time series data.

Step 6: Prediction of Returns:

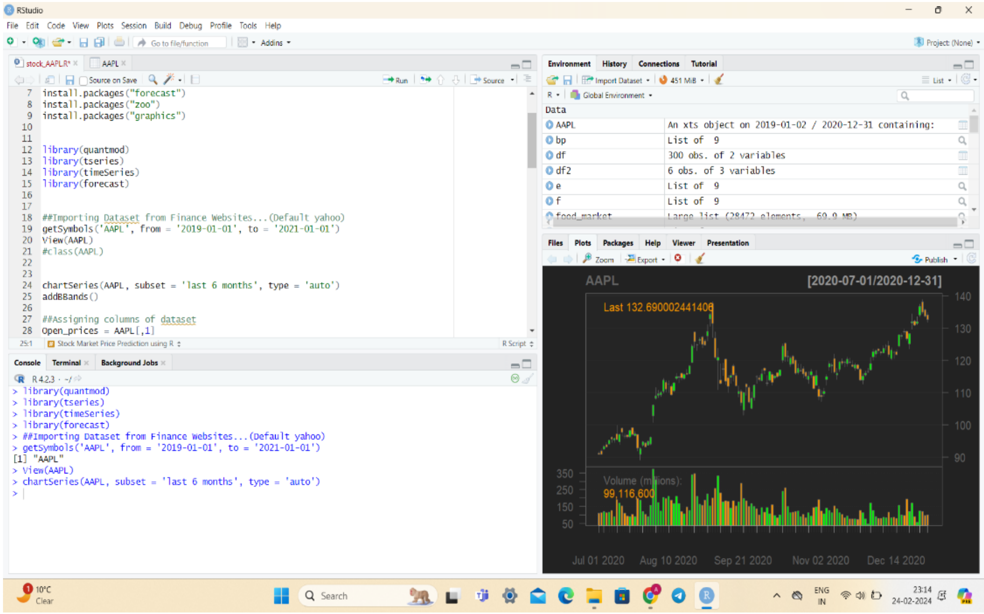
Next, the code calculates the returns of the stock prices using the logarithmic difference of adjusted prices. It splits the data into training and testing sets, fits an ARIMA (AutoRegressive Integrated Moving Average) model to the training data, and makes predictions for the testing set.

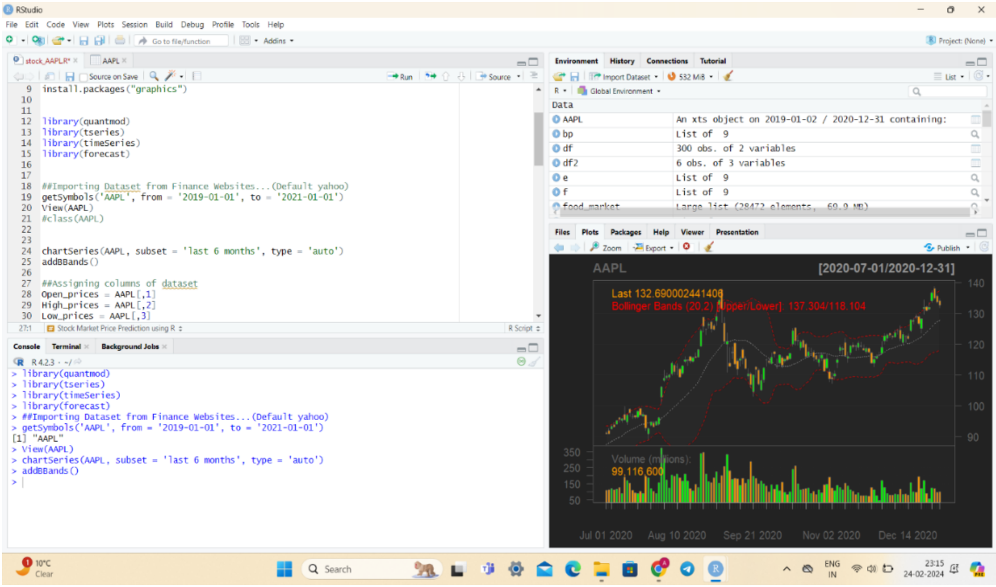
Step 7: Forecasting Predicted Results:

Finally, the code uses the fitted ARIMA model to forecast future stock returns. It generates forecasts for a specified horizon (in this case, 15 days) using the forecast() function from the forecast package and plots the forecasted results.

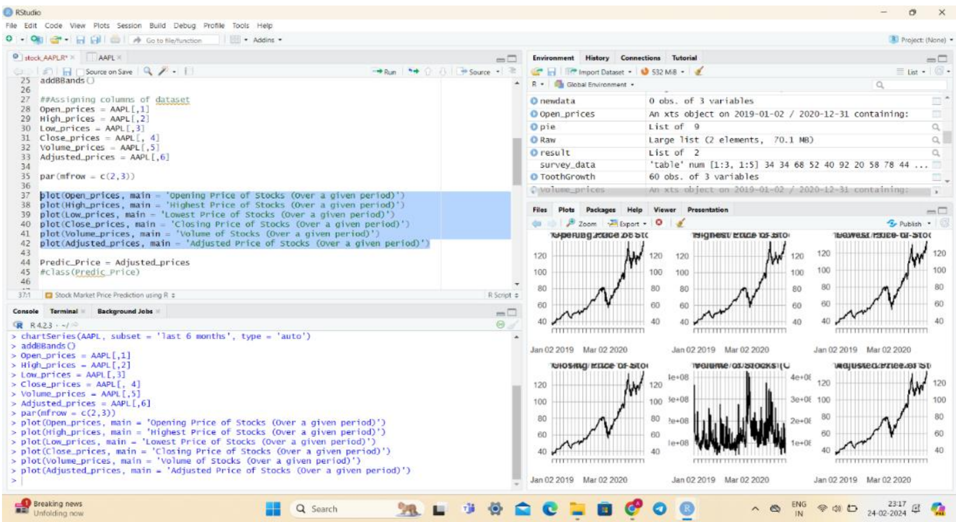
Overall, this mini-project demonstrates a basic workflow for forecasting stock prices using historical data and a market indicator (in this case, Bollinger Bands) in R programming. However, it's important to note that stock price forecasting is a complex task that often requires more sophisticated models and additional data sources for accurate predictions.

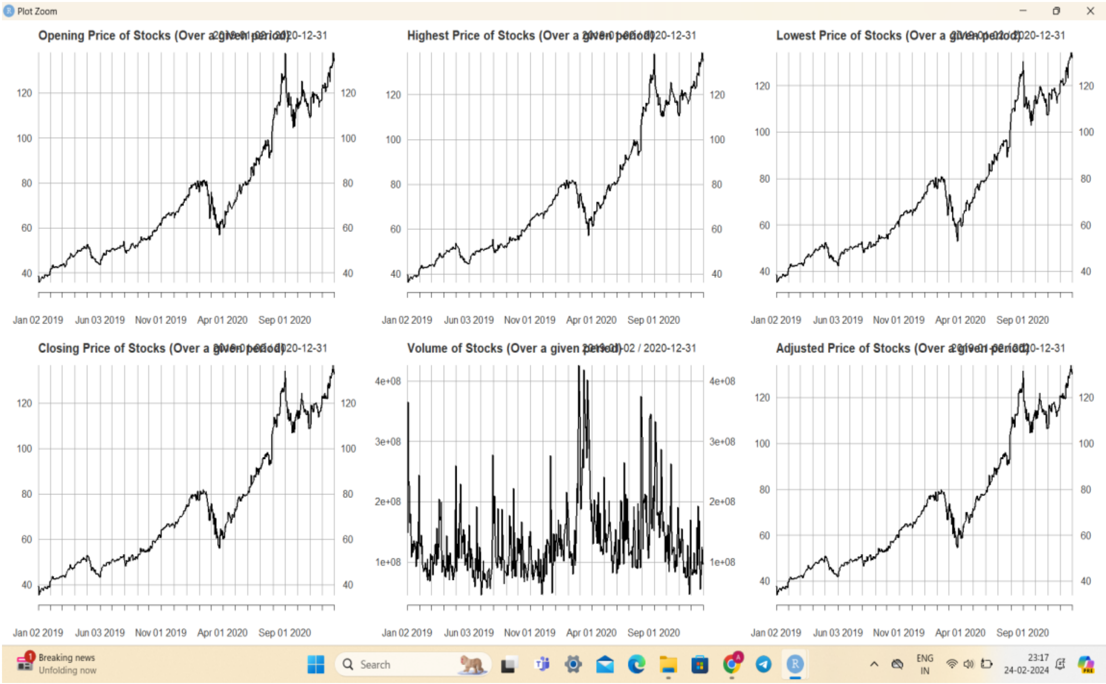
Result/Output:

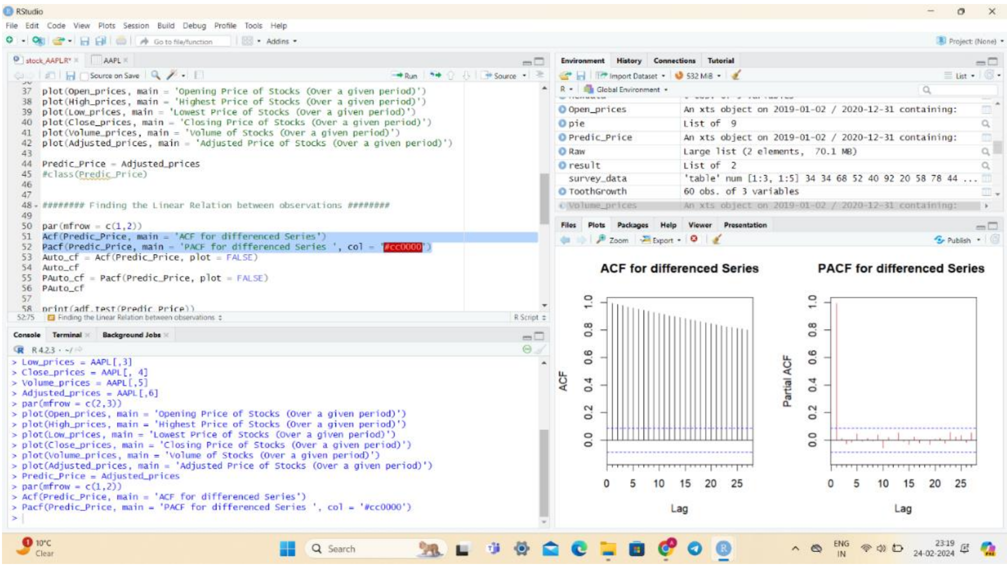


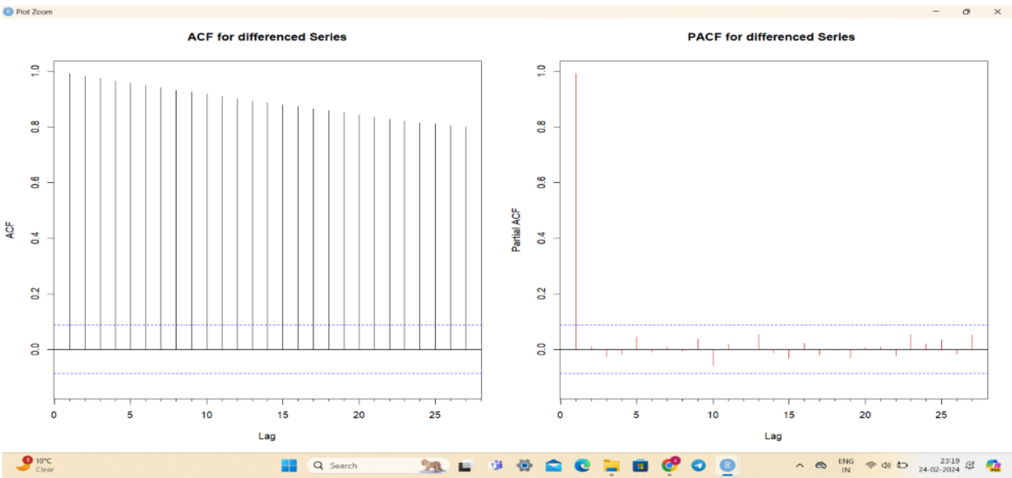

RStudio
stock_AAPL.R
install.packages("graphics")
library(quantmod)
library(tseries)
library(timeSeries)
library(forecast)
##Importing Dataset from Finance Websites...(Default yahoo)
getSymbols('AAPL', from = '2019-01-01', to = '2021-01-01')
View(AAPL)
#class(AAPL)
chartSeries(AAPL, subset = 'last 6 months', type = 'auto')
addBBands()
##Assigning columns of dataset
Open_prices = AAPL[,1]
High_prices = AAPL[,2]
Low_prices = AAPL[,3]
AAPL
An xts object on 2019-01-02 / 2020-12-31 containing:
300 obs. of 2 variables
6 obs. of 3 variables
[1] "AAPL"
[2020-07-01/2020-12-31]
Last 132.690002441406
Bollinger Bands (20,2) [Upper/Lower]: 137.304/118.104
Volume (millions):
99,116,600
Jul 01 2020
Aug 10 2020
Sep 21 2020
Nov 02 2020
Dec 14 2020

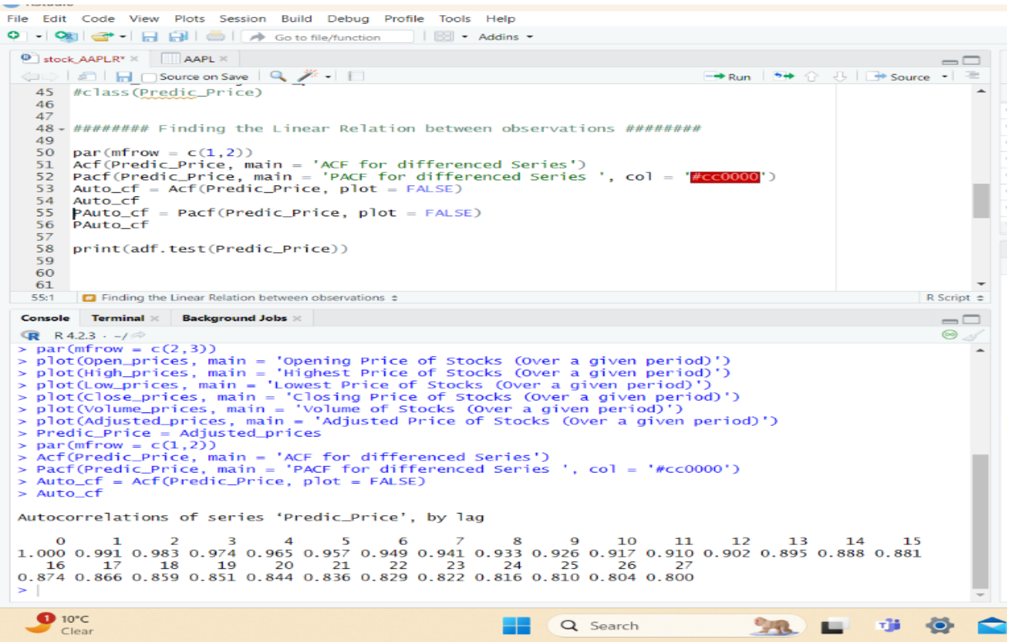

Plot Zoom
AAPL
[2020-07-01/2020-12-31]
Last 132.690002441406
Bollinger Bands (20,2) [Upper/Lower]: 137.304/118.104
Volume (millions):
99,116,600
Jul 01 2020
Jul 20 2020
Aug 03 2020
Aug 24 2020
Sep 14 2020
Oct 05 2020
Oct 19 2020
Nov 02 2020
Nov 16 2020
Dec 07 2020
Dec 28 2020

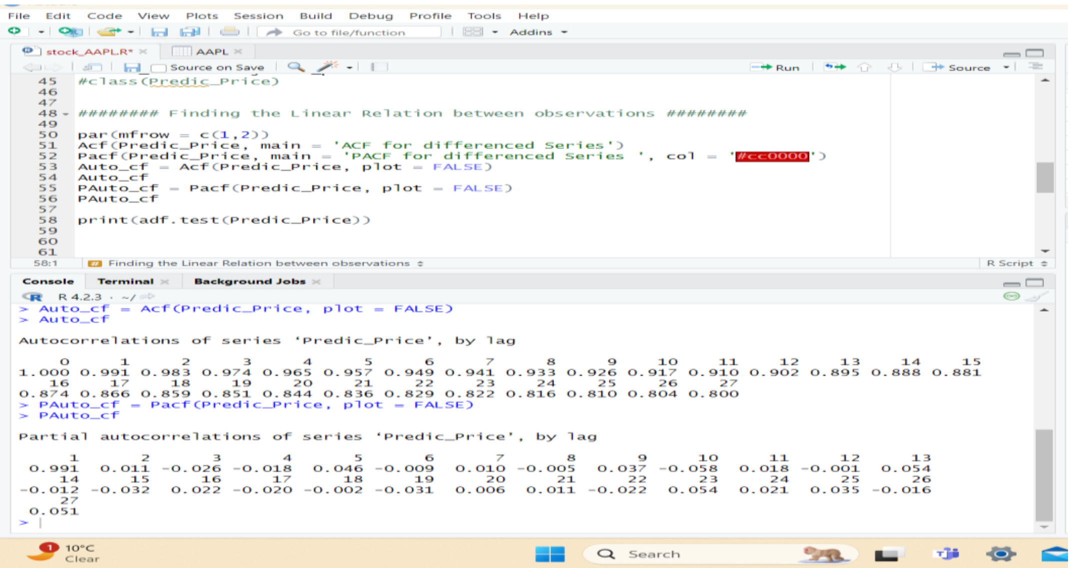

RStudio
addBBands()
##Assigning columns of dataset
Open_prices = AAPL[,1]
High_prices = AAPL[,2]
Low_prices = AAPL[,3]
Close_prices = AAPL[, 4]
Volume_prices = AAPL[,5]
Adjusted_prices = AAPL[,6]
par(mfrow = c(2,3))
plot(Open_prices, main = 'Opening Price of Stocks (Over a given period)')
plot(High_prices, main = 'Highest Price of Stocks (Over a given period)')
plot(Low_prices, main = 'Lowest Price of Stocks (Over a given period)')
plot(Close_prices, main = 'Closing Price of Stocks (Over a given period)')
plot(Volume_prices, main = 'Volume of Stocks (Over a given period)')
plot(Adjusted_prices, main = 'Adjusted Price of Stocks (Over a given period)')
Predic_Price = Adjusted_prices
#class(Predic_Price)
Jan 02 2019
Mar 02 2020

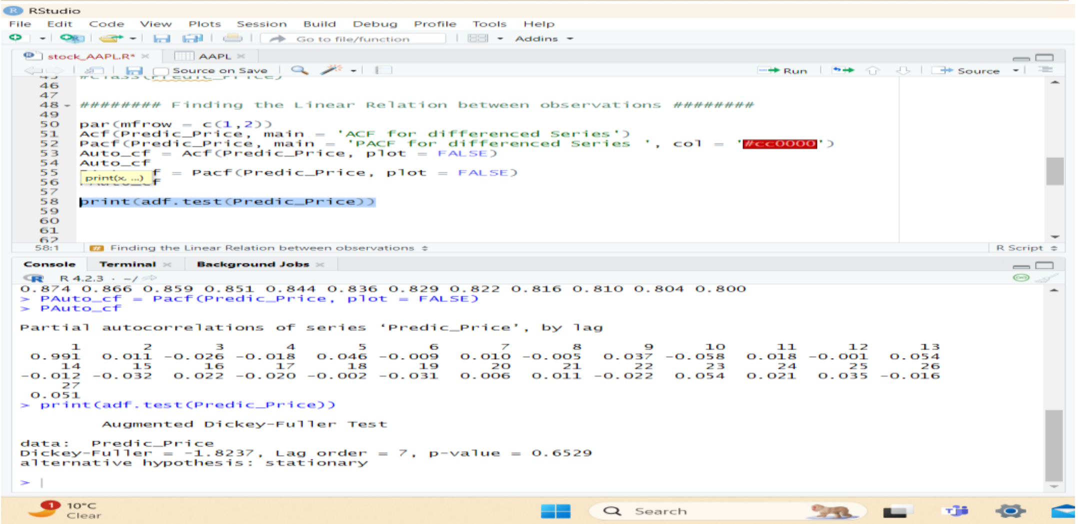

Plot Zoom
Opening Price of Stocks (Over a given period)
Highest Price of Stocks (Over a given period)
Lowest Price of Stocks (Over a given period)
Closing Price of Stocks (Over a given period)
Volume of Stocks (Over a given period)
Adjusted Price of Stocks (Over a given period)
Jan 02 2019
Jun 03 2019
Nov 01 2019
Apr 01 2020
Sep 01 2020


RStudio
stock_AAPL.R
AAPL
plot(Open_prices, main = 'Opening Price of Stocks (Over a given period)')
plot(High_prices, main = 'Highest Price of Stocks (Over a given period)')
plot(Low_prices, main = 'Lowest Price of Stocks (Over a given period)')
plot(Close_prices, main = 'Closing Price of Stocks (Over a given period)')
plot(Volume_prices, main = 'Volume of Stocks (Over a given period)')
plot(Adjusted_prices, main = 'Adjusted Price of Stocks (Over a given period)')
Predic_Price = Adjusted_prices
#class(Predic_Price)
######## Finding the Linear Relation between observations ########
par(mfrow = c(1,2))
Acf(Predic_Price, main = 'ACF for differenced Series')
Pacf(Predic_Price, main = 'PACF for differenced Series ', col = '#cc0000')
Auto_cf = Acf(Predic_Price, plot = FALSE)
Auto_cf
PAuto_cf = Pacf(Predic_Price, plot = FALSE)
PAuto_cf
print(adf.test(Predic_Price))
Console
R 4.2.3 · ~/
> Low_prices = AAPL[,3]
> Close_prices = AAPL[, 4]
> Volume_prices = AAPL[,5]
> Adjusted_prices = AAPL[,6]
> par(mfrow = c(2,3))
> plot(Open_prices, main = 'Opening Price of Stocks (Over a given period)')
> plot(High_prices, main = 'Highest Price of Stocks (Over a given period)')
> plot(Low_prices, main = 'Lowest Price of Stocks (Over a given period)')
> plot(Close_prices, main = 'Closing Price of Stocks (Over a given period)')
> plot(Volume_prices, main = 'Volume of Stocks (Over a given period)')
> plot(Adjusted_prices, main = 'Adjusted Price of Stocks (Over a given period)')
> Predic_Price = Adjusted_prices
> par(mfrow = c(1,2))
> Acf(Predic_Price, main = 'ACF for differenced Series')
> Pacf(Predic_Price, main = 'PACF for differenced Series ', col = '#cc0000')
ACF for differenced Series
PACF for differenced Series

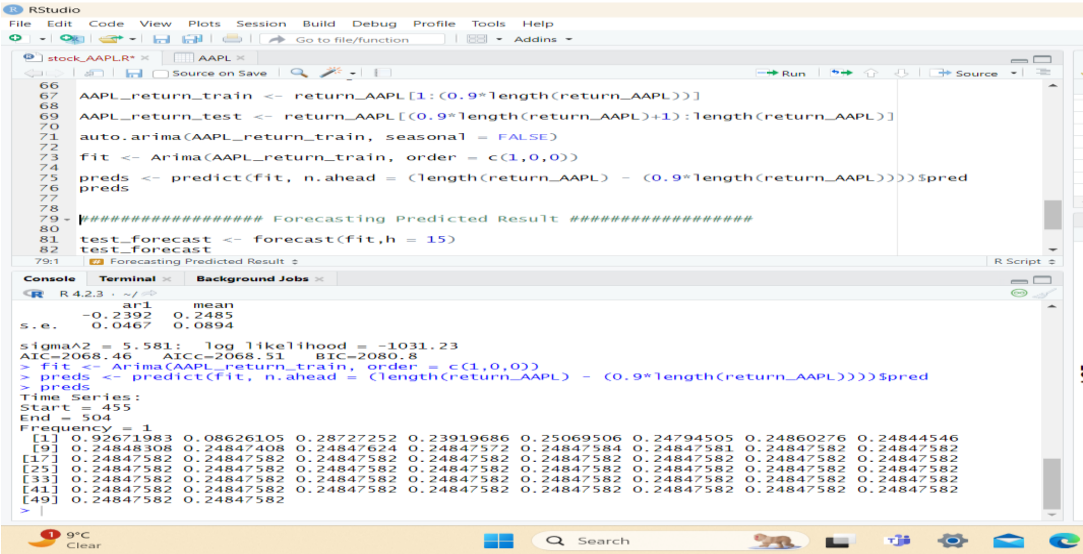

Plot Zoom
ACF for differenced Series
PACF for differenced Series
ACF
Partial ACF
Lag

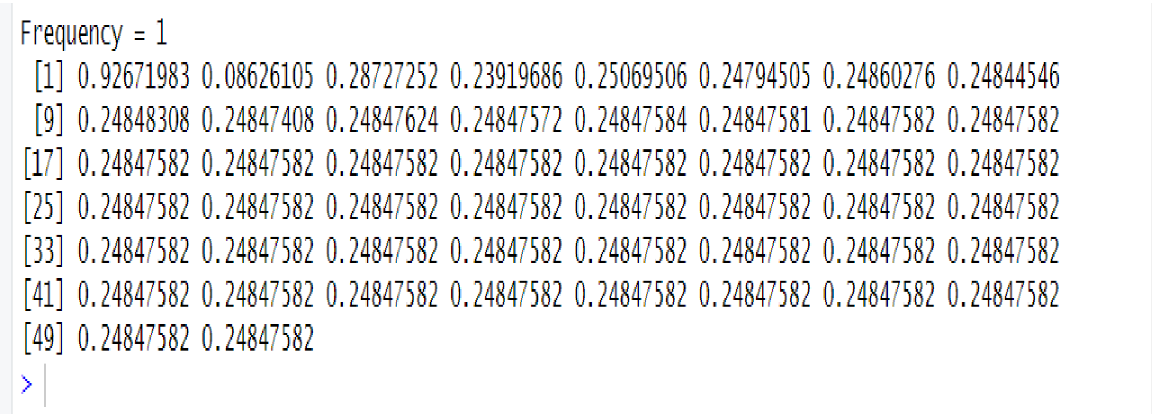




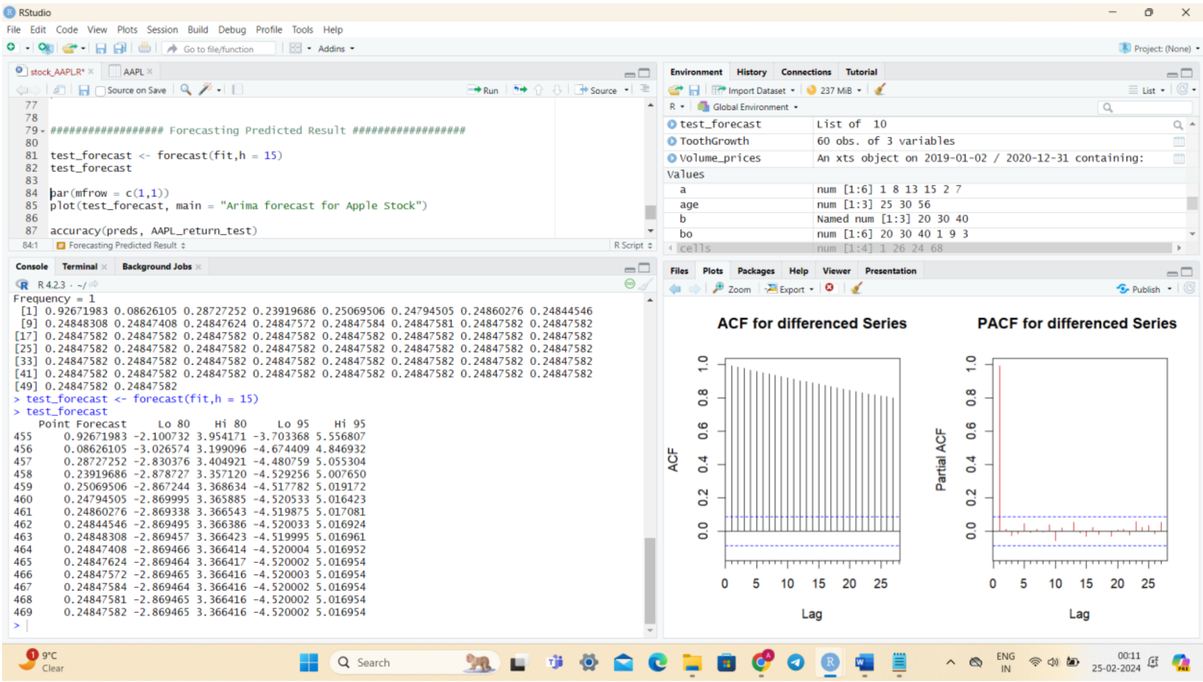


Prediction of Return:

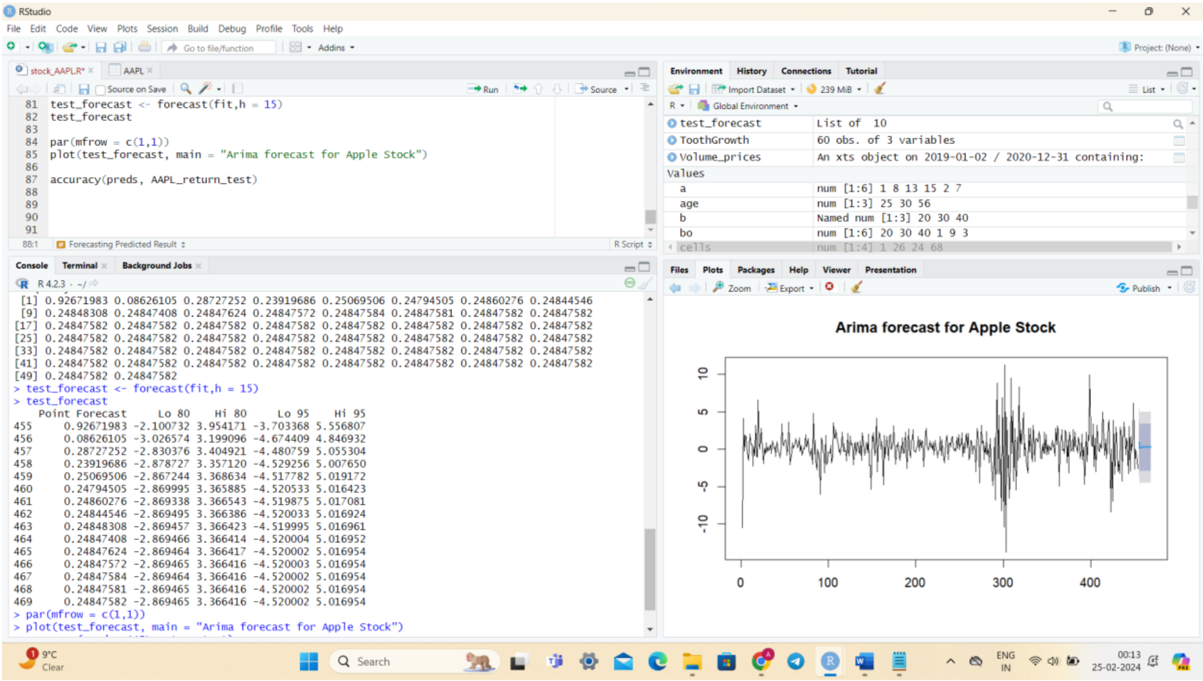


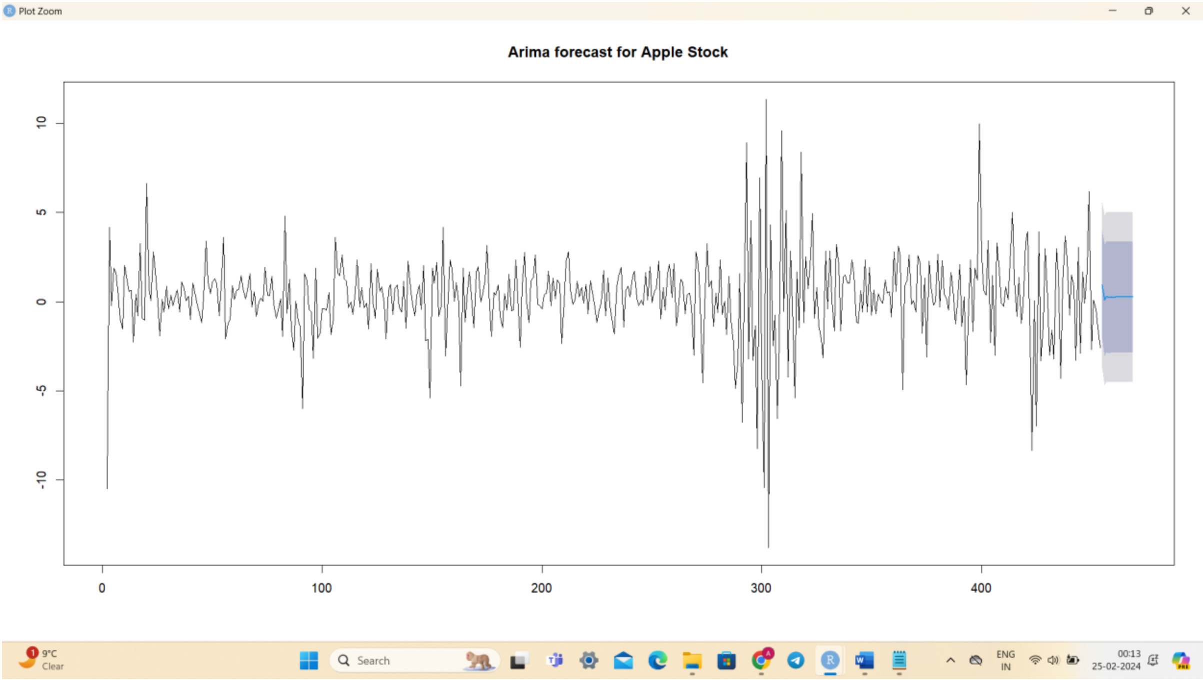


Forecasting Predicted Result



```
> test_forecast <- forecast(fit,h = 15)
> test_forecast
    Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
455     0.92671983 -2.100732 3.954171 -3.703368 5.556807
456     0.08626105 -3.026574 3.199096 -4.674409 4.846932
457     0.28727252 -2.830376 3.404921 -4.480759 5.055304
458     0.23919686 -2.878727 3.357120 -4.529256 5.007650
459     0.25069506 -2.867244 3.368634 -4.517782 5.019172
460     0.24794505 -2.869995 3.365885 -4.520533 5.016423
461     0.24860276 -2.869338 3.366543 -4.519875 5.017081
462     0.24844546 -2.869495 3.366386 -4.520033 5.016924
463     0.24848308 -2.869457 3.366423 -4.519995 5.016961
464     0.24847408 -2.869466 3.366414 -4.520004 5.016952
465     0.24847624 -2.869464 3.366417 -4.520002 5.016954
466     0.24847572 -2.869465 3.366416 -4.520003 5.016954
467     0.24847584 -2.869464 3.366416 -4.520002 5.016954
468     0.24847581 -2.869465 3.366416 -4.520002 5.016954
469     0.24847582 -2.869465 3.366416 -4.520002 5.016954
> 
```





Accuracy of model

```
> accuracy(preds, AAPL_return_test)
                 ME   RMSE      MAE      MPE     MAPE
Test set 0.02868578 2.0126 1.477334 54.19419 185.3524
>
```

Conclusion:

In this mini-project, we have explored the process of forecasting stock prices using historical data and market indicators in R programming. By leveraging the capabilities of quantitative analysis and time series forecasting techniques, we have gained insights into the potential future trends of Apple Inc. (AAPL) stock prices.

Through the utilization of the quantmod package, we retrieved historical stock price data from Yahoo Finance and visualized the trends using Bollinger Bands, allowing us to observe patterns and volatility in the data. Subsequently, by conducting time series analysis, we identified correlations and assessed the stationarity of the time series data.

Furthermore, we calculated stock returns and built an AutoRegressive Integrated Moving Average (ARIMA) model to forecast future stock prices. By splitting the data into training and testing sets, we evaluated the performance of our model and generated forecasts for future price movements.

While this mini-project provides a foundational understanding of stock market forecasting in R programming, it is essential to recognize the limitations and challenges associated with such predictions. Factors such as market sentiment, economic indicators, and unforeseen events can significantly impact stock prices, necessitating continuous refinement and validation of forecasting models.

In conclusion, this mini-project serves as a starting point for individuals interested in utilizing R programming for stock market analysis and forecasting. By combining technical analysis with quantitative methods, we can make more informed investment decisions and navigate the complexities of the financial markets with greater confidence.