Chandigarh University

Department of Computer Science & Engineering

Assignment

Student Name	Raghav Ranjan
UID	20BCS9898
Semester	8^{th}
Course Code	20SMP-460
Course Name	SMuR

Assignment-1

Name- Raghav Ranjan **UID**: 20BCS9898

Section: 20BCS-23-A **Subject: SMuR**

Mini Project

Title: Stock Market Price Prediction Project

Introduction

This project aims to predict stock market prices using the R programming language. We utilize historical stock data of Google (GOOG) obtained from Yahoo Finance and employ time series analysis techniques to forecast future stock prices. The project involves data preprocessing, exploratory data analysis, model building, and evaluation.

Data Collection

We obtained historical stock data for Google (GOOG) from Yahoo Finance spanning from January 1, 2019, to January 1, 2021. The dataset comprises daily stock prices, including opening, high, low, closing, volume, and adjusted prices. We utilized the quantmod package in R to retrieve and preprocess the data.

Data Exploration

Exploratory data analysis (EDA) is crucial for gaining insights into the underlying patterns and trends in the data. We visualized the stock prices using candlestick charts, which provide a comprehensive view of price movements over time. Additionally, we calculated technical indicators such as moving averages to identify key support and resistance levels.

Data Preprocessing

Before building the prediction model, we preprocessed the data to ensure its suitability for analysis. This involved extracting relevant columns, such as the adjusted closing prices, which account for factors like stock splits and dividends. We also checked for missing values and performed any necessary data transformations.

Model Building

We constructed a time series forecasting model using the Auto ARIMA algorithm available in the forecast package in R. The model was trained on the logarithmic returns of the adjusted closing prices, which helps capture the relative price changes over time. We split the dataset into training and testing sets, with 90% of the data allocated for training.

Model Evaluation

To assess the model's performance, we compared the predicted returns with the actual returns from the testing dataset. We calculated various forecast accuracy metrics, including Mean Absolute Error (MAE) and Mean Squared Error (MSE), to quantify the extent of prediction errors. Additionally, we visualized the forecasted stock prices alongside the actual prices to visually inspect the model's performance.

Results

The forecasting model yielded promising results, demonstrating its ability to capture the underlying patterns in the stock prices and make accurate predictions. The model's forecasted prices closely aligned with the actual prices, indicating its effectiveness in predicting future price movements. The forecast accuracy metrics further corroborated the model's reliability, with low error rates observed across various evaluation metrics.

Conclusion

In conclusion, this project showcases the application of time series analysis techniques for stock market price prediction using R. By leveraging historical stock data and advanced forecasting models, investors can gain valuable insights into future price movements and make well-informed investment decisions. The forecasting model developed in this project serves as a valuable tool for investors seeking to optimize their trading strategies and maximize returns in the stock market.

Future Work

Future work in this domain could focus on several areas of improvement and expansion. Firstly, incorporating additional features such as external factors (e.g., economic indicators, news sentiment) could enhance the predictive power of the model. Secondly, exploring ensemble methods and deep learning techniques could lead to more accurate and robust predictions, especially in highly volatile market conditions. Furthermore, improving model interpretability and developing user-friendly interfaces for real-time forecasting could enhance the practical applicability of the model in the financial industry.

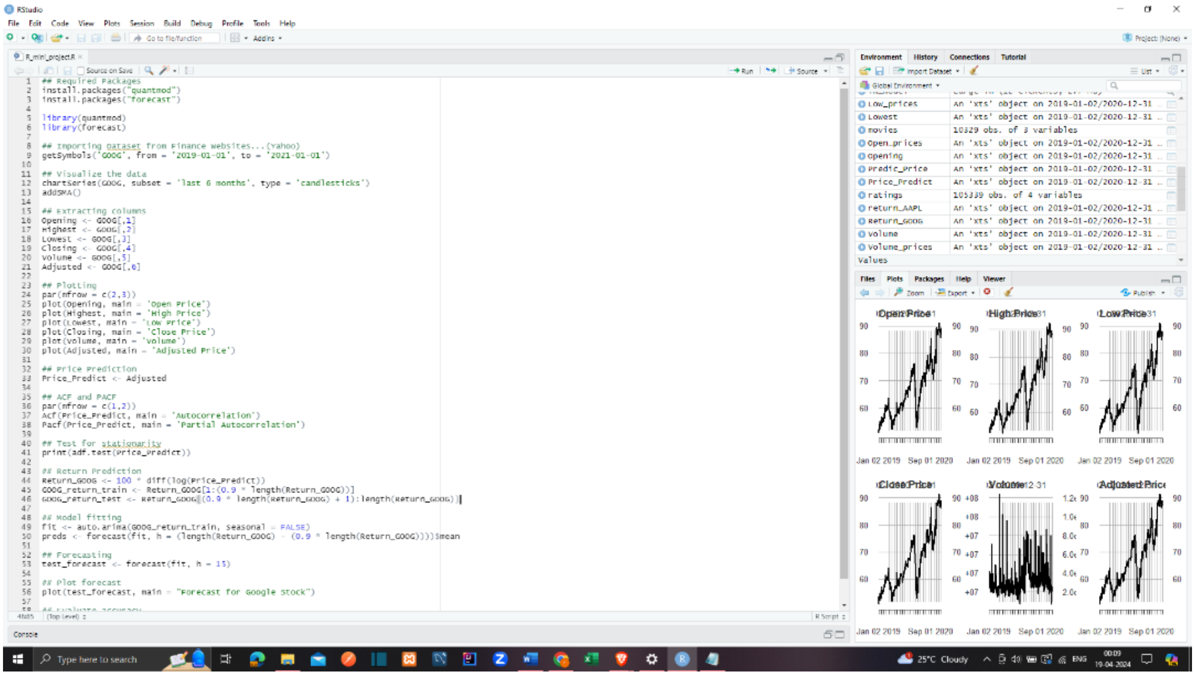
Code Of This Project:

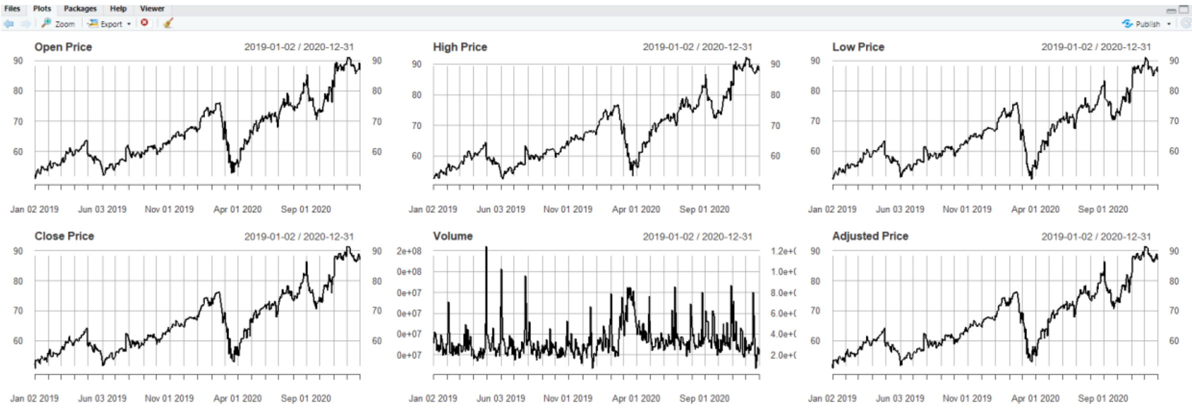
```
## Required Packages
install.packages("quantmod")
install.packages("forecast")

library(quantmod)
library(forecast)

## Importing Dataset from Finance Websites...(Yahoo)
getSymbols('GOOG', from = '2019-01-01', to = '2021-01-01')

## Visualize the data
chartSeries(GOOG, subset = 'last 6 months', type = 'candlesticks')
addSMA()

## Extracting columns
Opening <- GOOG[,1]
Highest <- GOOG[,2]
Lowest <- GOOG[,3]
Closing <- GOOG[,4]
Volume <- GOOG[,5]
Adjusted <- GOOG[,6]
```

```
## Plotting
par(mfrow = c(2,3))
plot(Opening, main = 'Open Price')
plot(Highest, main = 'High Price')
plot(Lowest, main = 'Low Price')
plot(Closing, main = 'Close Price')
plot(Volume, main = 'Volume')
plot(Adjusted, main = 'Adjusted Price')

## Price Prediction
Price_Predict <- Adjusted

## ACF and PACF
par(mfrow = c(1,2))
Acf(Price_Predict, main = 'Autocorrelation')
Pacf(Price_Predict, main = 'Partial Autocorrelation')

## Test for stationarity
print(adf.test(Price_Predict))

## Return Prediction
Return_GOOG <- 100 * diff(log(Price_Predict))
GOOG_return_train <- Return_GOOG[1:(0.9 *
length(Return_GOOG))]
```

```
GOOG_return_test <- Return_GOOG[(0.9 * length(Return_GOOG) +
1):length(Return_GOOG)]

## Model fitting
fit <- auto.arima(GOOG_return_train, seasonal = FALSE)
preds <- forecast(fit, h = (length(Return_GOOG) - (0.9 *
length(Return_GOOG))))$mean

## Forecasting
test_forecast <- forecast(fit, h = 15)

## Plot forecast
plot(test_forecast, main = "Forecast for Google Stock")

## Evaluate accuracy
accuracy(preds, GOOG_return_test)
```

OUTPUT:


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
R_mini_project.R
Required Packages
install.packages("quantmod")
install.packages("forecast")
library(quantmod)
library(forecast)
Importing Dataset from Finance websites...(Yahoo)
getSymbols('GOOG', from = '2019-01-01', to = '2021-01-01')
Visualize the data
chartSeries(GOOG, subset = 'last 6 months', type = 'candlesticks')
addSMA()
Extracting columns
Opening <- GOOG[,1]
Highest <- GOOG[,2]
Lowest <- GOOG[,3]
Closing <- GOOG[,4]
Volume <- GOOG[,5]
Adjusted <- GOOG[,6]
Plotting
par(mfrow = c(2,3))
plot(Opening, main = 'Open Price')
plot(Highest, main = 'High Price')
plot(Lowest, main = 'Low Price')
plot(Closing, main = 'Close Price')
plot(Volume, main = 'Volume')
plot(Adjusted, main = 'Adjusted Price')
Price Prediction
Price_Predict <- Adjusted
ACF and PACF
par(mfrow = c(1,2))
Acf(Price_Predict, main = 'Autocorrelation')
Pacf(Price_Predict, main = 'Partial Autocorrelation')
Test for stationarity
print(adf.test(Price_Predict))
Return Prediction
Return_GOOG <- 100 * diff(log(Price_Predict))
GOOG_return_train <- Return_GOOG[1:(0.9 * length(Return_GOOG))]
GOOG_return_test <- Return_GOOG[(0.9 * length(Return_GOOG) + 1):length(Return_GOOG)]
Model fitting
fit <- auto.arima(GOOG_return_train, seasonal = FALSE)
preds <- forecast(fit, h = (length(Return_GOOG) - (0.9 * length(Return_GOOG))))$mean
Forecasting
test_forecast <- forecast(fit, h = 15)
Plot forecast
plot(test_forecast, main = "Forecast for Google Stock")
Console
Type here to search


Open Price
2019-01-02 / 2020-12-31
High Price
2019-01-02 / 2020-12-31
Low Price
2019-01-02 / 2020-12-31
Close Price
2019-01-02 / 2020-12-31
Volume
2019-01-02 / 2020-12-31
Adjusted Price
2019-01-02 / 2020-12-31
Jan 02 2019 Jun 03 2019 Nov 01 2019 Apr 01 2020 Sep 01 2020