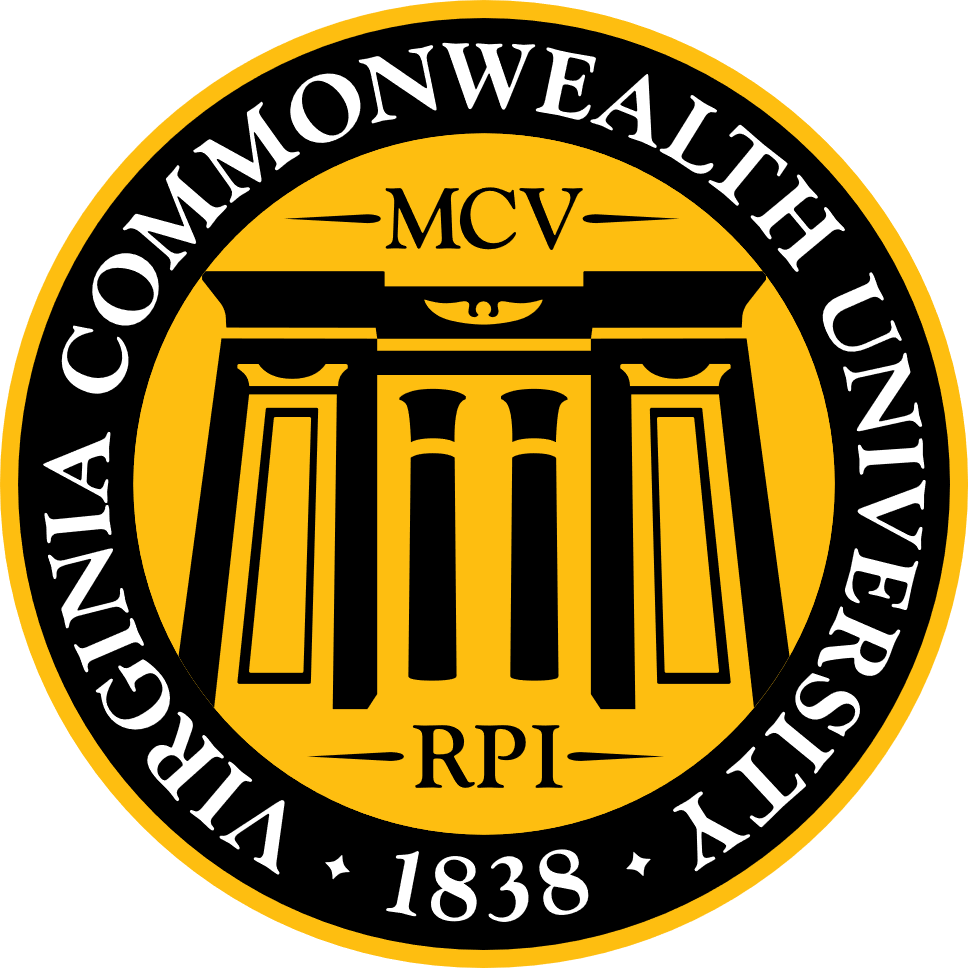
****

**VIRGINIA COMMONWEALTH UNIVERSITY**

**Statistical analysis and modelling (SCMA 632)**

**A6b-** **Time Series Analysis**

**(Part – A)**

**JYOTHIS KANIYAMPARAMBIL THANKACHAN**

**V01110144**

**Date of Submission: 25-07-2024**

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title** | **Page No.** |
| **1.** | Introduction | **1** |
| **2.** | Results | **2** |
| **5.** | Recommendations | **18** |
| **6** | Codes | **19** |
| **7.** | References | **28** |

**INTRODUCTION**

Investors, risk managers, and lawmakers all need to understand how volatile asset returns are in order to make smart decisions. Volatility, which is the change in the prices of assets, shows market risks and how stable investments are. The goal of this task is to use advanced econometric methods, especially the ARCH (Autoregressive Conditional Heteroskedasticity) and GARCH (Generalised Autoregressive Conditional Heteroskedasticity) models, to look at how volatile TESLA Corporation (TSLA) stock is.

TESLA is a major player in the tech industry and is known for its progress in artificial intelligence (AI) and graphics processing units (GPUs). The price of the company has gone up and down a lot, which makes it a great example of volatility research. This study looks at how volatile TSLA stock is to get a better idea of how its price changes and to guess what the future risk levels will be.

The results of this study will help us figure out how risky TESLA's stock is and can help us make business decisions and plan how to handle risk. The study's results will be shown on different graphs, like conditional volatility and predicted variance, which will give a full picture of TSLA's financial volatility.

**OBJECTIVES**

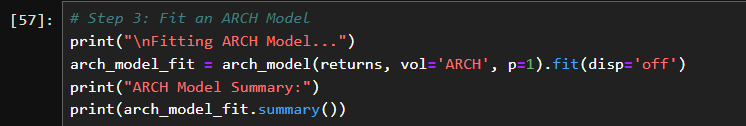
The main goals of this task are to use advanced economic methods to look at how volatile TESLA Corporation (TSLA) stock is and to make predictions about how volatile it will be in the future. In particular, the goals are to look for ARCH/GARCH effects, fit an ARCH/GARCH model, and guess the three-month volatility. Through detailed econometric analysis and projections, this organised method aims to give a full picture of how volatile TESLA's stock is and help people make smart decisions.

**BUSINESS SIGNIFICANCE**

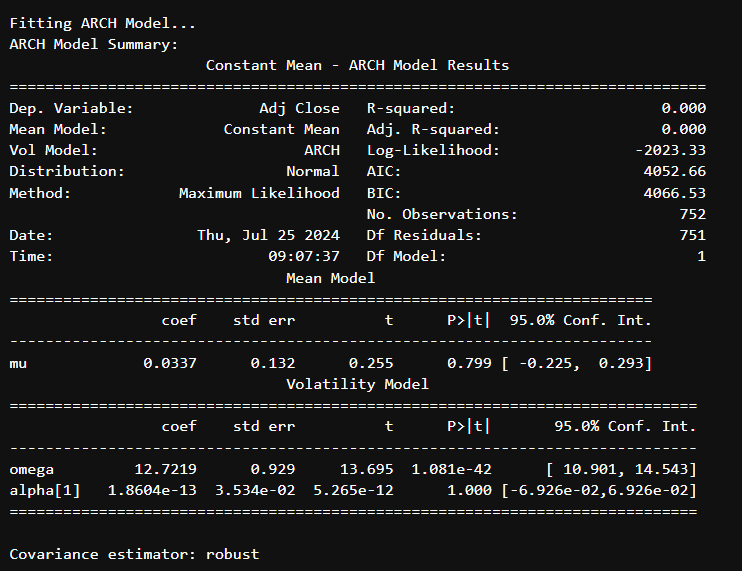
To make smart business choices and handle financial risks, you need to understand and analyse stock volatility. In conclusion, being able to correctly predict and understand stock volatility gives you useful information that can help you make better investment choices, handle financial risks better, and back up smart business actions.

**Result in Python Language**

#### Fit an ARCH Model

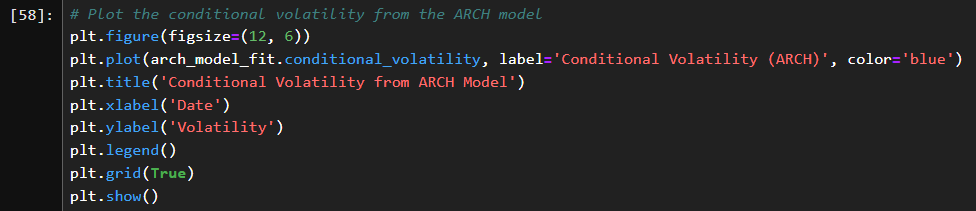
****

**Result**

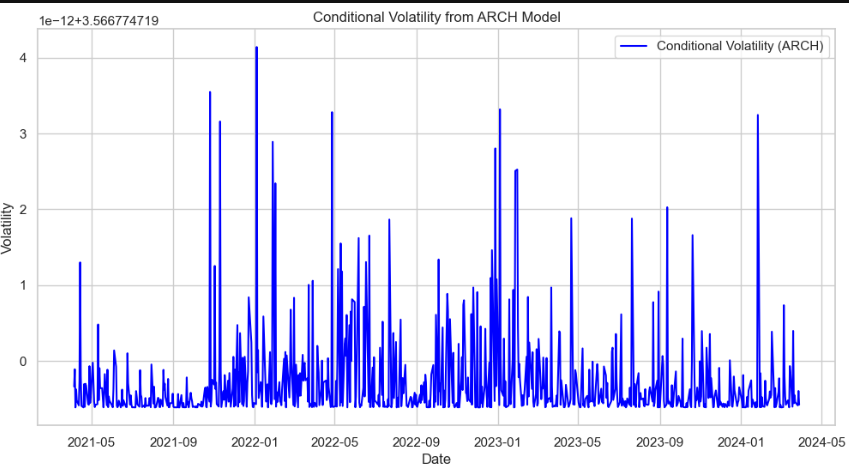


**Interpretation**: The ARCH model helps to identify periods of high and low volatility.

* **Plot Conditional Volatility**:

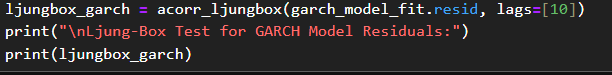


**Result**

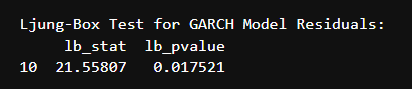


**Interpretation**: Helps in understanding the variability and pattern of volatility over time.

* **Check Residuals for Autocorrelation**:

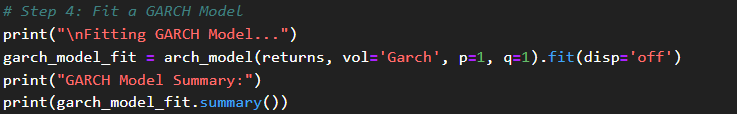
****

**Result**

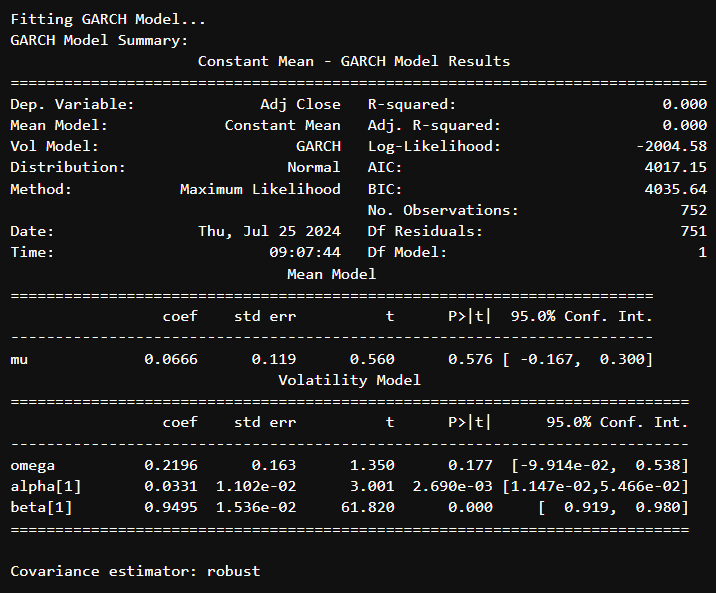
****

**Interpretation**: Significant p-values indicate autocorrelation, suggesting that the model may not fully capture the volatility dynamics.

#### Fit a GARCH Model

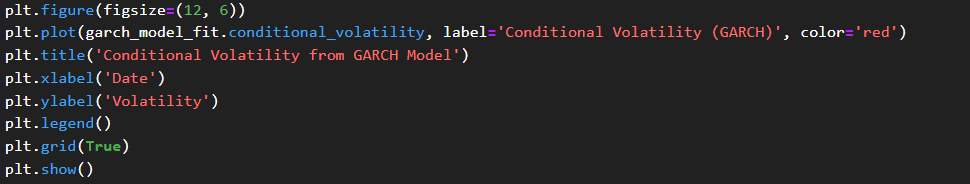
****

**Result**

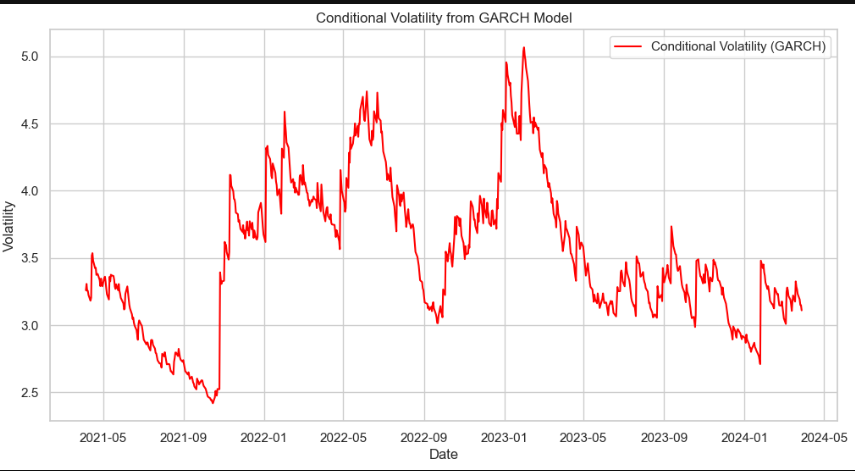


**Interpretation**: The GARCH model accounts for more complex volatility patterns compared to the ARCH model.

* **Plot Conditional Volatility**:

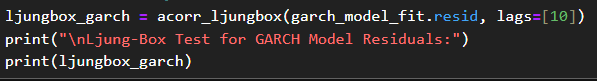
****

**Result**



**Interpretation**: Provides insights into how the GARCH model captures volatility patterns over time.

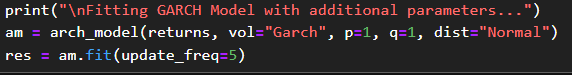
* **Check Residuals for Autocorrelation**:



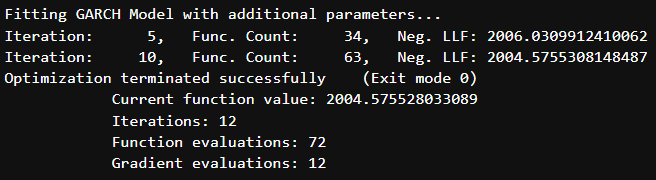
**Interpretation**: Like the ARCH model, significant p-values indicate residual autocorrelation, suggesting the need for further model refinement.



#### Fit GARCH Model with Additional Parameters

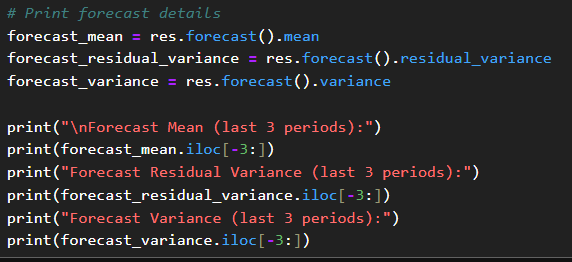
****

**Result**

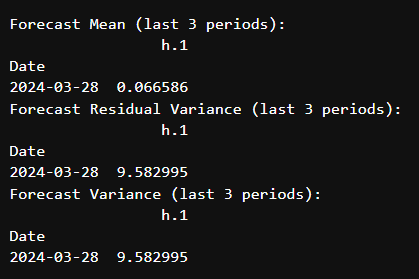


**Interpretation**: The inclusion of additional parameters and distribution assumptions helps in refining the forecast.

* **Print Forecast Details**:



**Result**



**Interpretation**: Indicates the expected future behavior of the stock's volatility.

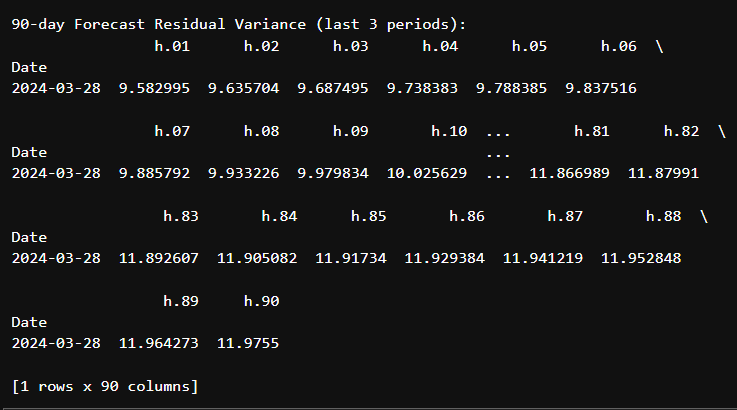
#### Forecasting with a Horizon of 90 Days

****

****

**Result**

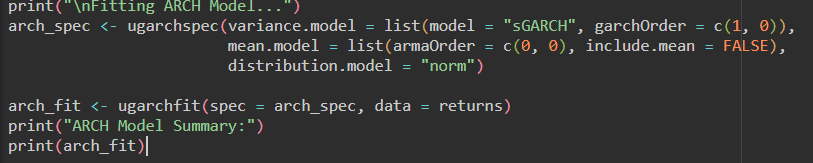




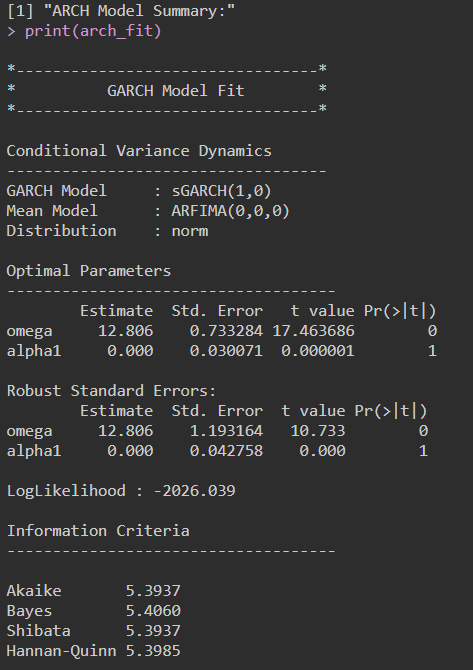
**Interpretation**: Forecasted residual variance provides insights into expected future volatility levels.

**R Language**

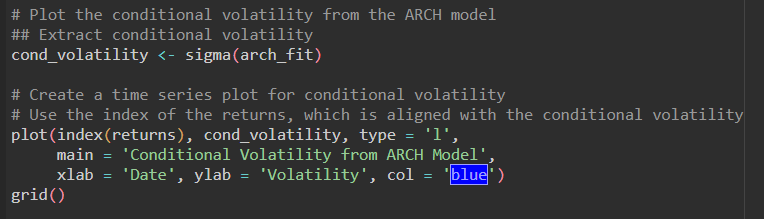
### **Fit an ARCH Model**



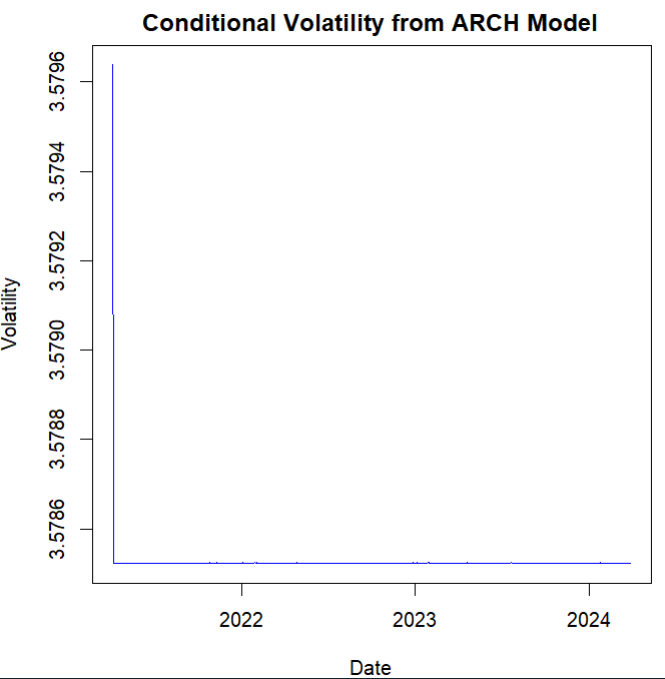
**Result**

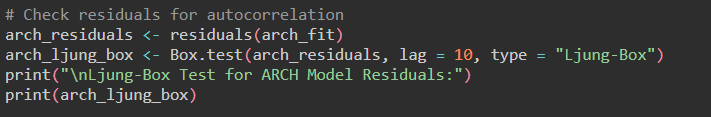


* **Time series plot**

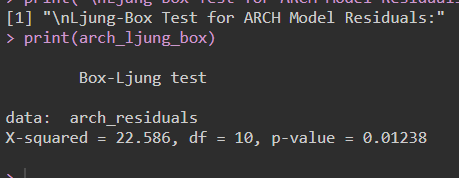
****

**Result**

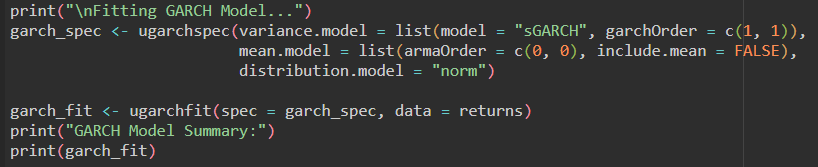


* 

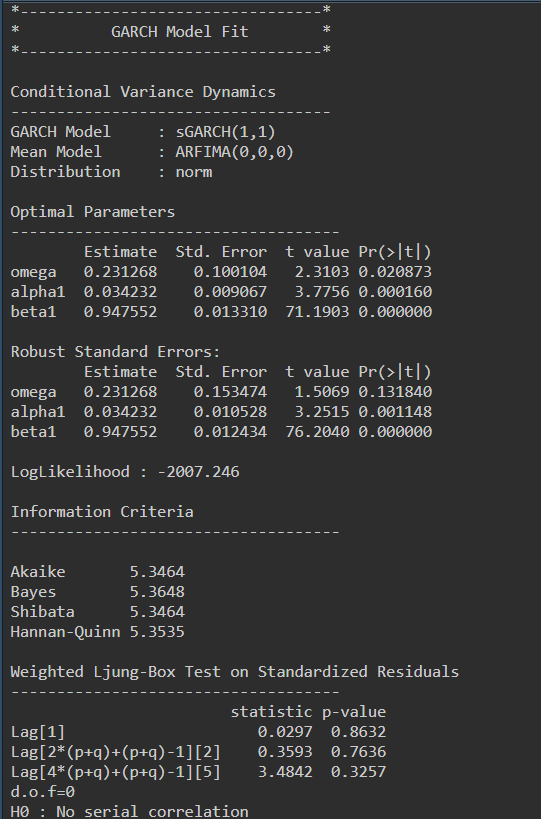
**Result**

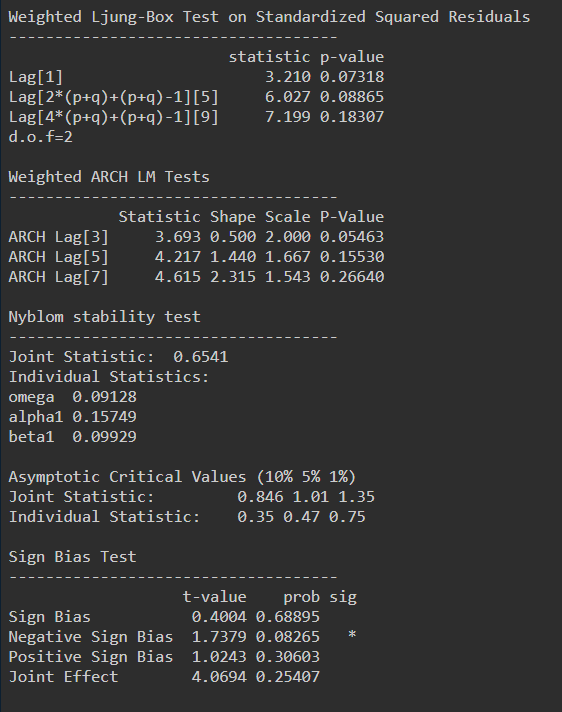


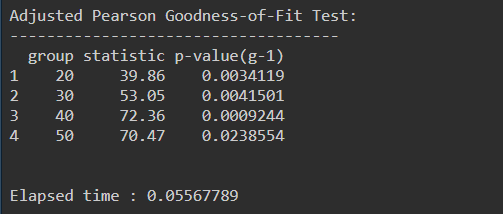
### **Fit a GARCH Model**

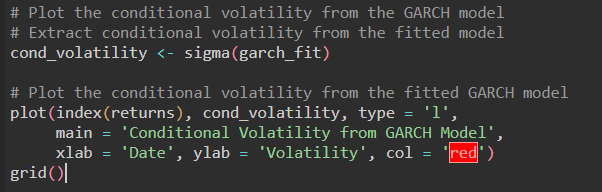


**Result**

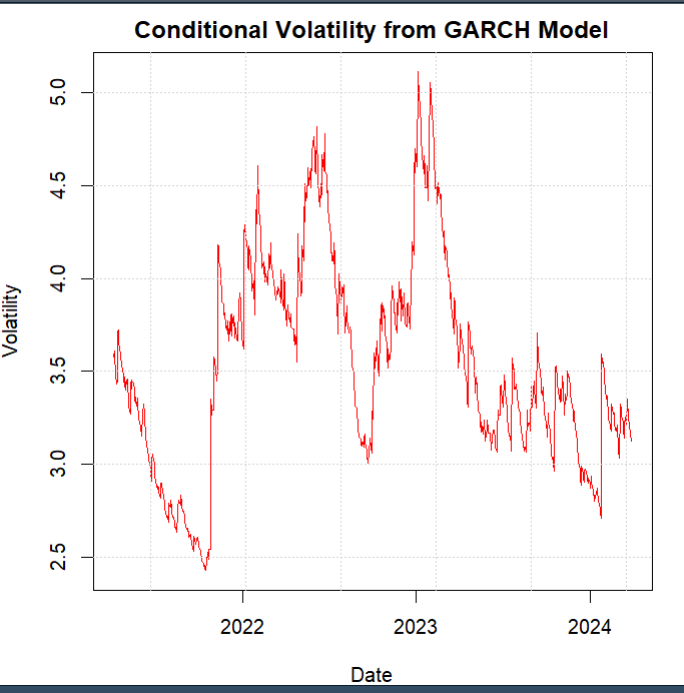
****

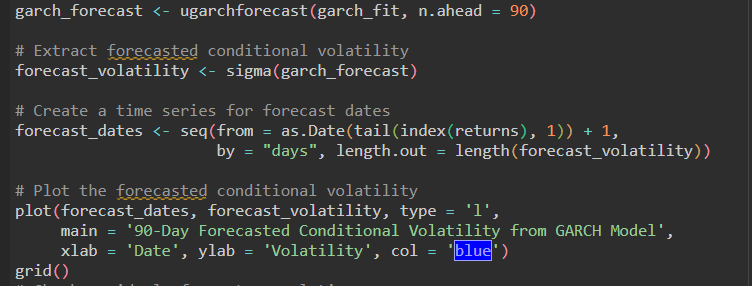




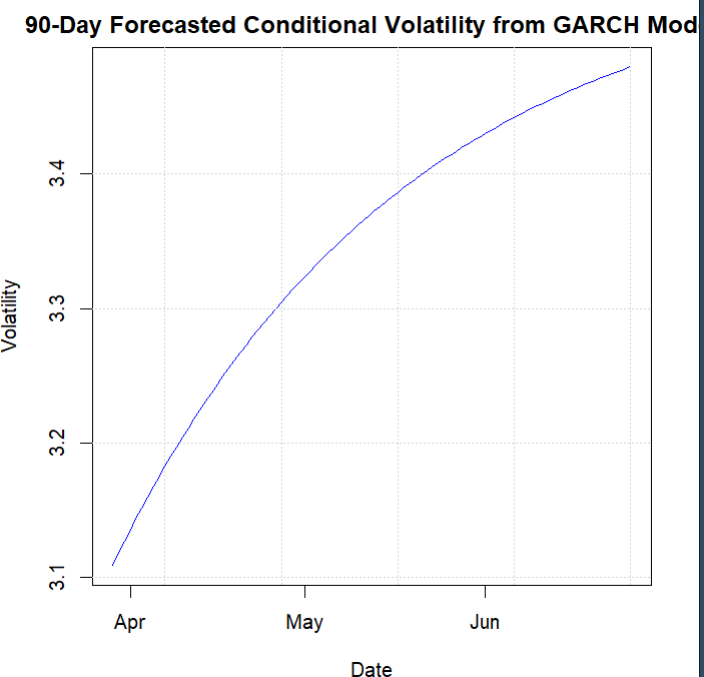
****

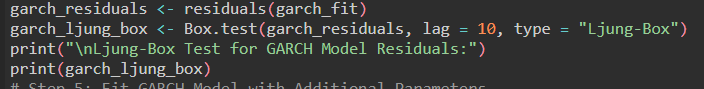
**Result**



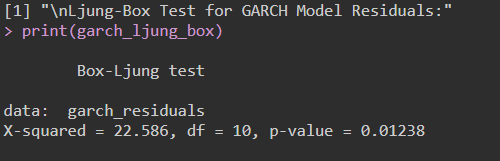
****

**Result**

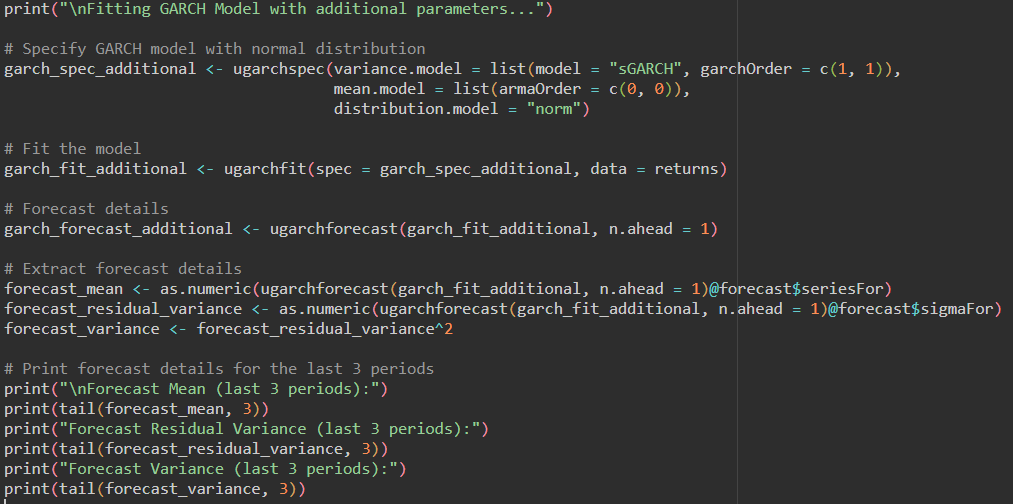
****



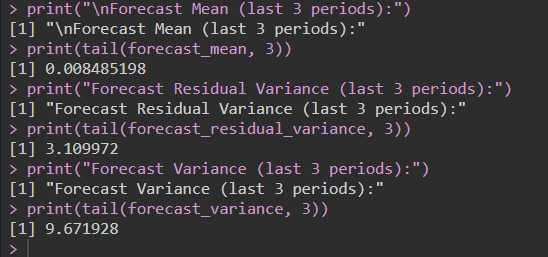
**Result**

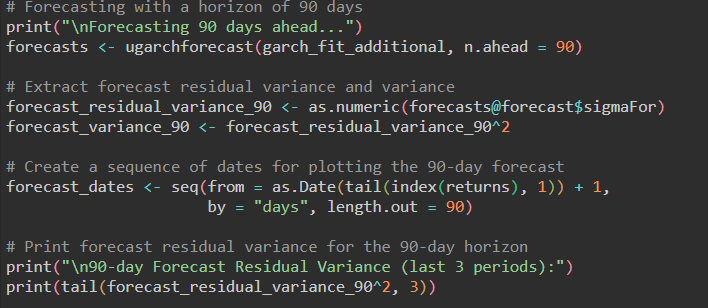


### **Fit GARCH Model with Additional Parameters**

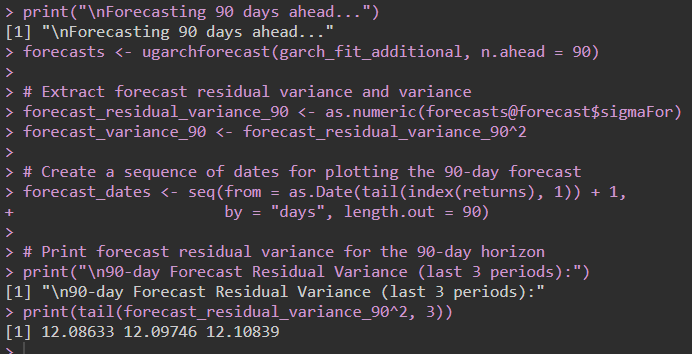


**Result**

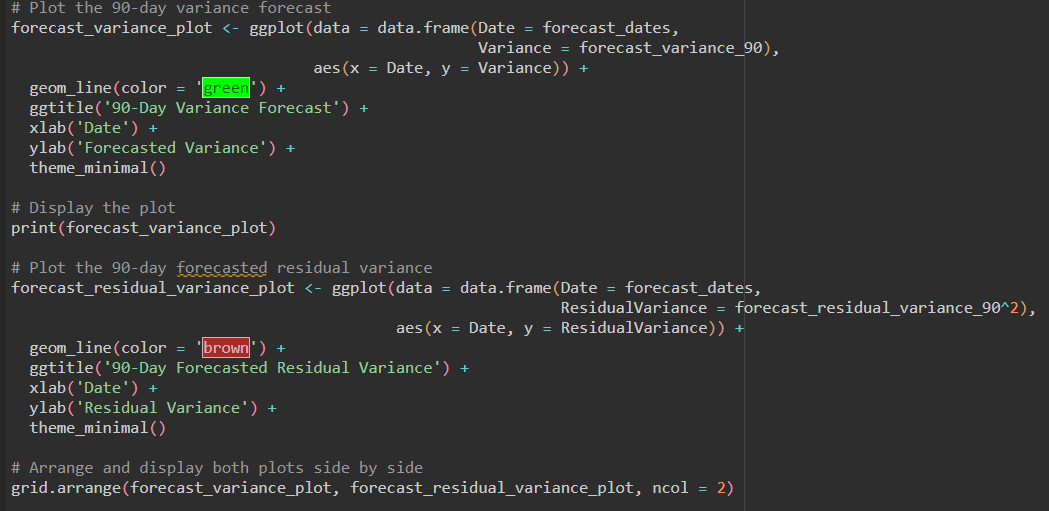


****

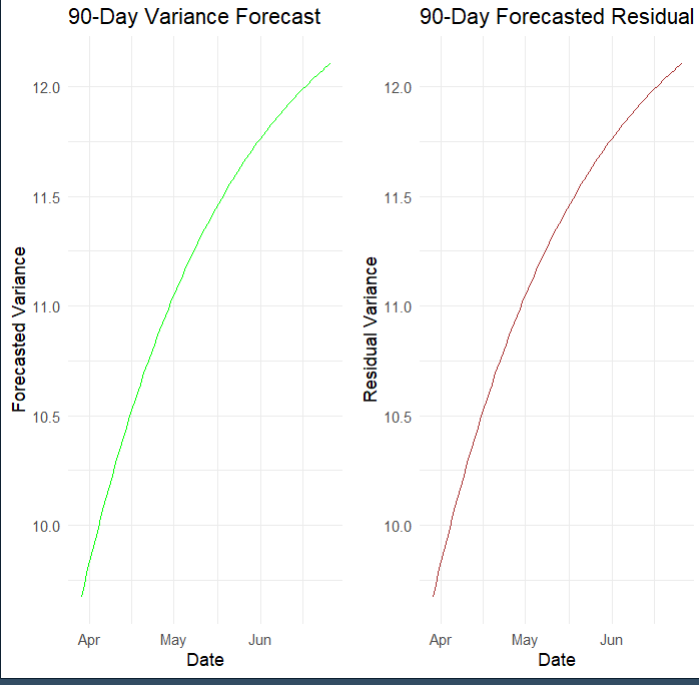
**Result**



### **Plot Forecasts**



**Result**



**RECOMMENDATIONS**

Based on the analysis using ARCH and GARCH models, the following recommendations are provided for effectively utilizing volatility forecasts and improving financial strategies. To use the information from the ARCH and GARCH models, you need to think strategically about risk management, making business decisions, and following the rules. Firms and investors can better handle uncertain markets, make their financial plans work better, and improve total financial security.

**CODES**

**R**

# Install required libraries if not already installed

required\_packages <- c("quantmod", "rugarch", "ggplot2", "tseries", "gridExtra")

new\_packages <- required\_packages[!(required\_packages %in% installed.packages()[,"Package"])]

if(length(new\_packages)) install.packages(new\_packages)

# Load required libraries

library(quantmod)

library(rugarch)

library(ggplot2)

library(tseries)

library(gridExtra)

# Step 1: Download Historical Data of Tesla

ticker <- "TSLA"

getSymbols(ticker, src = "yahoo", from = "2021-04-01", to = "2024-03-31")

# Extract adjusted close price and calculate returns

data <- Ad(get(ticker))

returns <- 100 \* diff(log(data))

returns <- na.omit(returns)

# Check data structure

print(head(TSLA))

print(str(TSLA))

# Step 2: Calculate Returns

market <- Cl(TSLA) # Adjusted Close prices

returns <- 100 \* diff(log(market)) # Convert to percentage returns

returns <- na.omit(returns)

# Step 3: Fit an ARCH Model

print("\nFitting ARCH Model...")

arch\_spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 0)),

mean.model = list(armaOrder = c(0, 0), include.mean = FALSE),

distribution.model = "norm")

arch\_fit <- ugarchfit(spec = arch\_spec, data = returns)

print("ARCH Model Summary:")

print(arch\_fit)

# Plot the conditional volatility from the ARCH model

## Extract conditional volatility

cond\_volatility <- sigma(arch\_fit)

# Create a time series plot for conditional volatility

# Use the index of the returns, which is aligned with the conditional volatility

plot(index(returns), cond\_volatility, type = 'l',

main = 'Conditional Volatility from ARCH Model',

xlab = 'Date', ylab = 'Volatility', col = 'blue')

grid()

# Check residuals for autocorrelation

arch\_residuals <- residuals(arch\_fit)

arch\_ljung\_box <- Box.test(arch\_residuals, lag = 10, type = "Ljung-Box")

print("\nLjung-Box Test for ARCH Model Residuals:")

print(arch\_ljung\_box)

data <- Ad(get(ticker))

returns <- 100 \* diff(log(data))

returns <- na.omit(returns)

# Step 4: Fit a GARCH Model

print("\nFitting GARCH Model...")

garch\_spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 1)),

mean.model = list(armaOrder = c(0, 0), include.mean = FALSE),

distribution.model = "norm")

garch\_fit <- ugarchfit(spec = garch\_spec, data = returns)

print("GARCH Model Summary:")

print(garch\_fit)

# Plot the conditional volatility from the GARCH model

# Extract conditional volatility from the fitted model

cond\_volatility <- sigma(garch\_fit)

# Plot the conditional volatility from the fitted GARCH model

plot(index(returns), cond\_volatility, type = 'l',

main = 'Conditional Volatility from GARCH Model',

xlab = 'Date', ylab = 'Volatility', col = 'red')

grid()

garch\_forecast <- ugarchforecast(garch\_fit, n.ahead = 90)

# Extract forecasted conditional volatility

forecast\_volatility <- sigma(garch\_forecast)

# Create a time series for forecast dates

forecast\_dates <- seq(from = as.Date(tail(index(returns), 1)) + 1,

by = "days", length.out = length(forecast\_volatility))

# Plot the forecasted conditional volatility

plot(forecast\_dates, forecast\_volatility, type = 'l',

main = '90-Day Forecasted Conditional Volatility from GARCH Model',

xlab = 'Date', ylab = 'Volatility', col = 'blue')

grid()

# Check residuals for autocorrelation

garch\_residuals <- residuals(garch\_fit)

garch\_ljung\_box <- Box.test(garch\_residuals, lag = 10, type = "Ljung-Box")

print("\nLjung-Box Test for GARCH Model Residuals:")

print(garch\_ljung\_box)

# Step 5: Fit GARCH Model with Additional Parameters

print("\nFitting GARCH Model with additional parameters...")

# Specify GARCH model with normal distribution

garch\_spec\_additional <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 1)),

mean.model = list(armaOrder = c(0, 0)),

distribution.model = "norm")

# Fit the model

garch\_fit\_additional <- ugarchfit(spec = garch\_spec\_additional, data = returns)

# Forecast details

garch\_forecast\_additional <- ugarchforecast(garch\_fit\_additional, n.ahead = 1)

# Extract forecast details

forecast\_mean <- as.numeric(ugarchforecast(garch\_fit\_additional, n.ahead = 1)@forecast$seriesFor)

forecast\_residual\_variance <- as.numeric(ugarchforecast(garch\_fit\_additional, n.ahead = 1)@forecast$sigmaFor)

forecast\_variance <- forecast\_residual\_variance^2

# Print forecast details for the last 3 periods

print("\nForecast Mean (last 3 periods):")

print(tail(forecast\_mean, 3))

print("Forecast Residual Variance (last 3 periods):")

print(tail(forecast\_residual\_variance, 3))

print("Forecast Variance (last 3 periods):")

print(tail(forecast\_variance, 3))

# Forecasting with a horizon of 90 days

print("\nForecasting 90 days ahead...")

forecasts <- ugarchforecast(garch\_fit\_additional, n.ahead = 90)

# Extract forecast residual variance and variance

forecast\_residual\_variance\_90 <- as.numeric(forecasts@forecast$sigmaFor)

forecast\_variance\_90 <- forecast\_residual\_variance\_90^2

# Create a sequence of dates for plotting the 90-day forecast

forecast\_dates <- seq(from = as.Date(tail(index(returns), 1)) + 1,

by = "days", length.out = 90)

# Print forecast residual variance for the 90-day horizon

print("\n90-day Forecast Residual Variance (last 3 periods):")

print(tail(forecast\_residual\_variance\_90^2, 3))

# Step 6: Plot Forecasts

# Plot the 90-day variance forecast

forecast\_variance\_plot <- ggplot(data = data.frame(Date = forecast\_dates,

Variance = forecast\_variance\_90),

aes(x = Date, y = Variance)) +

geom\_line(color = 'green') +

ggtitle('90-Day Variance Forecast') +

xlab('Date') +

ylab('Forecasted Variance') +

theme\_minimal()

# Display the plot

print(forecast\_variance\_plot)

# Plot the 90-day forecasted residual variance

forecast\_residual\_variance\_plot <- ggplot(data = data.frame(Date = forecast\_dates,

ResidualVariance = forecast\_residual\_variance\_90^2),

aes(x = Date, y = ResidualVariance)) +

geom\_line(color = 'brown') +

ggtitle('90-Day Forecasted Residual Variance') +

xlab('Date') +

ylab('Residual Variance') +

theme\_minimal()

# Arrange and display both plots side by side

grid.arrange(forecast\_variance\_plot, forecast\_residual\_variance\_plot, ncol = 2)

**REFERENCES**

1. <https://www.w3schools.com/>