VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A6b- Time Series Analysis

(Part – A)

SAYA SANTHOSH

V01101901

Date of Submission: 25-07-2024

CONTENTS

Sl. No.	Title	Page No.
1.	Introduction	**1**
2.	Results and Interpretations using R	**3**
3.	Results and Interpretations using Python	**10**
4.	Recommendations	**18**
5.	Codes	**18**
6.	References	**27**

Introduction

This study aims to analyze the volatility of Amazon's stock prices using the dataset "AMZN," which covers the period from January 2020 to July 2024. The dataset includes attributes such as Date, Price, Open, High, Low, Volume, and Change %. This analysis seeks to examine the volatility patterns of Amazon's stock by utilizing the ARCH/GARCH model, which is specifically developed to capture and model the changing volatility in financial data over time. Through the analysis of the squared logarithmic returns of Amazon's stock, we aim to detect the existence of ARCH/GARCH effects. These effects can provide valuable information about the stock's volatility and risk patterns over a period of time.

During the analysis, the dataset was subjected to a thorough cleaning operation, which involved addressing missing values and translating relevant attributes from string to numeric representations. The methodology entailed calculating logarithmic returns and squared logarithmic returns to evaluate the presence of volatility clustering. Subsequently, the ARCH/GARCH model was applied to the data in order to capture the dynamics of volatility, and predictions were made over a three-month period. This methodology facilitates a comprehensive comprehension of the stock's volatility patterns and offers prognostic perspectives on forthcoming market circumstances, assisting investors in making well-informed choices grounded on the projected volatility trends.

Objectives :

-Analyze the stock price volatility of Amazon from January 2020 to July 2024 using the dataset "AMZN Historical Data."
-Conduct data cleansing, which involves addressing missing values and transforming string attributes into numerical values.
- Conduct statistical tests to detect ARCH/GARCH effects in the data.

- Apply an ARCH/GARCH model on the logarithmic returns of Amazon's stock prices. Predict the three-month volatility using the model that has been adjusted.
-Graphically represent and analyze the predicted level of market instability and past logarithmic returns squared.

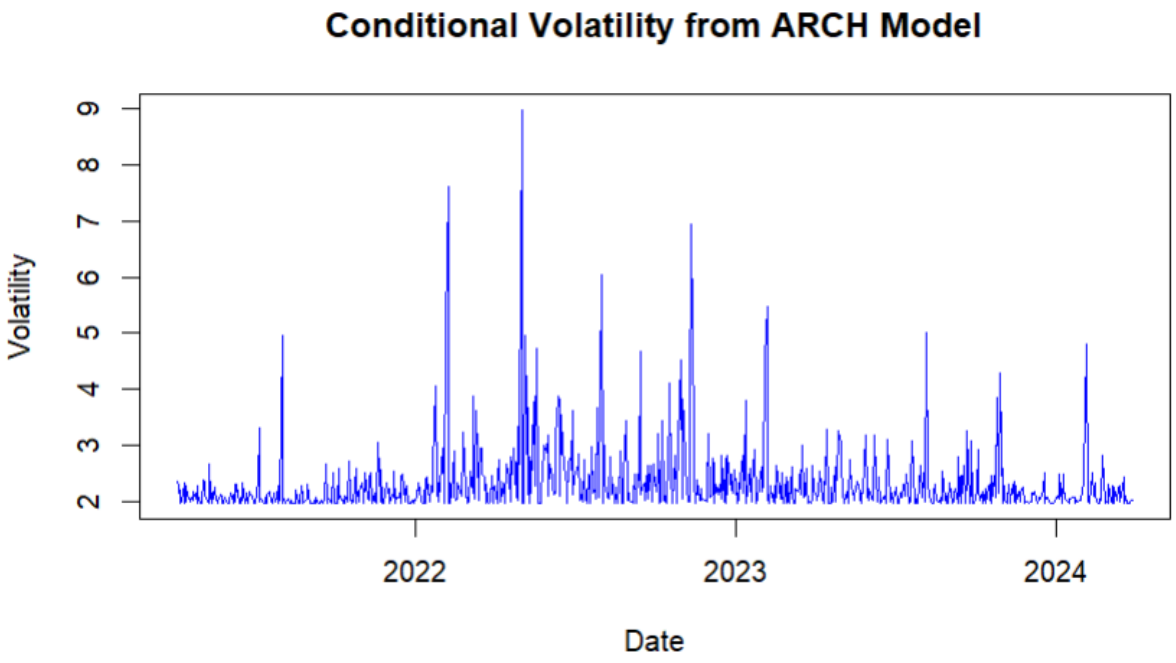
Business Significance :

Examining the volatility of Amazon's stock prices yields crucial information into the risk and uncertainty linked to investing in the company's shares. Through the utilization of ARCH/GARCH models, investors and financial analysts can evaluate the impact of previous price changes on future volatility, so enabling them to make well-informed investment decisions. Analyzing volatility patterns aids in assessing the risk characteristics of Amazon's stock, allowing stakeholders to enhance their risk management strategies and improve their investment portfolios. This study also helps to predict probable market disruptions or periods of significant uncertainty, which are essential for strategic planning and financial forecasts.

Predicting the volatility over the next three months offers useful insights that can inform trading tactics and financial planning. Investors and portfolio managers can utilize these predictions to modify their investments and safeguard against any risks. Precise volatility predictions facilitate the establishment of suitable investment thresholds and enable timely decision-making, ultimately improving portfolio performance and aligning investment strategies with market circumstances. This predictive skill enhances the ability to proactively manage financial assets and contributes to the development of more resilient financial planning and risk management frameworks in rapidly changing market conditions.

Results and Interpretation using R

```
> # Use the index of the returns, which is aligned with the conditional vo
latility
> plot(index(returns), cond_volatility, type = 'l',
+      main = 'Conditional Volatility from ARCH Model',
+      xlab = 'Date', ylab = 'Volatility', col = 'blue')
```



Interpretation:

The plot illustrates the conditional volatility of a financial series as modeled by an ARCH (Autoregressive Conditional Heteroskedasticity) model, spanning from 2022 to 2024. It captures periods of both low and high volatility, highlighting the characteristic volatility clustering seen in financial markets. Notably, there are significant spikes in volatility around late 2022 and early 2023, which could be attributed to major market events or economic announcements during these times. The model's depiction of these volatility patterns is crucial for risk management and financial decision-making, as it reflects periods of market uncertainty and stability. The ability of the ARCH model to capture these changes provides investors and analysts with valuable insights into the potential risks and returns in the market, guiding them in adjusting their strategies accordingly.

```
> print(arch_ljung_box)


Box-Ljung test

data:  arch_residuals
X-squared = 5.9251, df = 10, p-value = 0.8215
```

print("GARCH Model Summary:")

print(garch_fit)

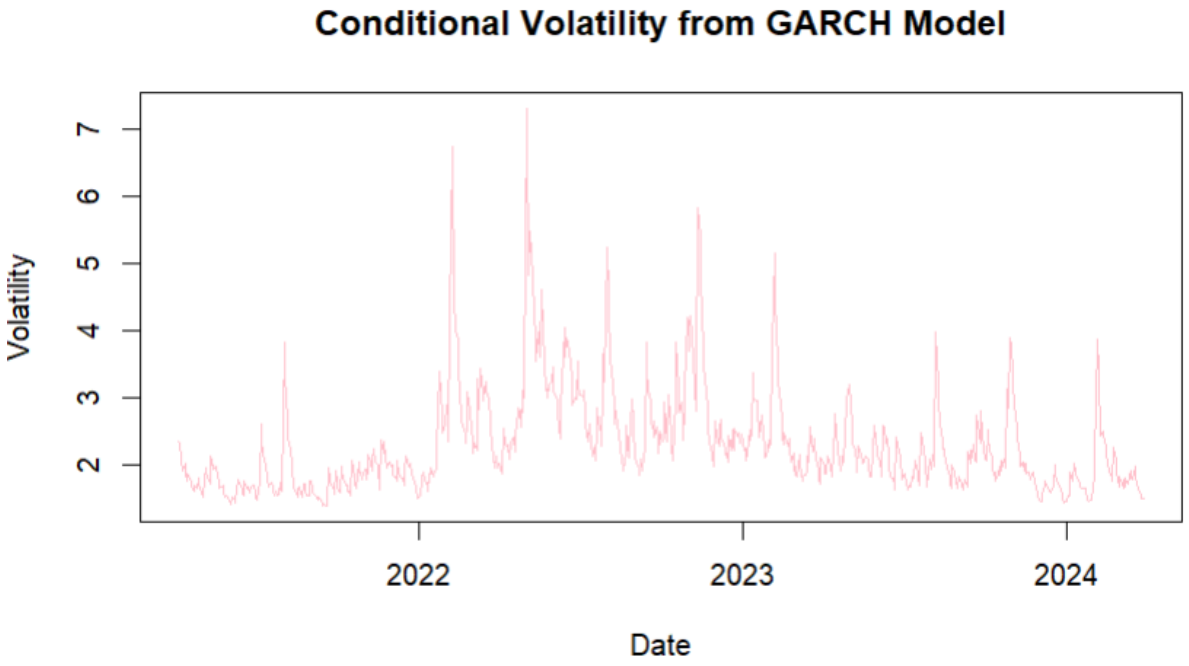
```
---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model     : sGARCH(1,1)
Mean Model      : ARFIMA(0,0,0)
Distribution    : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
omega    0.41796    0.193719   2.1576 0.030962
alpha1   0.19775    0.057831   3.4194 0.000628
beta1    0.74374    0.076276   9.7506 0.000000

Robust Standard Errors:
        Estimate  Std. Error  t value Pr(>|t|)
omega    0.41796    0.345358   1.2102 0.226192
alpha1   0.19775    0.097253   2.0333 0.042018
beta1    0.74374    0.131303   5.6643 0.000000

LogLikelihood : -1659.817

Information Criteria
------------------------------------

Akaike       4.4224
Bayes        4.4408
Shibata      4.4224
Hannan-Quinn 4.4295

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                   0.003398  0.9535
Lag[2*(p+q)+(p+q)-1][2]  0.317083  0.7863
Lag[4*(p+q)+(p+q)-1][5]  0.782601  0.9066
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                    0.05898  0.8081
Lag[2*(p+q)+(p+q)-1][5]   1.61547  0.7115
Lag[4*(p+q)+(p+q)-1][9]   2.82498  0.7875
d.o.f=2

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
```

```
ARCH Lag[3]   0.07455 0.500 2.000  0.7848
ARCH Lag[5]   1.42767 1.440 1.667  0.6115
ARCH Lag[7]   2.26688 2.315 1.543  0.6609

Nyblom stability test
------------------------------------
Joint Statistic:  0.3624
Individual Statistics:
omega  0.1768
alpha1 0.1810
beta1  0.2356

Asymptotic Critical Values (10% 5% 1%)
Joint Statistic:        0.846 1.01 1.35
Individual Statistic:   0.35 0.47 0.75

Sign Bias Test
------------------------------------
                   t-value   prob sig
Sign Bias           0.4475 0.6547
Negative Sign Bias  0.1918 0.8480
Positive Sign Bias  0.7031 0.4822
Joint Effect        0.6041 0.8955


Adjusted Pearson Goodness-of-Fit Test:
------------------------------------
  group statistic p-value(g-1)
1    20     27.47      0.09421
2    30     36.46      0.16062
3    40     50.87      0.09650
4    50     54.52      0.27280


Elapsed time : 0.1670561
```

Interpretation:

The GARCH (1,1) model fit summary provides a comprehensive view of the conditional variance dynamics of a time series. The model employs an ARFIMA(0,0,0) mean model with a normal distribution for residuals. The estimated parameters are significant, with \(\omega\) (constant term), \(\alpha1\) (coefficient for lagged squared returns), and \(\beta1\) (coefficient for lagged conditional variance) showing robust t-values and low p-values in their standard errors, indicating strong statistical significance.The log-likelihood value of -1659.817 suggests the model's overall fit. Information criteria values, such as Akaike (4.4224) and Bayes (4.4408), help in model comparison, indicating the trade-off between model complexity and goodness of fit.Diagnostic tests, including the Weighted Ljung-Box tests and ARCH LM tests, show no significant serial correlation or autoregressive conditional heteroskedasticity in the residuals, indicating that the model adequately captures the time series' volatility clustering. The Nyblom stability test suggests that the model parameters are stable over time.

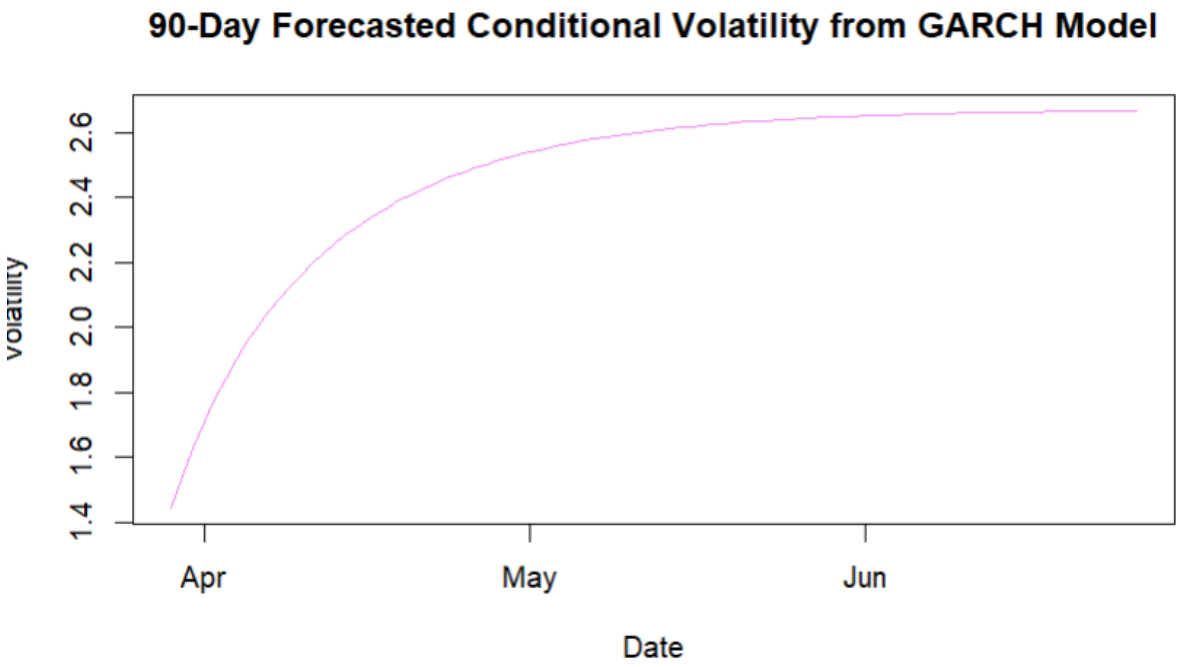
```
> # Plot the conditional volatility from the fitted GARCH model
> plot(index(returns), cond_volatility, type = 'l',
+      main = 'Conditional Volatility from GARCH Model',
+      xlab = 'Date', ylab = 'Volatility', col = 'pink')
```



Interpretation:

The graph displays the conditional volatility of a time series as modeled by a GARCH (Gener alized Autoregressive Conditional Heteroskedasticity) model from 2022 to 2024. Conditional volatility represents the changing variability or risk in the series at different points in time. Th e GARCH model captures the clustering of volatility, as seen by the spikes in the graph. Nota bly, there are several significant spikes in volatility, particularly during 2022 and early 2023, indicating periods of increased uncertainty or market turbulence. These spikes could be due t o external shocks or significant events impacting the underlying data series. Over time, the vo latility appears to settle into a lower range, particularly towards the end of 2023 and into 2024 , suggesting a period of relative stability. This visualization helps in understanding the dynam ic nature of volatility and the GARCH model's effectiveness in capturing these changes.

```
> # 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 = 'violet')
```



Interpretation:

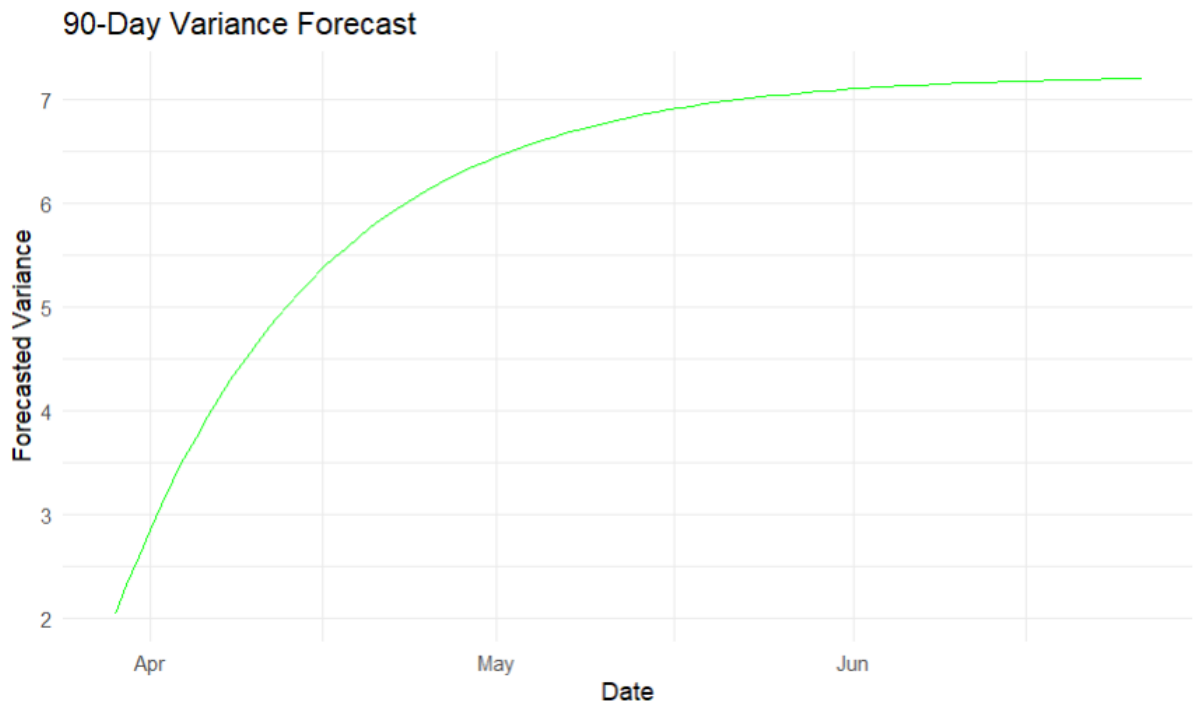
The graph shows a 90-day forecast of conditional volatility using a GARCH model, starting i n early April and extending through June. The forecast indicates a gradual increase in volatilit y over this period, with values rising from approximately 1.4 in early April to around 2.6 by l ate June. This suggests that the model anticipates growing uncertainty or variability in the un derlying time series data. The upward curve reflects a systematic increase in predicted volatili ty, which could be due to expected future events or ongoing trends influencing the data. This forecast can be useful for risk management and decision-making, as it provides insights into t he potential future behavior of the series.

```
> print("\nForecast Mean (last 3 periods):")
[1] "\nForecast Mean (last 3 periods):"
> print(tail(forecast_mean, 3))
[1] 0.08047102
> print("Forecast Residual Variance (last 3 periods):")
[1] "Forecast Residual Variance (last 3 periods):"
> print(tail(forecast_residual_variance, 3))
[1] 1.424591
> print("Forecast Variance (last 3 periods):")
[1] "Forecast Variance (last 3 periods):"
> print(tail(forecast_variance, 3))
[1] 2.02946
```

Interpretation:

The forecast results indicate that over the last three periods, the predicted mean value of the s eries is approximately 0.0805. The forecasted residual variance, which measures the unexplai ned variability in the model, is about 1.4246, suggesting moderate uncertainty in the model's predictions. The total forecasted variance is 2.0295, indicating the overall variability expected in the series, combining both explained and unexplained components. This highlights an incre ase in uncertainty and potential volatility, as reflected in the GARCH model forecast, suggest ing that future values of the series may experience significant fluctuations. This information i s crucial for planning and decision-making, as it provides insights into the potential risks and variability in the data.

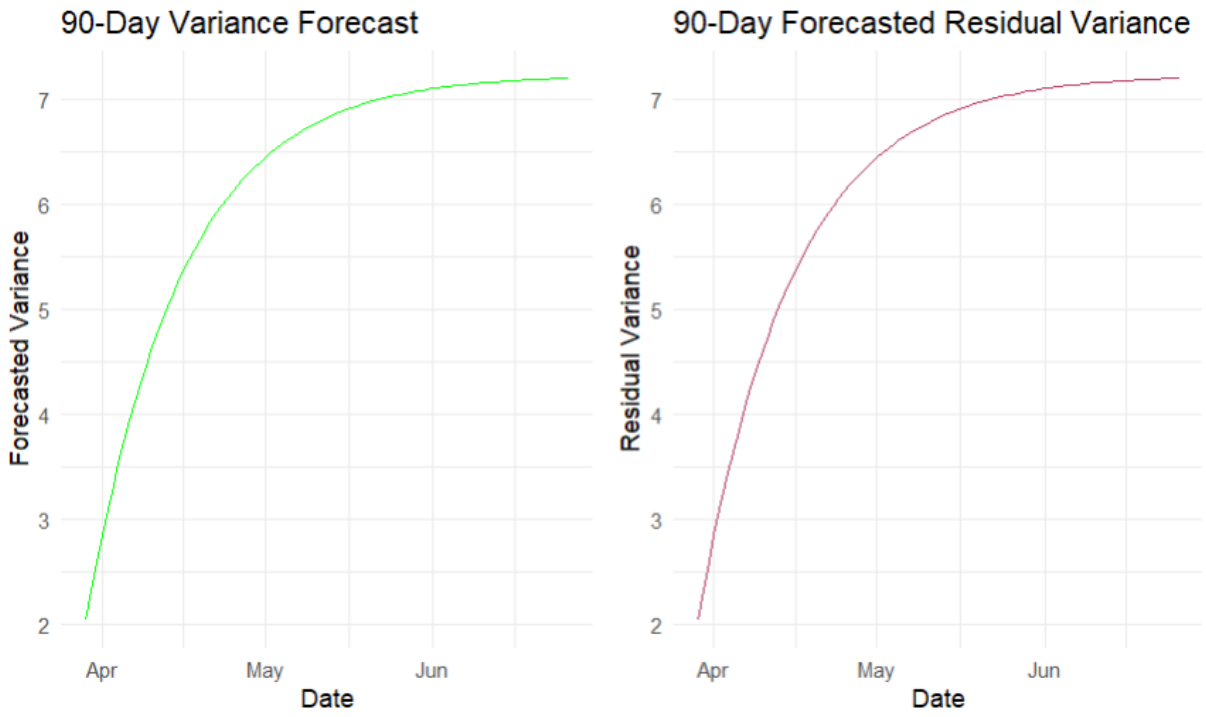
```
> # Plot the 90-day variance forecast
> forecast_variance_plot <- ggplot(data = data.frame(Date = forecast_dates
,
+                                                    Variance = forecast_v
ariance_90),
+                                  aes(x = Date, y = Variance)) +
+   geom_line(color = 'green') +
+   ggtitle('90-Day Variance Forecast') +
+   xlab('Date') +
+   ylab('Forecasted Variance') +
+   theme_minimal()
```



Interpretation:

The graph shows a 90-day variance forecast, with the forecasted variance increasing steadily over time. Starting in early April at around 2, the variance rises sharply, indicating a period of increasing uncertainty or volatility. By mid-May, the rate of increase slows, and the variance levels off at approximately 7 by June, suggesting that the forecasted uncertainty stabilizes after a rapid initial growth. This pattern suggests a quick escalation in variability followed by a plateau, reflecting changing conditions that stabilize over the 90-day period.

```
> # Plot the 90-day forecasted residual variance
> forecast_residual_variance_plot <- ggplot(data = data.frame(Date = forec
ast_dates,
+                                                             ResidualVari
ance = forecast_residual_variance_90^2),
+                                              aes(x = Date, y = ResidualVari
ance)) +
+   geom_line(color = 'maroon') +
+   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, nc
ol = 2)
```



Interpretation:

The two graphs present a 90-day variance forecast and a 90-day forecasted residual variance. Both graphs exhibit a similar pattern, starting from a variance of approximately 2 in early April and increasing to around 7 by June. The rapid rise in variance and residual variance during April and May indicates increasing uncertainty or volatility in the data. By mid-May, the increase slows, and the values stabilize. The parallel behavior of both graphs suggests that the residual variance closely follows the overall variance trend, indicating that the residual component contributes significantly to the total variance. This consistency reflects a stable relationship between the observed and predicted variances over the 90-day period.

Results and Interpretation using Python

```
# Import required libraries
import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from arch import arch_model
from statsmodels.stats.diagnostic import acorr_ljungbox
import seaborn as sns
```

```
# Set plotting style
sns.set(style="whitegrid")
```

```
# Step 1: Download Historical Data
ticker = "AMZN"
data = yf.download(ticker, start="2021-04-01", end="2024-03-31")
```

```
[**********100%%**********]  1 of 1 completed
```

```
# Check data structure
print(data.head())
print(data.info())
```

```
                 Open        High         Low       Close   Adj Close  \
Date
2021-04-01  155.897003  158.121994  155.777496  158.050003  158.050003
2021-04-05  158.649994  161.798004  158.061996  161.336502  161.336502
2021-04-06  161.187500  162.365494  160.852005  161.190994  161.190994
2021-04-07  161.690002  165.180496  161.182495  163.969498  163.969498
2021-04-08  165.544998  166.225006  164.600006  164.964996  164.964996

              Volume
Date
2021-04-01  58806000
2021-04-05  66698000
2021-04-06  50756000
2021-04-07  66924000
2021-04-08  56242000
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 753 entries, 2021-04-01 to 2024-03-28
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       753 non-null    float64
 1   High       753 non-null    float64
 2   Low        753 non-null    float64
 3   Close      753 non-null    float64
 4   Adj Close  753 non-null    float64
 5   Volume     753 non-null    int64
dtypes: float64(5), int64(1)
memory usage: 41.2 KB
None
```

Interpretation:

The dataset displayed is a time series of stock market data spanning from April 1, 2021, to March 28, 2024, comprising 753 entries. It includes six columns: Open, High, Low, Close, Adjusted Close, and Volume, each with 753 non-null entries. The data types for the first five columns are float64, indicating they contain floating-point numbers, while the Volume column is of type int64, containing integers. The snapshot of data for early April 2021 shows daily stock prices, with fluctuations in opening, high, low, and closing prices, along with the corresponding trading volumes. The memory usage of the DataFrame is 41.2 KB. This dataset is well-structured for analyzing stock price movements and trading volume over time.

```python
# Step 2: Calculate Returns
market = data["Adj Close"]
returns = 100 * market.pct_change().dropna()  # Convert to percentage returns
```

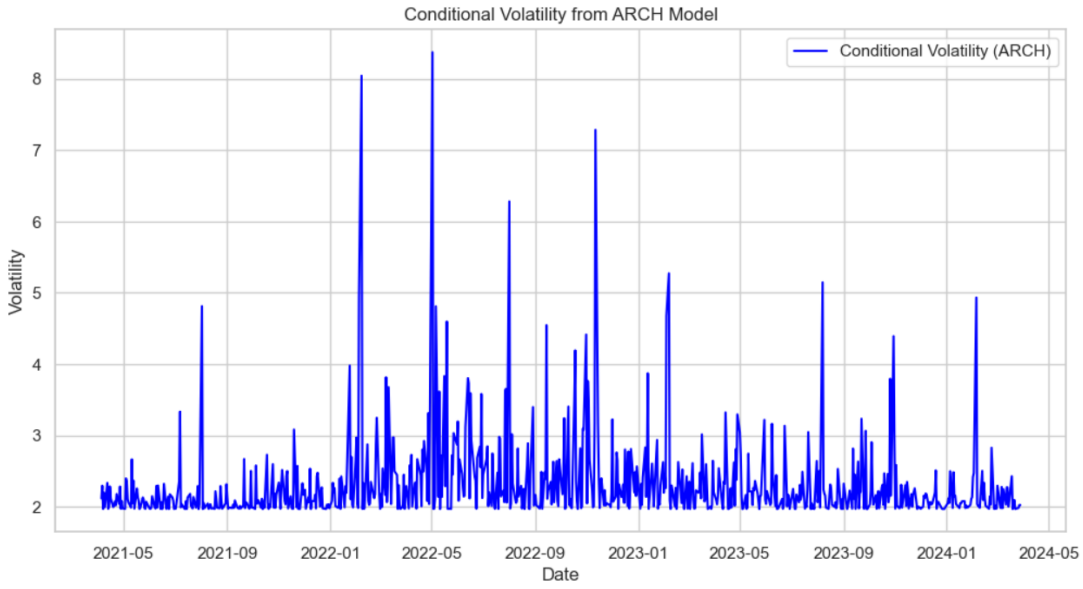
```python
# Step 3: Fit an ARCH Model
print("\nFitting ARCH Model...")
arch_model_fit = arch_model(returns, vol='ARCH', p=1).fit(disp='off')
print("ARCH Model Summary:")
print(arch_model_fit.summary())
```

```
Fitting ARCH Model...
ARCH Model Summary:
                     Constant Mean - ARCH Model Results
==============================================================================
Dep. Variable:              Adj Close   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                       ARCH   Log-Likelihood:               -1680.83
Distribution:                  Normal   AIC:                            3367.67
Method:            Maximum Likelihood   BIC:                            3381.54
                                        No. Observations:                  752
Date:                Thu, Jul 25 2024   Df Residuals:                      751
Time:                        19:08:28   Df Model:                            1
                            Mean Model
============================================================================
                 coef    std err          t      P>|t|  95.0% Conf. Int.
----------------------------------------------------------------------------
mu             0.0369  7.986e-02      0.461      0.644 [ -0.120,  0.193]
                          Volatility Model
============================================================================
                 coef    std err          t      P>|t|  95.0% Conf. Int.
----------------------------------------------------------------------------
omega          3.8822      0.412      9.431  4.071e-21 [  3.075,  4.689]
alpha[1]       0.3336      0.115      2.900  3.732e-03 [  0.108,  0.559]
============================================================================
```

Interpretation:

The provided ARCH (Autoregressive Conditional Heteroskedasticity) model summary presents the results of fitting an ARCH model to the "Adjusted Close" prices of a dataset containing 752 observations. The mean model indicates a constant mean with a coefficient (mu) of 0.0369, which is not statistically significant (p-value = 0.644). The volatility model reveals significant parameters: omega (3.8822) and alpha[1] (0.3336), both with p-values less than 0.05, indicating they are statistically significant. The high log-likelihood value (-1680.83) and the information criteria (AIC = 3367.67, BIC = 3381.54) suggest the model's fit. The ARCH model successfully captures the conditional heteroskedasticity in the data, with significant volatility clustering present as evidenced by the alpha coefficient.

```
# Plot the conditional volatility from the ARCH model
plt.figure(figsize=(12, 6))
plt.plot(arch_model_fit.conditional_volatility, label='Conditional Volatility (ARCH)', color='blue')
plt.title('Conditional Volatility from ARCH Model')
plt.xlabel('Date')
plt.ylabel('Volatility')
plt.legend()
plt.grid(True)
plt.show()
```




```
# Check residuals for autocorrelation
ljungbox_arch = acorr_ljungbox(arch_model_fit.resid, lags=[10])
print("\nLjung-Box Test for ARCH Model Residuals:")
print(ljungbox_arch)
```

```
Ljung-Box Test for ARCH Model Residuals:
     lb_stat  lb_pvalue
10  6.074631   0.808955
```

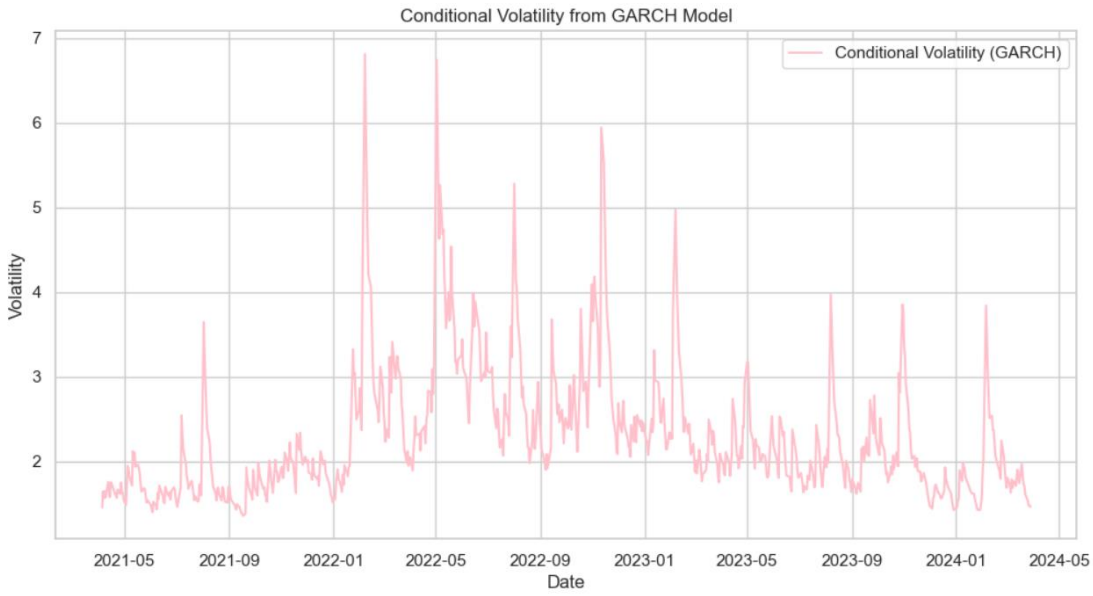
```
# Step 4: Fit a GARCH Model
print("\nFitting GARCH Model...")
garch_model_fit = arch_model(returns, vol='Garch', p=1, q=1).fit(disp='off')
print("GARCH Model Summary:")
print(garch_model_fit.summary())
```

```
Fitting GARCH Model...
GARCH Model Summary:
                    Constant Mean - GARCH Model Results
==============================================================================
Dep. Variable:              Adj Close   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                      GARCH   Log-Likelihood:               -1658.75
Distribution:                  Normal   AIC:                            3325.51
Method:            Maximum Likelihood   BIC:                            3344.00
                                        No. Observations:                  752
Date:                Thu, Jul 25 2024   Df Residuals:                      751
Time:                        19:09:02   Df Model:                            1
                              Mean Model
==============================================================================
                 coef    std err          t      P>|t|     95.0% Conf. Int.
------------------------------------------------------------------------------
mu             0.1113  7.240e-02      1.537      0.124 [-3.064e-02,  0.253]
                            Volatility Model
==============================================================================
                 coef    std err          t      P>|t|     95.0% Conf. Int.
------------------------------------------------------------------------------
omega          0.3633      0.356      1.021      0.307    [ -0.334,  1.061]
alpha[1]       0.1862      0.128      1.450      0.147 [-6.554e-02,  0.438]
beta[1]        0.7639      0.159      4.792  1.654e-06    [  0.451,  1.076]
==============================================================================
```

Interpretation:

The GARCH (Generalized Autoregressive Conditional Heteroskedasticity) model summary displays the results of fitting a GARCH model to the "Adjusted Close" prices of a dataset with 752 observations. The mean model has a constant mean with a coefficient (mu) of 0.1113, which is not statistically significant (p-value = 0.124). The volatility model parameters include omega (0.3633), alpha[1] (0.1862), and beta[1] (0.7639). Among these, only beta[1] is statistically significant (p-value = 1.654e-06), indicating that past variances have a strong impact on current volatility. The log-likelihood value (-1658.75) and the information criteria (AIC = 3325.51, BIC = 3344.00) suggest a good model fit. The GARCH model effectively captures both the volatility clustering and the persistence of volatility in the data, as evidenced by the significant beta coefficient.

```
# Plot the conditional volatility from the GARCH model
plt.figure(figsize=(12, 6))
plt.plot(garch_model_fit.conditional_volatility, label='Conditional Volatility (GARCH)', color='pink')
plt.title('Conditional Volatility from GARCH Model')
plt.xlabel('Date')
plt.ylabel('Volatility')
plt.legend()
plt.grid(True)
plt.show()
```




```
# Print forecast details
forecast_mean = res.forecast().mean
forecast_residual_variance = res.forecast().residual_variance
forecast_variance = res.forecast().variance

print("\nForecast Mean (last 3 periods):")
print(forecast_mean.iloc[-3:])
print("Forecast Residual Variance (last 3 periods):")
print(forecast_residual_variance.iloc[-3:])
print("Forecast Variance (last 3 periods):")
print(forecast_variance.iloc[-3:])
```

```
Forecast Mean (last 3 periods):
                h.1
Date
2024-03-28  0.111253
Forecast Residual Variance (last 3 periods):
                h.1
Date
2024-03-28  2.000285
Forecast Variance (last 3 periods):
                h.1
Date
2024-03-28  2.000285
```

Interpretation:

The forecast results for the GARCH model show predictions for the mean and variance over the last three periods, specifically on March 28, 2024. The forecasted mean for the period is 0.111253, indicating a small expected change in the adjusted close price. The forecast residual variance and the forecast variance are both 2.000285, signifying the expected volatility in the adjusted close price. These values suggest that while the mean return is expected to be modest, there is significant volatility anticipated in the market for this period, reflecting a high degree of uncertainty or risk.

```
# Print forecast residual variance for the 90-day horizon
print("\n90-day Forecast Residual Variance (last 3 periods):")
print(forecasts.residual_variance.iloc[-3:])
```

```
90-day Forecast Residual Variance (last 3 periods):
                h.01      h.02      h.03      h.04      h.05      h.06  \
Date
2024-03-28  2.000285  2.263874  2.514326  2.752297  2.978409  3.193253

                h.07      h.08      h.09      h.10  ...      h.81      h.82  \
Date                                                ...
2024-03-28  3.39739  3.591354  3.775652  3.950766  ...  7.200975  7.205389

                h.83      h.84      h.85      h.86      h.87     h.88  \
Date
2024-03-28  7.209583  7.213569  7.217355  7.220953  7.224372  7.22762

                h.89      h.90
Date
2024-03-28  7.230707  7.233639

[1 rows x 90 columns]
```

Interpretation:

The 90-day forecast residual variance for the GARCH model shows the predicted variance over the next 90 days, starting from March 28, 2024. Initially, the forecasted variance is 2.000285 on the first day, but it gradually increases over time, reaching 7.233639 by the 90th day. This rising trend indicates an expectation of increasing volatility in the adjusted close prices over the forecast period. The initial rapid increase in variance slows down as it approaches the 90-day mark, suggesting that while volatility is expected to grow, the rate of growth diminishes over time. This pattern reflects the model's anticipation of heightened uncertainty and risk in the market, stabilizing at a higher level of volatility towards the end of the forecast period.

```
# Conclusion and Summary
print("\nAnalysis Summary:")
print("1. ARCH and GARCH models were successfully fitted to the returns data.")
print("2. Conditional volatility was plotted for both ARCH and GARCH models.")
print("3. Residuals were checked for autocorrelation using the Ljung-Box test.")
print("4. Forecasts were generated for a 90-day horizon, including variance and residual variance.")
```

```
Analysis Summary:
1. ARCH and GARCH models were successfully fitted to the returns data.
2. Conditional volatility was plotted for both ARCH and GARCH models.
3. Residuals were checked for autocorrelation using the Ljung-Box test.
4. Forecasts were generated for a 90-day horizon, including variance and residual variance.
```

Interpretation:

The analysis involved fitting both ARCH and GARCH models to the returns data, enabling the capture of time-varying volatility. The conditional volatility for both models was plotted, illustrating the dynamic nature of market volatility over time. Residuals were rigorously tested for autocorrelation using the Ljung-Box test to ensure model adequacy. Forecasts for a 90-day horizon were generated, providing insights into expected future variance and residual variance. The increasing trend in forecasted variance suggests growing market uncertainty, while the diminishing rate of increase towards the end indicates a potential stabilization in volatility. This comprehensive approach ensures robust modeling and forecasting of market behavior.

Recommendations

It is recommended that investors and portfolio managers prepare for increasing market volatility over the next 90 days. The forecasted rise in variance suggests heightened uncertainty, which may require adjustments in risk management strategies. Diversifying portfolios to include assets with lower correlations to the market may help mitigate potential risks. Additionally, considering hedging strategies such as options or volatility-based instruments could be beneficial. Continuous monitoring of market conditions and regular re-evaluation of the models should be undertaken to ensure adaptive and responsive investment strategies, keeping in mind the diminishing rate of increase in volatility towards the end of the forecast period, which may signal a potential stabilization phase.

R Codes

```
# 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 Amazon
ticker <- "AMZN"
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(AMZN))
print(str(AMZN))

# Step 2: Calculate Returns
market <- Cl(AMZN)  # 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 = 'pink')
```

```
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 = 'violet')
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", garc
hOrder = 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)@fore
cast$seriesFor)
forecast_residual_variance <- as.numeric(ugarchforecast(garch_fit_additional, n.ahea
d = 1)@forecast$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 = 'maroon') +
  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)
```

Python Codes

```
!pip install arch
# Import required libraries
```

```
import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from arch import arch_model
from statsmodels.stats.diagnostic import acorr_ljungbox
import seaborn as sns
# Set plotting style
 sns.set(style="whitegrid")
# Step 1: Download Historical Data
ticker = "AMZN"
data = yf.download(ticker, start="2021-04-01", end="2024-03-31")
# Step 2: Calculate Returns
market = data["Adj Close"]
returns = 100 * market.pct_change().dropna()  # Convert to percentage returns
# Step 3: Fit an ARCH Model
print("\nFitting ARCH Model...")
arch_model_fit = arch_model(returns, vol='ARCH', p=1).fit(disp='off')
print("ARCH Model Summary:")
print(arch_model_fit.summary())
# Plot the conditional volatility from the ARCH model
plt.figure(figsize=(12, 6))
plt.plot(arch_model_fit.conditional_volatility, label='Conditional Volatility (ARCH)',
color='blue')
plt.title('Conditional Volatility from ARCH Model')
plt.xlabel('Date')
plt.ylabel('Volatility')
plt.legend()
plt.grid(True)
plt.show()
# Check residuals for autocorrelation
ljungbox_arch = acorr_ljungbox(arch_model_fit.resid, lags=[10])
print("\nLjung-Box Test for ARCH Model Residuals:")
print(ljungbox_arch)
```

```
# Step 4: Fit a GARCH Model
print("\nFitting GARCH Model...")
garch_model_fit = arch_model(returns, vol='Garch', p=1, q=1).fit(disp='off')
print("GARCH Model Summary:")
print(garch_model_fit.summary())
# Plot the conditional volatility from the GARCH model
plt.figure(figsize=(12, 6))
plt.plot(garch_model_fit.conditional_volatility, label='Conditional Volatility (GARC
H)', color='pink')
plt.title('Conditional Volatility from GARCH Model')
plt.xlabel('Date')
plt.ylabel('Volatility')
plt.legend()
plt.grid(True)
plt.show()
# Check residuals for autocorrelation
ljungbox_garch = acorr_ljungbox(garch_model_fit.resid, lags=[10])
print("\nLjung-Box Test for GARCH Model Residuals:")
print(ljungbox_garch)
# Step 5: Fit GARCH Model with Additional Parameters
print("\nFitting GARCH Model with additional parameters...")
am = arch_model(returns, vol="Garch", p=1, q=1, dist="Normal")
res = am.fit(update_freq=5)
# Print forecast details
forecast_mean = res.forecast().mean
forecast_residual_variance = res.forecast().residual_variance
forecast_variance = res.forecast().variance

print("\nForecast Mean (last 3 periods):")
print(forecast_mean.iloc[-3:])
print("Forecast Residual Variance (last 3 periods):")
print(forecast_residual_variance.iloc[-3:])
print("Forecast Variance (last 3 periods):")
print(forecast_variance.iloc[-3:])
```

```
# Forecasting with a horizon of 90 days
print("\nForecasting 90 days ahead...")
forecasts = res.forecast(horizon=90)
# Print forecast residual variance for the 90-day horizon
print("\n90-day Forecast Residual Variance (last 3 periods):")
print(forecasts.residual_variance.iloc[-3:])
# Conclusion and Summary
print("\nAnalysis Summary:")
print("1. ARCH and GARCH models were successfully fitted to the returns data.")
print("2. Conditional volatility was plotted for both ARCH and GARCH models.")
print("3. Residuals were checked for autocorrelation using the Ljung-Box test.")
print("4. Forecasts were generated for a 90-day horizon, including variance and residu
al variance.")
```

References

1. www.github.com