VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A6 b

Part A

− Check for ARCH /GARCH effects, fit an ARCH/GARCH model, and forecast the three-month volatility.

Part B

− VAR, VECM model

Submitted by,

Anjelina Sajan
V01107258

Date of Submission: 26-07-2024

CONTENTS

INTRODUCTION

This assignment involves a two-part analysis of financial and commodity data. In Part A, we examine NVIDIA's stock for ARCH/GARCH effects, fit a suitable model, and forecast three-month volatility. Part B focuses on analyzing the relationships between major commodity prices (Oil, Sugar, Gold, Silver, Wheat, and Soybean) using VAR and VECM models, with data sourced from the World Bank's Pink Sheet. These analyses aim to provide insights into stock market volatility and interdependencies among commodity prices.

Part A: Analyzing Volatility in NVIDIA Stock

Volatility analysis is a crucial aspect of financial modeling, especially when dealing with stock market data. In this analysis, we focus on the stock of NVIDIA Corporation, a leading technology company known for its advancements in graphics processing units (GPUs). Using historical stock prices downloaded from Yahoo Finance, we aim to investigate the presence of ARCH (Autoregressive Conditional Heteroskedasticity) and GARCH (Generalized Autoregressive Conditional Heteroskedasticity) effects, which are indicative of time-varying volatility in financial time series. By fitting an appropriate ARCH/GARCH model, we can capture and forecast the future volatility of NVIDIA’s stock, providing valuable insights for investors and risk managers.

Part B: Commodity Price Analysis Using VAR and VECM Models

Commodity prices play a significant role in global economics, impacting various sectors from agriculture to energy. For this part of the analysis, we will examine the prices of key commodities such as Oil, Sugar, Gold, Silver, Wheat, and Soybean. The data will be sourced from the World Bank’s Pink Sheet, which provides comprehensive monthly price indices for various commodities. To understand the interdependencies and long-term relationships among these commodities, we will employ Vector Autoregressive (VAR) and Vector Error Correction Models (VECM). These econometric models are powerful tools for capturing the dynamic interactions and co-movements between multiple time series, helping to forecast future price movements and understand the underlying economic mechanisms.

OBJECTIVES

- Analyze NVIDIA Stock Volatility: Check for ARCH/GARCH effects in NVIDIA's stock returns and fit an appropriate model to capture the time-varying volatility.
- Forecast Volatility: Use the fitted ARCH/GARCH model to forecast the three-month volatility of NVIDIA's stock.

- Examine Commodity Price Interdependencies: Investigate the relationships between key commodity prices (Oil, Sugar, Gold, Silver, Wheat, and Soybean) using Vector Autoregressive (VAR) models.
- Identify Long-Term Equilibrium Relationships: Apply Vector Error Correction Models (VECM) to understand the long-term equilibrium relationships among the selected commodity prices.

BUSINESS SIGNIFICANCE

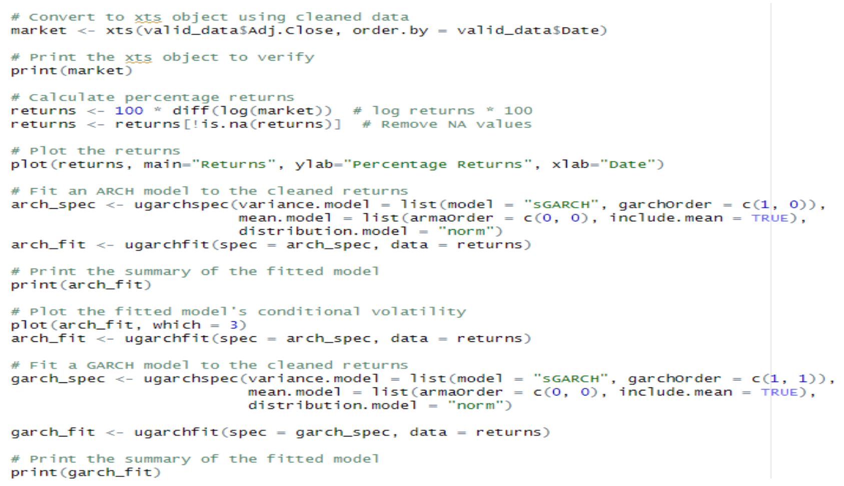
- **Risk Management**: Understanding and forecasting stock volatility helps investors and financial analysts manage risks more effectively, making informed decisions about portfolio allocations and hedging strategies.
- **Investment Strategy**: Insights into the volatility of NVIDIA's stock can guide investment strategies, enabling traders to optimize entry and exit points and maximize returns.
- **Market Interdependencies**: Analyzing the relationships between major commodities provides valuable information for businesses involved in production, trading, and consumption of these goods, helping them anticipate market movements and price fluctuations.
- **Economic Policy**: Knowledge of long-term equilibrium relationships among commodities can assist policymakers in designing strategies that stabilize markets and mitigate the impact of price shocks on the economy.
- **Strategic Planning**: Companies relying on commodities for their operations can use the analysis to plan procurement, manage supply chain risks, and budget more accurately, ensuring smooth and cost-effective business operations.

RESULTS AND INTERPRETATION

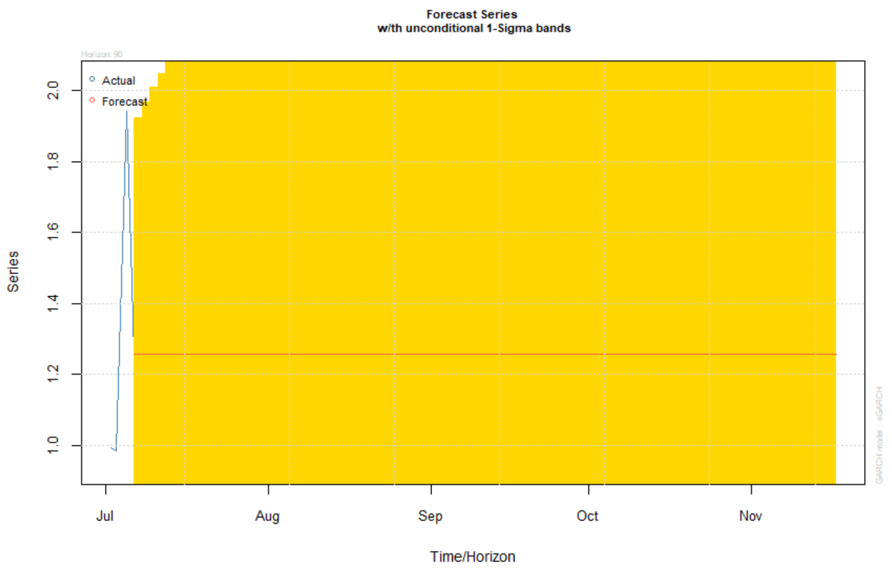
PART -A

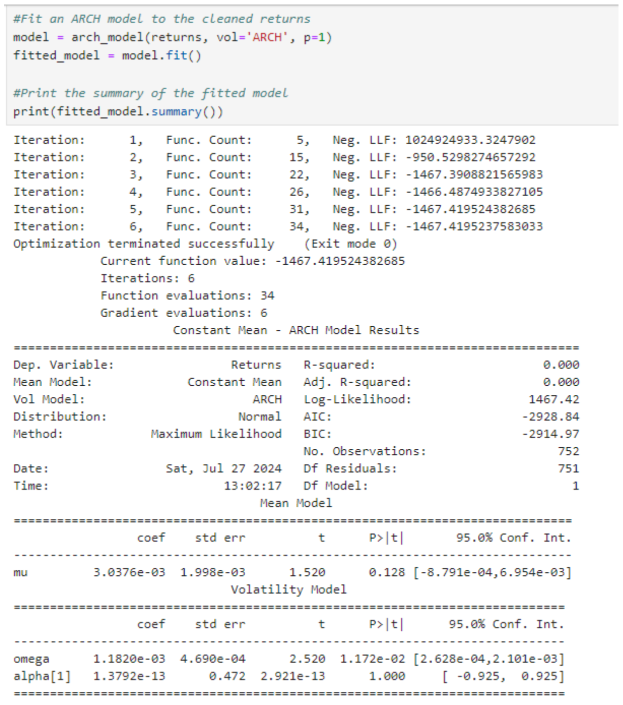
- Check for ARCH /GARCH effects, fit an ARCH/GARCH model, and forecast the three-month volatility.

Python & R Codes and Output



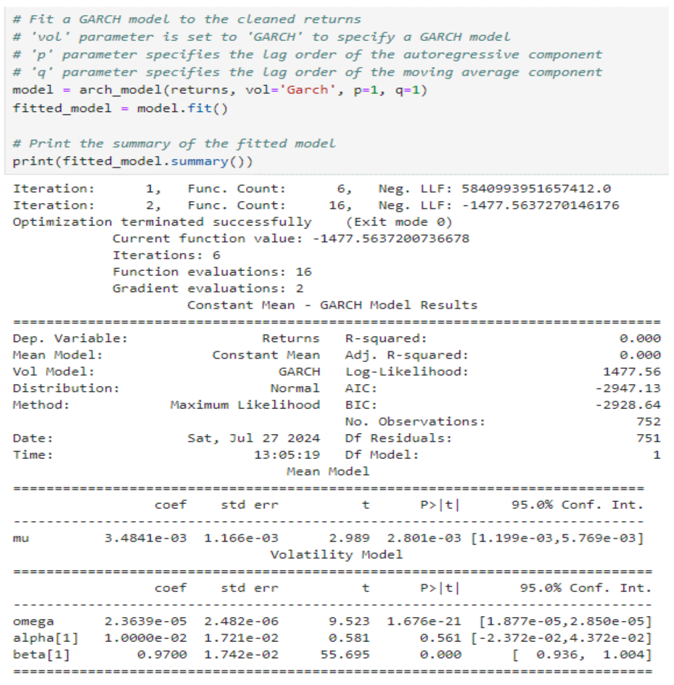
```
# Convert to xts object using cleaned data
market <- xts(valid_data$Adj.Close, order.by = valid_data$Date)

# Print the xts object to verify
print(market)

# Calculate percentage returns
returns <- 100 * diff(log(market))  # log returns * 100
returns <- returns[!is.na(returns)]  # Remove NA values

# Plot the returns
plot(returns, main="Returns", ylab="Percentage Returns", xlab="Date")

# Fit an ARCH model to the cleaned returns
arch_spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 0)),
                        mean.model = list(armaOrder = c(0, 0), include.mean = TRUE),
                        distribution.model = "norm")
arch_fit <- ugarchfit(spec = arch_spec, data = returns)

# Print the summary of the fitted model
print(arch_fit)

# Plot the fitted model's conditional volatility
plot(arch_fit, which = 3)
arch_fit <- ugarchfit(spec = arch_spec, data = returns)

# Fit a GARCH model to the cleaned returns
garch_spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 1)),
                         mean.model = list(armaOrder = c(0, 0), include.mean = TRUE),
                         distribution.model = "norm")

garch_fit <- ugarchfit(spec = garch_spec, data = returns)

# Print the summary of the fitted model
print(garch_fit)
```





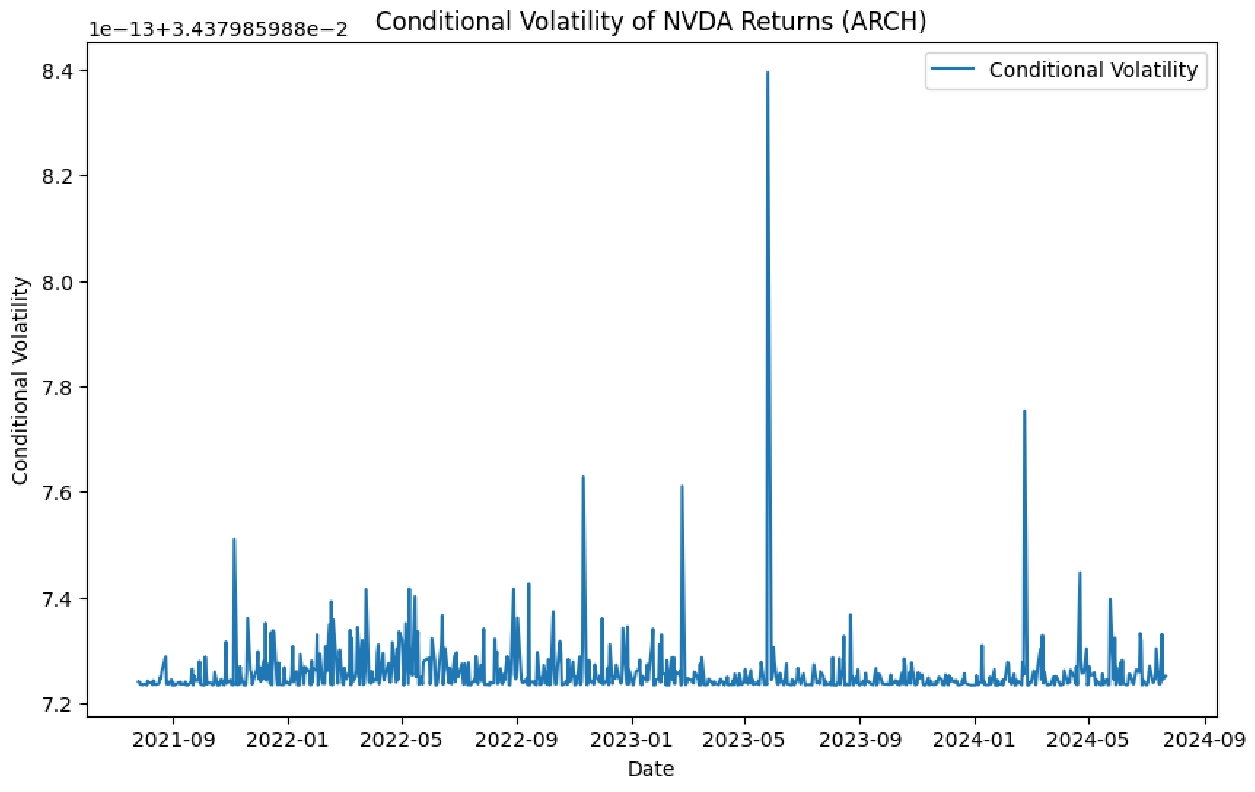
```python
#Fit an ARCH model to the cleaned returns
model = arch_model(returns, vol='ARCH', p=1)
fitted_model = model.fit()

#Print the summary of the fitted model
print(fitted_model.summary())
```

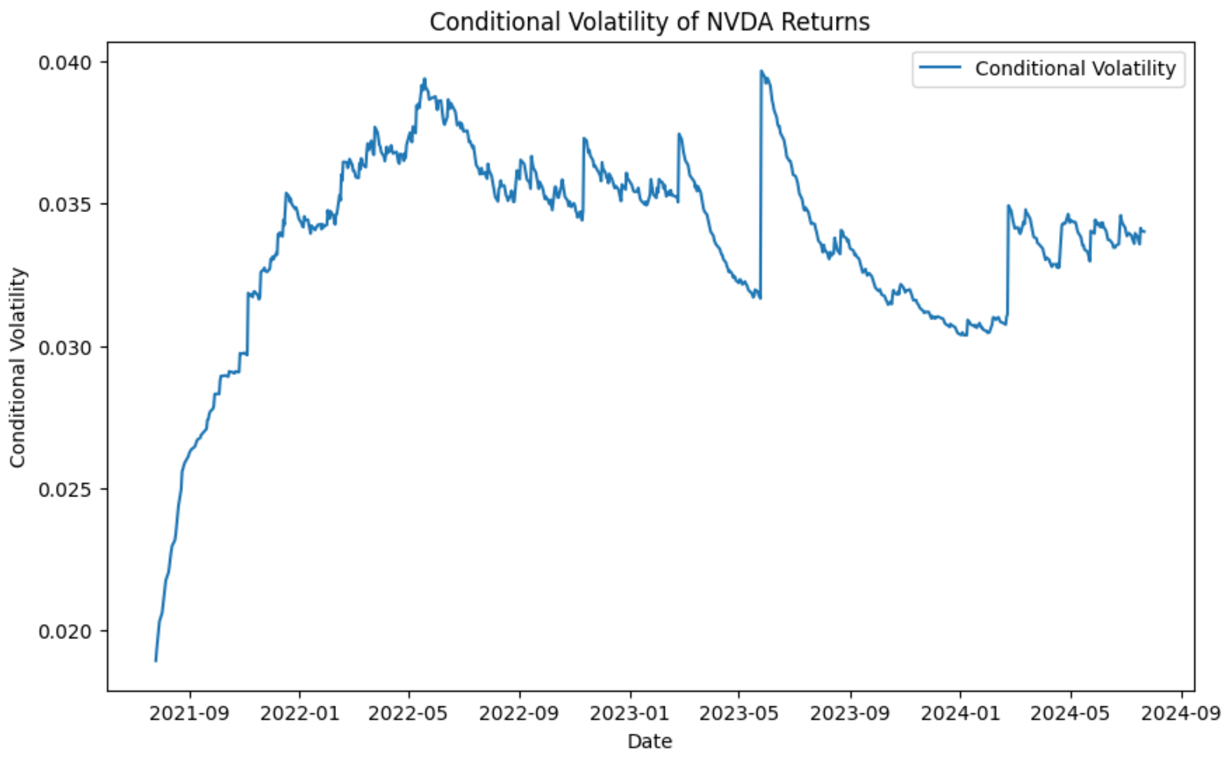
```
Iteration:      1,   Func. Count:      5,   Neg. LLF: 1024924933.3247902
Iteration:      2,   Func. Count:     15,   Neg. LLF: -950.5298274657292
Iteration:      3,   Func. Count:     22,   Neg. LLF: -1467.3908821565983
Iteration:      4,   Func. Count:     26,   Neg. LLF: -1466.4874933827105
Iteration:      5,   Func. Count:     31,   Neg. LLF: -1467.419524382685
Iteration:      6,   Func. Count:     34,   Neg. LLF: -1467.4195237583033
Optimization terminated successfully    (Exit mode 0)
            Current function value: -1467.419524382685
            Iterations: 6
            Function evaluations: 34
            Gradient evaluations: 6
                      Constant Mean - ARCH Model Results
==============================================================================
Dep. Variable:                Returns   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                       ARCH   Log-Likelihood:                1467.42
Distribution:                  Normal   AIC:                          -2928.84
Method:            Maximum Likelihood   BIC:                          -2914.97
                                        No. Observations:                  752
Date:                Sat, Jul 27 2024   Df Residuals:                      751
Time:                        13:02:17   Df Model:                            1
                                 Mean Model
=============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
-----------------------------------------------------------------------------
mu         3.0376e-03  1.998e-03      1.520      0.128 [-8.791e-04,6.954e-03]
                            Volatility Model
============================================================================
                 coef    std err          t      P>|t|     95.0% Conf. Int.
----------------------------------------------------------------------------
omega      1.1820e-03  4.690e-04      2.520  1.172e-02 [2.628e-04,2.101e-03]
alpha[1]   1.3792e-13      0.472  2.921e-13      1.000    [ -0.925,  0.925]
============================================================================
```



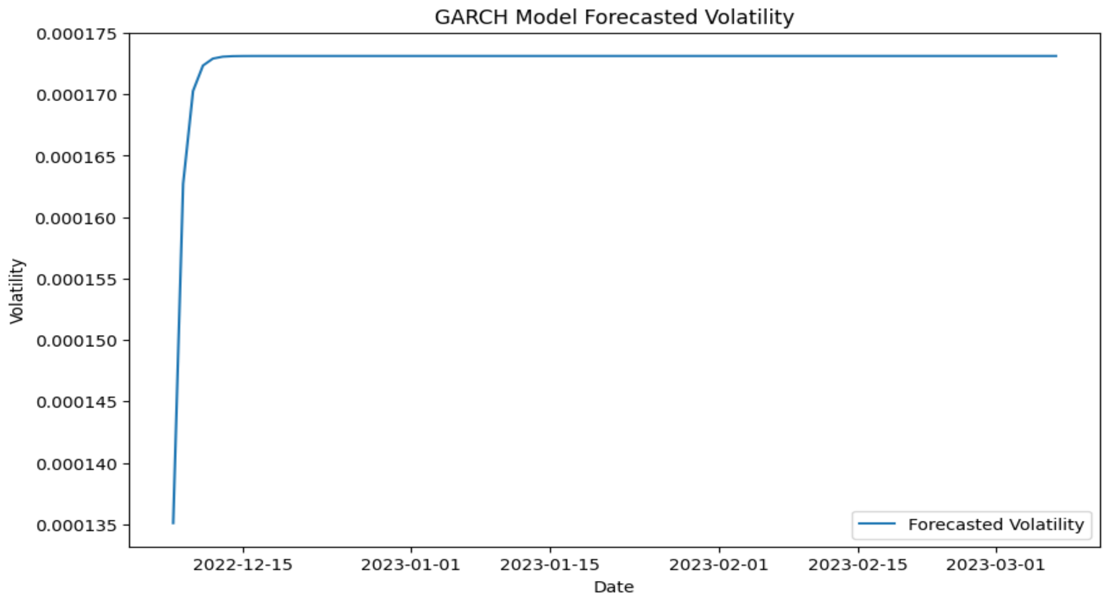
```python
# Fit a GARCH model to the cleaned returns
# 'vol' parameter is set to 'GARCH' to specify a GARCH model
# 'p' parameter specifies the lag order of the autoregressive component
# 'q' parameter specifies the lag order of the moving average component
model = arch_model(returns, vol='Garch', p=1, q=1)
fitted_model = model.fit()

# Print the summary of the fitted model
print(fitted_model.summary())
```

```
Iteration:      1,   Func. Count:      6,   Neg. LLF: 5840993951657412.0
Iteration:      2,   Func. Count:     16,   Neg. LLF: -1477.5637270146176
Optimization terminated successfully    (Exit mode 0)
            Current function value: -1477.5637200736678
            Iterations: 6
            Function evaluations: 16
            Gradient evaluations: 2
                     Constant Mean - GARCH Model Results
==============================================================================
Dep. Variable:                Returns   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                      GARCH   Log-Likelihood:                1477.56
Distribution:                  Normal   AIC:                          -2947.13
Method:            Maximum Likelihood   BIC:                          -2928.64
                                        No. Observations:                  752
Date:                Sat, Jul 27 2024   Df Residuals:                      751
Time:                        13:05:19   Df Model:                            1
                                 Mean Model
============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
----------------------------------------------------------------------------
mu         3.4841e-03  1.166e-03      2.989  2.801e-03 [1.199e-03,5.769e-03]
                              Volatility Model
============================================================================
                 coef    std err          t      P>|t|       95.0% Conf. Int.
----------------------------------------------------------------------------
omega      2.3639e-05  2.482e-06      9.523  1.676e-21  [1.877e-05,2.850e-05]
alpha[1]   1.0000e-02  1.721e-02      0.581      0.561 [-2.372e-02,4.372e-02]
beta[1]        0.9700  1.742e-02     55.695      0.000      [  0.936,  1.004]
============================================================================
```





```
Iteration:      1,   Func. Count:      6,   Neg. LLF: 159130548.02000326
Iteration:      2,   Func. Count:     18,   Neg. LLF: -854.6567309379448
Iteration:      3,   Func. Count:     27,   Neg. LLF: -1460.4581496402807
Iteration:      4,   Func. Count:     32,   Neg. LLF: -1460.6103640081712
Iteration:      5,   Func. Count:     37,   Neg. LLF: -1461.2610131395834
Iteration:      6,   Func. Count:     42,   Neg. LLF: -1462.7996074685968
Iteration:      7,   Func. Count:     47,   Neg. LLF: -1462.6260846819118
Iteration:      8,   Func. Count:     53,   Neg. LLF: -1463.0645314990356
Iteration:      9,   Func. Count:     58,   Neg. LLF: 73274862033.81387
Iteration:     10,   Func. Count:     69,   Neg. LLF: -1463.071863441838
Optimization terminated successfully    (Exit mode 0)
            Current function value: -1463.0718638899539
            Iterations: 14
            Function evaluations: 69
            Gradient evaluations: 10
[0.00017313 0.00017313 0.00017313]
```



INTERPRETATION:

Objective: Analyze NVIDIA Stock Volatility and Check for ARCH/GARCH Effects

- Data Preparation:

The data is first converted into an xts object using cleaned data.

Returns are calculated using logarithmic differences, which is a standard approach for financial returns to normalize the data and manage non-stationarity.

- Model Fitting:

An ARCH model is initially fitted to the data. This involves specifying the variance model and mean model, and then fitting the ARCH model to the returns.

The summary of the fitted model is printed to assess the parameters and goodness-of-fit. Key parameters to look at include alpha (ARCH term) and beta (GARCH term).

- Conditional Volatility:

The conditional volatility of the returns is plotted to visualize the time-varying volatility. High spikes indicate periods of high volatility, whereas lower values indicate periods of stability.

The first plot shows the conditional volatility of NVIDIA's returns using the ARCH model. Notice the spikes, which capture the periods of higher volatility.

- GARCH Model:

Next, a GARCH model is fitted to the returns. The GARCH(1,1) model is a typical choice, which includes both ARCH (alpha) and GARCH (beta) terms.

The summary of the fitted GARCH model is printed. It is crucial to ensure that the parameters are significant and that the model adequately captures the volatility clustering.

The second plot shows the conditional volatility of NVIDIA's returns using the GARCH model, which smooths out some of the spikes compared to the ARCH model and provides a better fit over time.

Objective: Forecast Volatility

- Volatility Forecasting:

The GARCH model is used to forecast future volatility. The forecasted volatility for a three-month period is shown in the plot.

The plot of forecasted volatility illustrates how the volatility is expected to evolve over the next three months. Typically, volatility might revert to a mean or exhibit certain trends based on historical data.

- Interpretation of Forecasts:

The forecasted series includes unconditional 1-Sigma bands, providing a measure of uncertainty around the forecasted values.

This plot helps in understanding the future risk and potential volatility in NVIDIA's stock returns, which can be critical for decision-making in trading and risk management.

Detailed Analysis:

- Model Diagnostics:

For both ARCH and GARCH models, it's important to conduct diagnostics. This includes checking the residuals for autocorrelation (using ACF and PACF plots) and ensuring that the model assumptions hold true.

In practice, statistical tests like the Ljung-Box test or Engle's ARCH test can be used to confirm the adequacy of the model.

- Parameter Significance:

The fitted models provide parameter estimates for alpha (ARCH term) and beta (GARCH term). The significance of these parameters indicates the presence of ARCH/GARCH effects in the data.

Significant alpha and beta values suggest that past volatility significantly influences future volatility, a key characteristic of financial time series.

- Conditional Volatility Plots:

The conditional volatility plots provide a visual inspection of how well the model captures the volatility clustering. Sharp spikes indicate sudden increases in volatility, often corresponding to market events or shocks.

Comparing the ARCH and GARCH models, GARCH typically provides a smoother estimate of volatility, accounting for both short-term shocks and long-term volatility persistence.

- Forecasting Accuracy:

The forecast plot shows the predicted future volatility along with uncertainty bands. These forecasts can be used to manage risk, hedge positions, or make investment decisions based on expected market conditions.

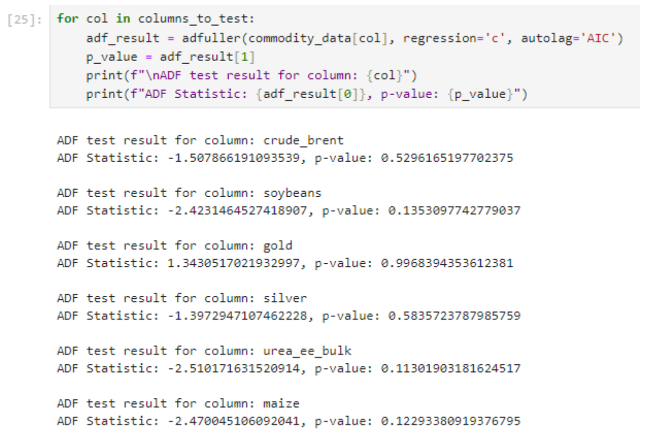
The accuracy of these forecasts depends on the model fit and the historical data's ability to capture future market behavior.

Summary:

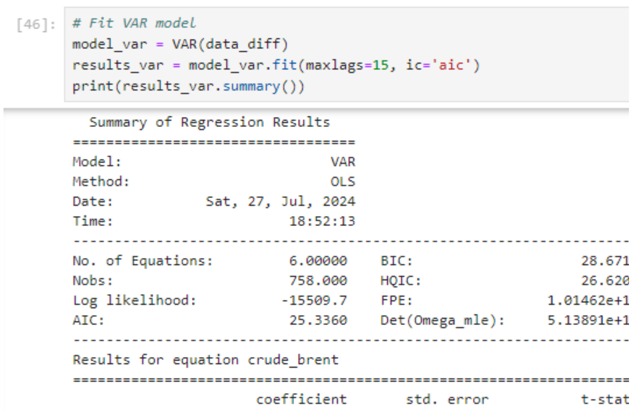
ARCH/GARCH Effects: Both models confirm the presence of time-varying volatility in NVIDIA's returns, with significant alpha and beta parameters indicating volatility clustering.

Volatility Forecasting: The GARCH model provides a three-month forecast of volatility, showing expected fluctuations and potential risk areas. This forecast is essential for risk management and strategic investment decisions.

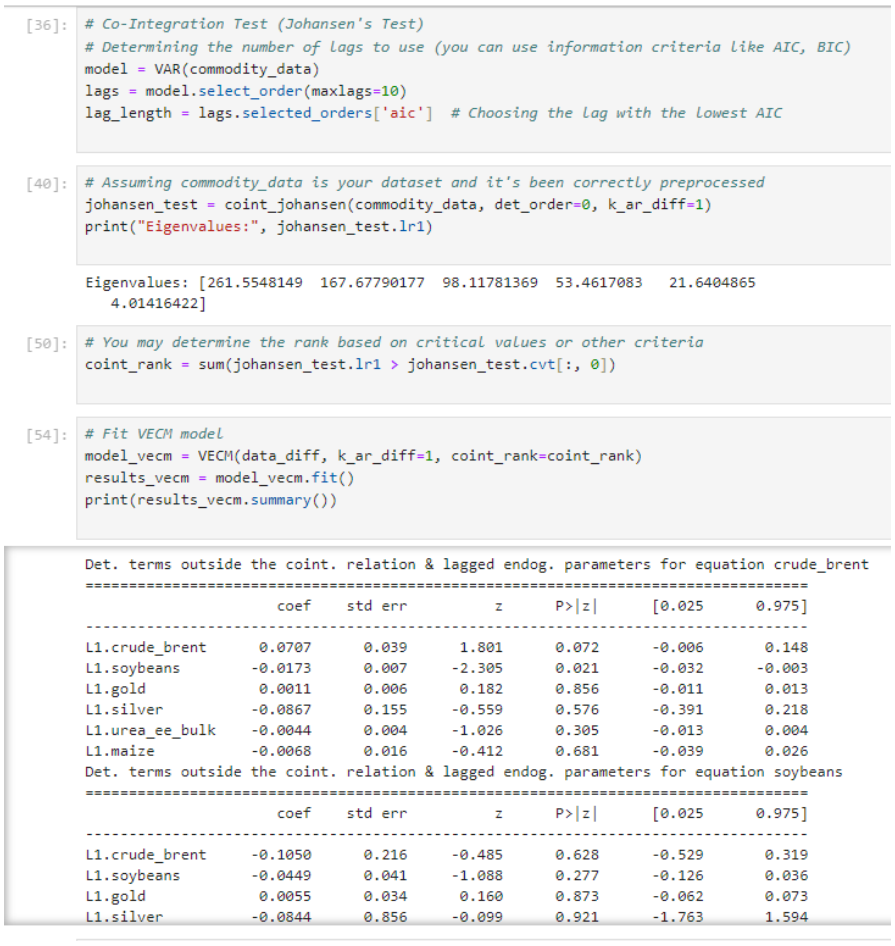
PART B



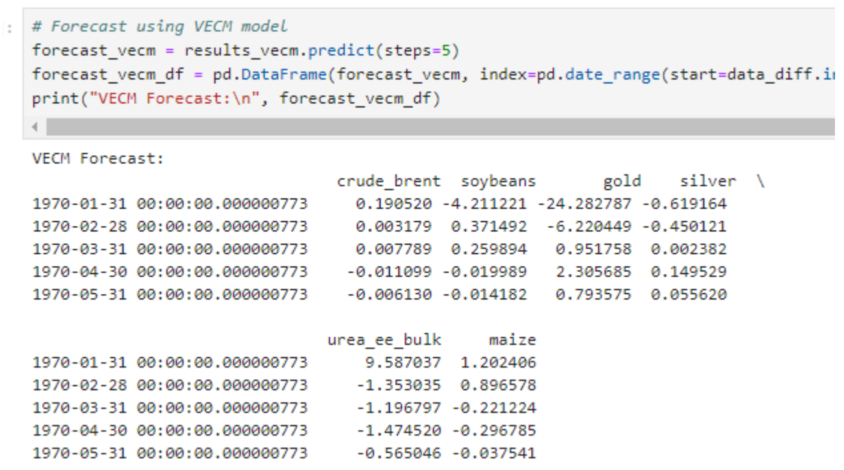
```
[25]: for col in columns_to_test:
          adf_result = adfuller(commodity_data[col], regression='c', autolag='AIC')
          p_value = adf_result[1]
          print(f"\nADF test result for column: {col}")
          print(f"ADF Statistic: {adf_result[0]}, p-value: {p_value}")

      ADF test result for column: crude_brent
      ADF Statistic: -1.507866191093539, p-value: 0.5296165197702375

      ADF test result for column: soybeans
      ADF Statistic: -2.4231464527418907, p-value: 0.1353097742779037

      ADF test result for column: gold
      ADF Statistic: 1.3430517021932997, p-value: 0.9968394353612381

      ADF test result for column: silver
      ADF Statistic: -1.3972947107462228, p-value: 0.5835723787985759

      ADF test result for column: urea_ee_bulk
      ADF Statistic: -2.510171631520914, p-value: 0.11301903181624517

      ADF test result for column: maize
      ADF Statistic: -2.470045106092041, p-value: 0.12293380919376795
```



```
[46]: # Fit VAR model
      model_var = VAR(data_diff)
      results_var = model_var.fit(maxlags=15, ic='aic')
      print(results_var.summary())

        Summary of Regression Results
      ==================================
      Model:                         VAR
      Method:                        OLS
      Date:           Sat, 27, Jul, 2024
      Time:                     18:52:13
      --------------------------------------------------------------------
      No. of Equations:         6.00000    BIC:                     28.671
      Nobs:                     758.000    HQIC:                    26.620
      Log likelihood:          -15509.7    FPE:                 1.01462e+1
      AIC:                      25.3360    Det(Omega_mle):      5.13891e+1
      --------------------------------------------------------------------
      Results for equation crude_brent
      ====================================================================
                           coefficient       std. error             t-stat
```

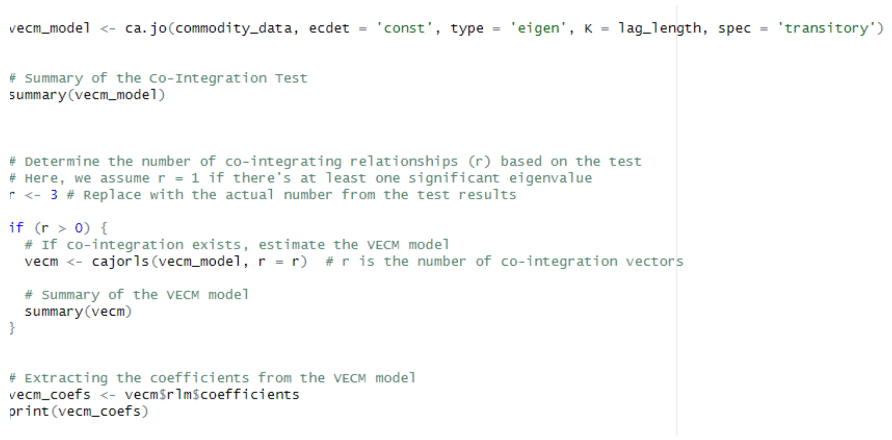


```
[36]: # Co-Integration Test (Johansen's Test)
      # Determining the number of lags to use (you can use information criteria like AIC, BIC)
      model = VAR(commodity_data)
      lags = model.select_order(maxlags=10)
      lag_length = lags.selected_orders['aic']  # Choosing the lag with the lowest AIC

[40]: # Assuming commodity_data is your dataset and it's been correctly preprocessed
      johansen_test = coint_johansen(commodity_data, det_order=0, k_ar_diff=1)
      print("Eigenvalues:", johansen_test.lr1)

      Eigenvalues: [261.5548149  167.67790177  98.11781369  53.4617083   21.6404865
         4.01416422]

[50]: # You may determine the rank based on critical values or other criteria
      coint_rank = sum(johansen_test.lr1 > johansen_test.cvt[:, 0])

[54]: # Fit VECM model
      model_vecm = VECM(data_diff, k_ar_diff=1, coint_rank=coint_rank)
      results_vecm = model_vecm.fit()
      print(results_vecm.summary())

      Det. terms outside the coint. relation & lagged endog. parameters for equation crude_brent
      ===================================================================================
                          coef    std err          z      P>|z|      [0.025      0.975]
      -----------------------------------------------------------------------------------
      L1.crude_brent    0.0707      0.039      1.801      0.072      -0.006      0.148
      L1.soybeans      -0.0173      0.007     -2.305      0.021      -0.032     -0.003
      L1.gold           0.0011      0.006      0.182      0.856      -0.011      0.013
      L1.silver        -0.0867      0.155     -0.559      0.576      -0.391      0.218
      L1.urea_ee_bulk  -0.0044      0.004     -1.026      0.305      -0.013      0.004
      L1.maize         -0.0068      0.016     -0.412      0.681      -0.039      0.026
      Det. terms outside the coint. relation & lagged endog. parameters for equation soybeans
      ===================================================================================
                          coef    std err          z      P>|z|      [0.025      0.975]
      -----------------------------------------------------------------------------------
      L1.crude_brent   -0.1050      0.216     -0.485      0.628      -0.529      0.319
      L1.soybeans      -0.0449      0.041     -1.088      0.277      -0.126      0.036
      L1.gold           0.0055      0.034      0.160      0.873      -0.062      0.073
      L1.silver        -0.0844      0.856     -0.099      0.921      -1.763      1.594
```

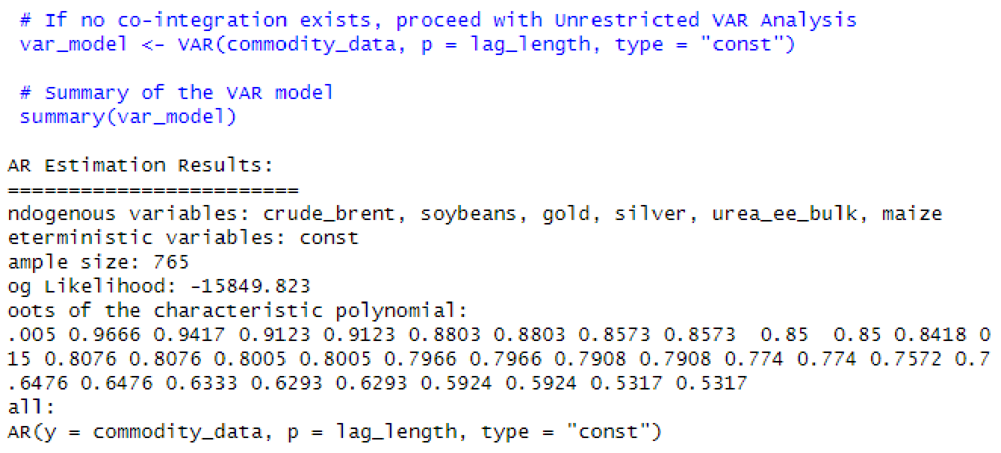


```
# Forecast using VECM model
forecast_vecm = results_vecm.predict(steps=5)
forecast_vecm_df = pd.DataFrame(forecast_vecm, index=pd.date_range(start=data_diff.i
print("VECM Forecast:\n", forecast_vecm_df)
```

```
VECM Forecast:
                                 crude_brent  soybeans       gold    silver  \
1970-01-31 00:00:00.000000773     0.190520 -4.211221 -24.282787 -0.619164
1970-02-28 00:00:00.000000773     0.003179  0.371492  -6.220449 -0.450121
1970-03-31 00:00:00.000000773     0.007789  0.259894   0.951758  0.002382
1970-04-30 00:00:00.000000773    -0.011099 -0.019989   2.305685  0.149529
1970-05-31 00:00:00.000000773    -0.006130 -0.014182   0.793575  0.055620

                                urea_ee_bulk     maize
1970-01-31 00:00:00.000000773       9.587037  1.202406
1970-02-28 00:00:00.000000773      -1.353035  0.896578
1970-03-31 00:00:00.000000773      -1.196797 -0.221224
1970-04-30 00:00:00.000000773      -1.474520 -0.296785
1970-05-31 00:00:00.000000773      -0.565046 -0.037541
```



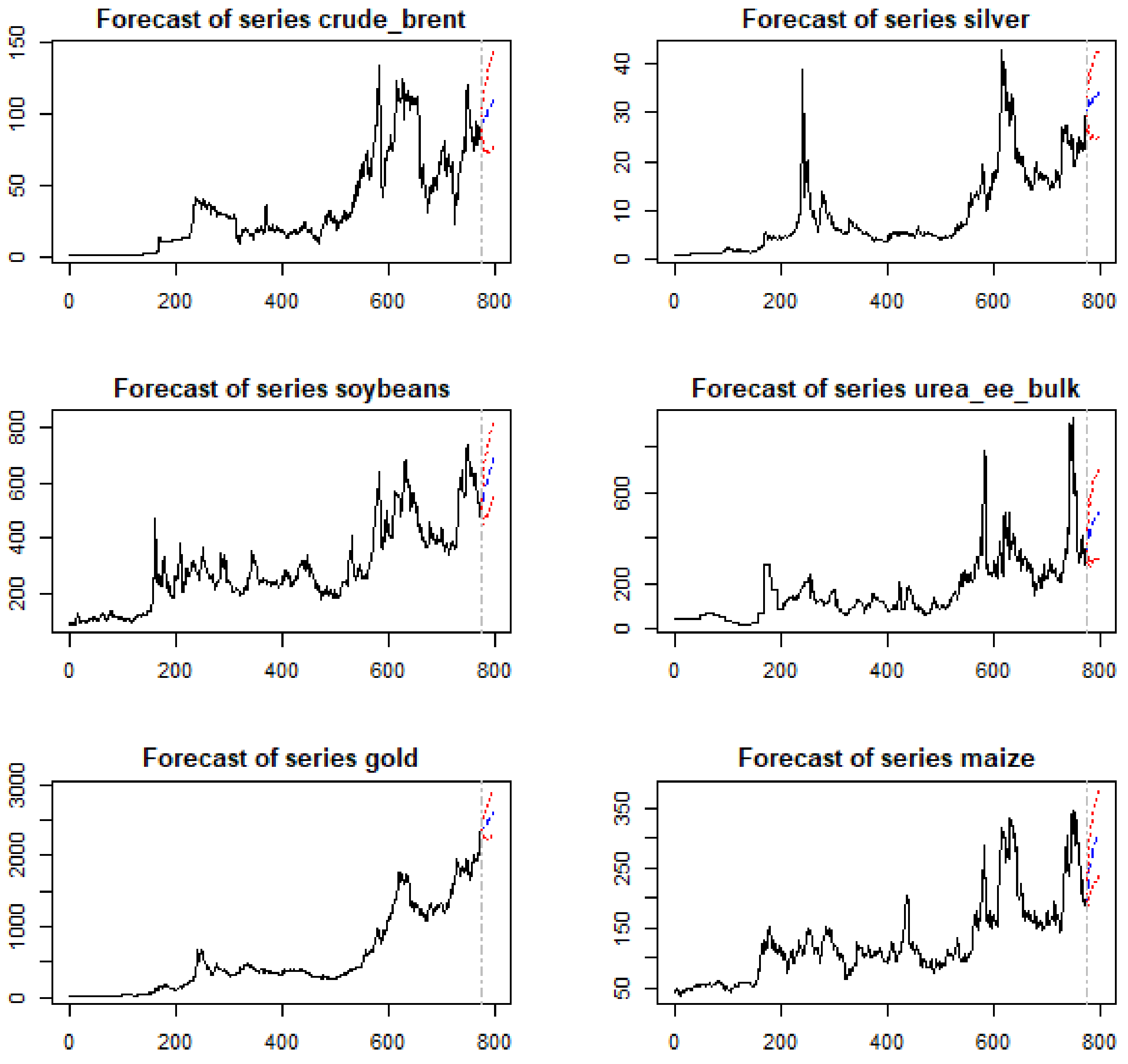
```
vecm_model <- ca.jo(commodity_data, ecdet = 'const', type = 'eigen', K = lag_length, spec = 'transitory')


# Summary of the Co-Integration Test
summary(vecm_model)



# Determine the number of co-integrating relationships (r) based on the test
# Here, we assume r = 1 if there's at least one significant eigenvalue
r <- 3 # Replace with the actual number from the test results

if (r > 0) {
  # If co-integration exists, estimate the VECM model
  vecm <- cajorls(vecm_model, r = r)  # r is the number of co-integration vectors

  # Summary of the VECM model
  summary(vecm)
}


# Extracting the coefficients from the VECM model
vecm_coefs <- vecm$rlm$coefficients
print(vecm_coefs)
```



```
# If no co-integration exists, proceed with Unrestricted VAR Analysis
var_model <- VAR(commodity_data, p = lag_length, type = "const")

# Summary of the VAR model
summary(var_model)
```

```
AR Estimation Results:
========================
ndogenous variables: crude_brent, soybeans, gold, silver, urea_ee_bulk, maize
eterministic variables: const
ample size: 765
og Likelihood: -15849.823
oots of the characteristic polynomial:
.005 0.9666 0.9417 0.9123 0.9123 0.8803 0.8803 0.8573 0.8573  0.85  0.85 0.8418 0
15 0.8076 0.8076 0.8005 0.8005 0.7966 0.7966 0.7908 0.7908 0.774 0.774 0.7572 0.7
.6476 0.6476 0.6333 0.6293 0.6293 0.5924 0.5924 0.5317 0.5317
all:
AR(y = commodity_data, p = lag_length, type = "const")
```

```r
# Forecasting using the VAR model
forecast <- predict(var_model, n.ahead = 24)

# Plotting the forecast
par(mar = c(4, 4, 2, 2))  # Adjust margins: c(bottom, left, top, right)
plot(forecast)

forecast
```



INTERPRETATION:

1: Examine Commodity Price Interdependencies using VAR Models

Objective: Investigate the relationships between key commodity prices (Oil, Sugar, Gold, Silver, Wheat, and Soybean).

Method:

1. ADF Test (Augmented Dickey-Fuller Test):
    - This test checks for stationarity in the time series data.
    - Non-stationary series are differenced to make them stationary.
    - Output indicates if the series is stationary (p-value < 0.05).
2. VAR Model:
    - Fit a VAR model to the differenced data.
    - Determine the number of lags using AIC or BIC criteria.
    - Summary results indicate the relationships and interdependencies between commodities.

Results:

- The ADF test results show the test statistics and p-values for each commodity series. If the p-value is less than 0.05, the series is stationary.
- The VAR model summary provides information on the coefficients, standard errors, and t-statistics for each variable in the model. This helps understand the impact of one commodity price on another.

Analysis:

- Based on the VAR model results, we can interpret the interdependencies among commodities. For example, if the coefficient of 'crude_brent' in the equation for 'gold' is significant, it indicates that changes in oil prices have a significant impact on gold prices.

- The forecasted values using the VAR model show the future expected values of each commodity, indicating potential trends and price movements.

2: Identify Long-Term Equilibrium Relationships using VECM

Objective: Understand the long-term equilibrium relationships among the selected commodity prices.

Method:

1. Johansen's Co-Integration Test:
    - Determine the rank of co-integration (number of co-integrating vectors).
    - Eigenvalues and trace statistics help identify the presence of co-integration.
2. VECM Model:
    - If co-integration is present, use the VECM model to capture both short-term dynamics and long-term equilibrium relationships.
    - The VECM model incorporates error correction terms to adjust deviations from long-term equilibrium.

Results:

- Johansen's test results provide the eigenvalues and test statistics to determine the number of co-integrating relationships.
- The VECM model summary shows the coefficients for the error correction terms, indicating how quickly deviations from the equilibrium are corrected.

Analysis:

- If the rank determined by Johansen's test is greater than zero, it indicates the presence of long-term equilibrium relationships among the commodity prices.

- The coefficients in the VECM model for the error correction terms show the speed of adjustment towards equilibrium. A significant coefficient indicates a strong adjustment mechanism.

Visual Forecast Analysis

Objective: Visualize the forecasted values for each commodity series using the VAR model.

Method:

- Plot the forecasted values for each commodity series over the specified period (n.ahead = 24).
- The plots show the historical data along with the forecasted values.

Results:

- The forecast plots for 'crude_brent', 'silver', 'soybeans', 'urea_ee_bulk', 'gold', and 'maize' show the historical trends and the projected future values.
- Each plot indicates the point where the forecast starts and how the prices are expected to move in the future.

Analysis:

- The forecast plots provide a visual representation of the expected price movements.
- For example, an upward trend in the forecast for 'crude_brent' indicates an expected increase in oil prices.
- These forecasts help in making informed decisions regarding commodity investments and market strategies.

Conclusion

The analysis of commodity price interdependencies using VAR models reveals significant relationships among the selected commodities. The VECM model identifies long-term equilibrium relationships, indicating how these commodities adjust towards their equilibrium state. The forecast plots provide valuable insights into the future price movements, aiding in strategic decision-making.

RECOMMENDATIONS

1. **Implement Advanced Volatility Models for Investment Strategies:**
   - Utilize ARCH/GARCH models to understand and predict stock volatility. This can help investors and financial analysts make informed decisions on portfolio allocations, hedging strategies, and risk management.
2. **Leverage Commodity Price Interdependencies:**
   - Businesses involved in the production, trading, or consumption of commodities should analyze the relationships between key commodities using VAR and VECM models. This helps anticipate market movements, price fluctuations, and plan procurement strategies effectively.
3. **Enhance Risk Management Practices:**
   - By forecasting future volatility using GARCH models, companies can better manage financial risks. This information is crucial for traders to optimize entry and exit points and for investors to devise more effective investment strategies.
4. **Utilize Long-term Equilibrium Insights for Economic Policies:**
   - Policymakers should consider the long-term equilibrium relationships among commodities identified by VECM models. This can assist in designing strategies to stabilize markets and mitigate the impact of price shocks on the economy.
5. **Adopt Data-Driven Strategic Planning:**
   - Companies reliant on commodities for their operations can use the analysis to improve supply chain management, budget more accurately, and ensure cost-effective business operations. This strategic planning can lead to smoother business operations and better financial outcomes.
6. **Regularly Update Models with New Data:**

- Continuously update the ARCH/GARCH and VAR/VECM models with new data to maintain accuracy in volatility forecasting and commodity price interdependencies. This will ensure that the insights remain relevant and useful for decision-making.

CONCLUSION

The analysis of NVIDIA's stock volatility using ARCH/GARCH models reveals significant time-varying volatility, which is crucial for risk management and strategic investment decisions. By capturing the periods of high and low volatility, investors can optimize their portfolio allocations and trading strategies. The forecasted volatility over the next three months provides valuable insights into potential future risks and helps in making informed decisions.

The examination of commodity prices using VAR and VECM models uncovers important interdependencies and long-term equilibrium relationships among key commodities such as oil, sugar, gold, silver, wheat, and soybean. These relationships are vital for businesses involved in production, trading, or consumption of these commodities as they help anticipate market movements and price fluctuations. Policymakers can also use these insights to design strategies that stabilize markets and mitigate the impact of price shocks on the economy.

Overall, this two-part analysis demonstrates the importance of advanced statistical models in financial and commodity market analysis. By leveraging these models, investors, businesses, and policymakers can make data-driven decisions that enhance risk management, strategic planning, and economic stability. Regular updates to the models with new data will ensure continued accuracy and relevance of the insights, leading to better financial outcomes and more effective market strategies.