Gold Price Forecasting Using Time Series

Introduction: The following document outlines the process of performing time series analysis on gold price data using Python. This analysis aims to forecast future gold prices based on historical price data and identify underlying patterns and trends. The analysis includes data preparation, exploratory data analysis, feature engineering, stationarity checks, model selection (ARIMA and SARIMA), forecasting, evaluation, and visualization of results.

Code Explanation:

1. **Data Preparation:**
    - Read the data from a CSV file.
    - Set the 'Date' column as the index.
    - Resample the data to a daily frequency.
    -
2. **Exploratory Data Analysis (EDA):**
    - This section is left empty for users to perform their own analysis.
    - Explore and visualize the time series data to gain insights.
3. **Feature Engineering:**
    - Calculate the daily returns, which can be used as a feature.
    - Create lagged returns to capture historical price information.
4. **Check for Stationarity using Rolling Statistics:**
    - Compute the rolling mean and standard deviation of the 'Close' prices.
    - Plot the rolling statistics to visually inspect the stationarity of the data.
5. **Check for Stationarity using Dickey-Fuller Test:**
    - Perform the Dickey-Fuller test on the 'Close' prices.
    - Print the test statistic, p-value, and critical values to assess stationarity.
6. **Make the Data Stationary:**
    - Compute the differenced series of the 'Close' prices to achieve stationarity.
7. **Check for Stationarity of Differenced Data using Rolling Statistics:**
    - Calculate the rolling mean and standard deviation of the differenced series.
    - Plot the rolling statistics to visually inspect the stationarity of the differenced data.
8. **Check for Stationarity of Differenced Data using Dickey-Fuller Test:**
    - Apply the Dickey-Fuller test to the differenced series.
    - Print the test statistic, p-value, and critical values to assess stationarity.
9. **Plot Autocorrelation and Partial Autocorrelation Functions:**
    - This section is left empty for users to plot the ACF and PACF if needed.

- ACF and PACF plots can provide insights into the underlying patterns and potential model orders.

10. Split the Data into Training and Test Sets:
- Divide the differenced series into training and test sets.
- Use 80% of the data for training.

11. Select the Best ARIMA Parameters using Grid Search:
- Perform a grid search over different combinations of ARIMA parameters.
- Find the model with the lowest AIC (Akaike Information Criterion) on the training data.

12. Fit the ARIMA Model:
- Fit the ARIMA model with the best parameters to the training data.

13. Forecast Future Prices:
- Generate predictions for the test set using the trained ARIMA model.

14. Evaluate the ARIMA Model:
- Calculate evaluation metrics (MAE, MSE, RMSE, R-squared) by comparing the predicted values with the actual values in the test set.

15. Visualize the Predictions for ARIMA:
- Plot the actual and predicted values of the stationary close price for visualization.

16. Scatter Plot of price_change vs model price_change:
- Create a scatter plot comparing the actual stationary close prices with the predicted prices.

17. Scatter Plot of Residuals:
- Create a scatter plot of the actual stationary close prices against the residuals (the difference between actual and predicted prices).

18. Select the Best SARIMA Parameters using Grid Search:
- Perform a grid search over different combinations of SARIMA parameters.
- Find the model with the lowest AIC on the training data.

19. Fit the SARIMA Model:
- Fit the SARIMA model with the best parameters to the training data.

20. Forecast Future Prices:
- Generate predictions for the test set using the trained SARIMA model.

21. Evaluate the SARIMA Model:
- Calculate evaluation metrics (MAE, MSE, RMSE, R-squared) by comparing the predicted values with the actual values in the test set.

22. Visualize the Predictions for SARIMA:
- Plot the actual and predicted values of the stationary close price for visualization.

23. **Scatter Plot of price_change vs model price_change:**
    - Create a scatter plot comparing the actual stationary close prices with the predicted prices.
24. **Scatter Plot of Residuals:**
    - Create a scatter plot of the actual stationary close prices against the residuals (the difference between actual and predicted prices).

Conclusion:

Comparing the results of the ARIMA model and the SARIMA model, we can draw the following conclusions:

1. **Mean Absolute Error (MAE):** The MAE for both models is similar, with the SARIMA model having a slightly lower value (2.6309) compared to the ARIMA model (2.6338). This indicates that the SARIMA model has a slightly better accuracy in predicting the absolute difference between the predicted and actual values.

2. **Mean Squared Error (MSE):** The MSE for both models is also quite close, with the SARIMA model having a slightly lower value (35.3033) compared to the ARIMA model (35.2906). Again, this suggests that the SARIMA model performs slightly better in terms of predicting the squared difference between the predicted and actual values.

3. **Root Mean Squared Error (RMSE):** The RMSE for both models is very similar as well, with the SARIMA model having a slightly higher value (5.9417) compared to the ARIMA model (5.9406). The difference is minimal, indicating that both models have similar accuracy in predicting the target variable.

4. **R-squared Score:** The R-squared score for the SARIMA model (-7.8197e-05) is close to zero, indicating that the model explains almost no variation in the target variable. The R-squared score for the ARIMA model (0.0003) is also very low. In both cases, the models have limited ability to capture the variation in the data.

Based on these evaluation metrics, we can conclude that there is not much difference in performance between the ARIMA and SARIMA models. Both models have similar accuracy in predicting the target variable, as indicated by MAE, MSE, and RMSE. However, it is important to note that the R-squared score for both models is very low, suggesting that the models may not be effectively capturing the underlying patterns or relationships in the data.