Candidate: 14

Exam – Predictive Analytics

Introduction

Air pollution is one of the most serious problems around the world, taking into account its various impacts on human health and ecosystems and climate systems. In particular, in urban areas, harmful air pollutants like carbon monoxide (CO) and nitrogen dioxide (NO2) are generated from industrial processes, vehicle emissions, and energy production. Known associations between these pollutants and a wide range of respiratory and cardiovascular diseases enforce the fundamental requirement for highly accurate air quality assessment and forecasting. In addition, RH impacts air quality directly through its role in pollution distribution and chemical transformation.

Air quality trends are usually difficult to understand and predict. Most of the air quality time-series data show periodicity, sudden changes, and interrelated relations among multiple factors, which call for advanced analytics methodologies that can model not only the distinct trend of pollutant level time series but also relationships among multiple variables.

Time series modeling methodologies include appropriately designed ones for these purposes: the AutoRegressive Integrated Moving Average-ARIMA-and Vector AutoRegressive-VAR-models. While ARIMA models focus on univariate time series data, the VAR model allows for a multivariate analysis; thus, the latter is particularly useful in the analysis of relationships between the pollutants.

This analysis takes full advantage of an extended dataset of hourly CO, NO2, and RH measured over quite some time. These variables have been chosen not only for their individual merits but also for the added benefit derived from the understanding of their relationships. It's a very good basis for doing trend analysis, seasonality, and understanding the ability to forecast. Therefore, the selected methodology for the current study will mainly focus on the predictive performance of the ARIMA and VAR models. In doing so, this study also attempts to point out the underlying advantages and disadvantages of each in relation to proper air quality indices representation. The secondary statistical tools to be used will include an Augmented Dickey-Fuller (ADF) test and Granger causality test in examining stationarity and the relationship between the variables.

An extra exercise considers the effectiveness of using different ARIMA specifications for each series individually, rather than the multivariate VAR approach.

The results of this study have great implications for urban planning and policy formulation. For instance, strong predictive models should be able to inform anticipatory measures or strategies that may help mitigate air pollution and thus improve the state of public health. What is obtained from this paper goes beyond mere academic relevance to contributing toward a better development of useful tools for air quality management. This report is organized as follows: Task 1 delivers an exploratory analysis of the data, including visualization and basic statistical metrics. Task 2 focuses on the investigation of statistical dependencies together with the Granger causality test. In Task 3, the performance comparison of the univariate versus multivariate forecasting models is performed. Task 4 evaluates the accuracy of those models, while in another task, it considers the application of specific ARIMA models for each variable.
This will help in presenting a comparative study in terms of air quality dynamics and techniques of forecasting through a structured approach.

Tools and Methods
This research implemented different Python libraries with structured approaches for efficient preprocessing, analysis, and modeling of the air quality dataset. This has been in extensive use within data manipulation, integrating the Date and Time columns as a single datetime index, so offering a logical temporal framework. Within the continuity of time series data, missing values, represented as -200, have been replaced by using the forward-filling and interpolation techniques. Preprocessing cleaned up the dataset, making it ready to be worked with.

Exploratory Data analysis (EDA) was helpful to understand the dataset. Basic visualizations using Matplotlib and Seaborn were used to show the characteristics, trends, and relationships among the variables. Line plots of time series were observed for each variable individually, and then it shows a temporal pattern. Then, a correlation matrix was visualized using heatmapping to look at dependency among the variables. These were useful to obtain a basic understanding of the underlying dataset and helped in subsequent modeling decisions.

In order to be ready for the predictive modeling part, the stationarity of each time series was evaluated utilizing the Augmented Dickey-Fuller (ADF) available in Statsmodels. Time series identified as non-stationary were subjected to differencing and subsequently re-evaluated until stationarity was accomplished, thereby fulfilling a fundamental prerequisite for time series models. Furthermore, Granger Causality Tests were employed to identify causal connections among variables, thereby rationalizing the choice of a multivariate model, such as the Vector Autoregressive (VAR) model, for forecasting purposes.

There are mainly two ways to perform predictive modeling here: VAR and ARIMA. First, the VAR model from Statsmodels was applied to the previously differenced multivariate data set, and the best order of lags was determined according to the AIC and BIC metrics. Meanwhile, for every variable separately, Auto ARIMA from the Pmdarima library was used to model it. These two models then made predictions over the validation set. Their accuracy was measured using MAE and RMSE. The models were also plotted with the real observations to check their prediction capability.

Final model performance evaluation was systematically done through analysis of different metrics for each variable: for example, visual checks were made via residual plots that indicate biases or failures in the models. This method allowed deep understanding of the dataset, including the predictive power of the applied techniques; hence, substantial findings were made in trends and forecasting of air quality.

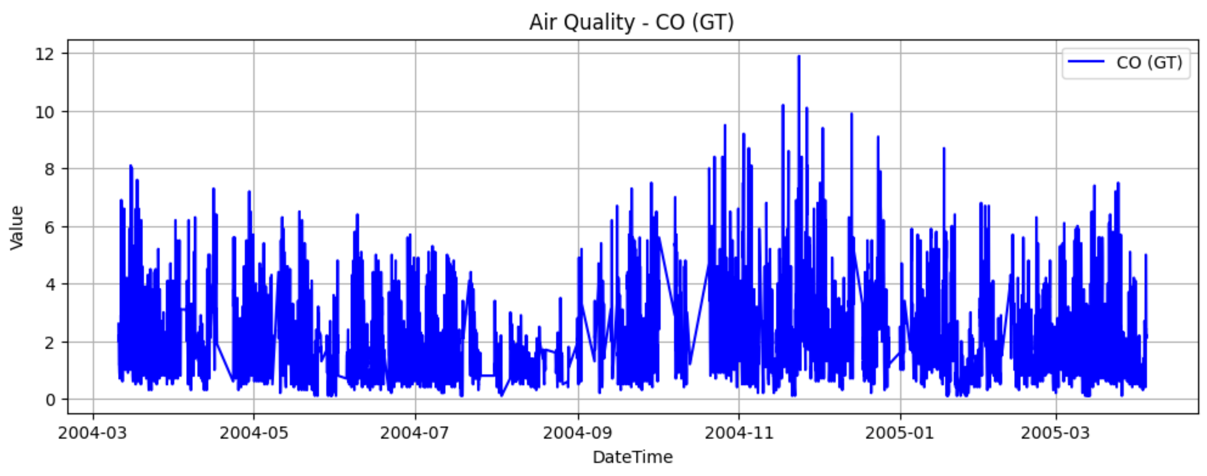
Task 1 – Understanding the Data

The dataset used in this analysis contains hourly averaged air quality measurements collected from a chemical multi-sensor device deployed in a polluted area of an Italian city. Data were recorded in an Italian city polluted area during a period from March 2004 to February 2005, consisting of 9358 observations altogether. Each response is themed into three classes of interest regarding CO, NO2, and RH.

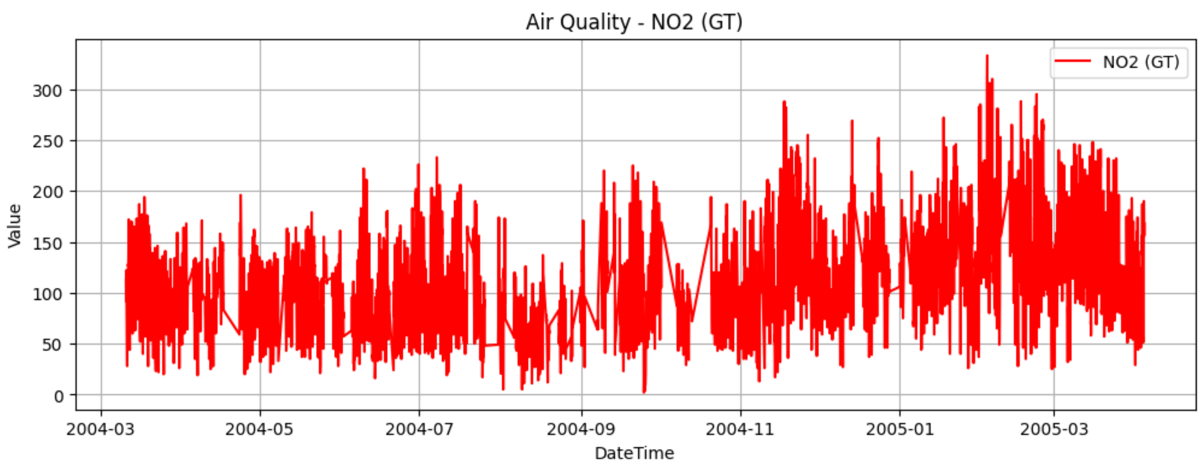
The time series structure of the data necessitated combining the Date and Time columns into a single Datetime column, which was then set as the index. The result was correct chronological order of data that would be suitable for any form of time series analysis and visualization. Preprocessing steps included dealing with missing values initially encoded as -200 that needed to be replaced by appropriate interpolated values to maintain continuity in the data.

Time Series Visualization

Each variable was individually plotted to inspect their temporal behavior:

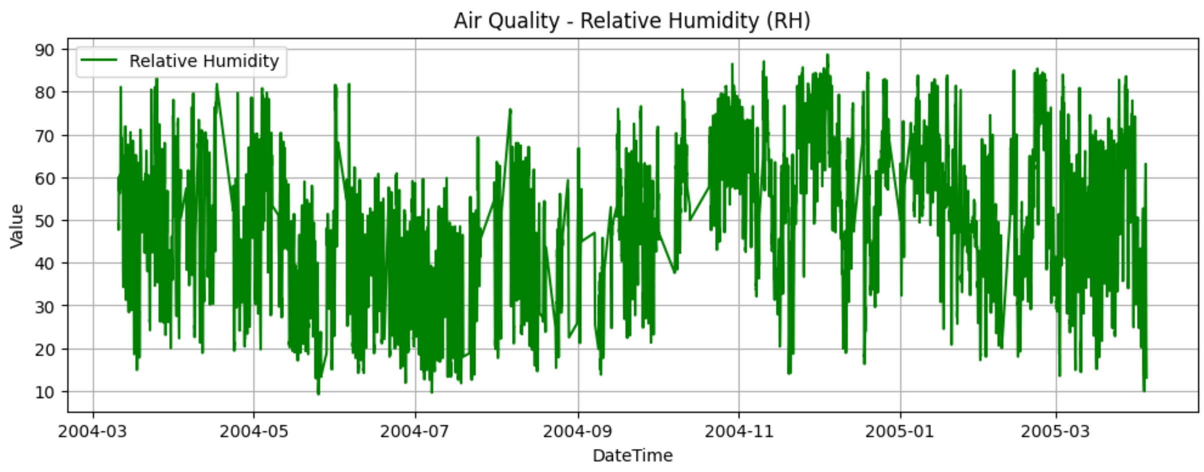


- CO (GT): This is a time series with a structure of clear fluctuations; peaks reflect episodes of higher carbon monoxide concentrations that might be related to a certain time of day or specific environmental conditions. Obvious seasonal and trend components are seen to vary with time.



- NO2 (GT): Similar to CO, the NO2 series is variable with peaks in the timeline. It also seems to have a periodic pattern; this might suggest that it is influenced by some external factors, such as traffic or industrial activities, in cycles.

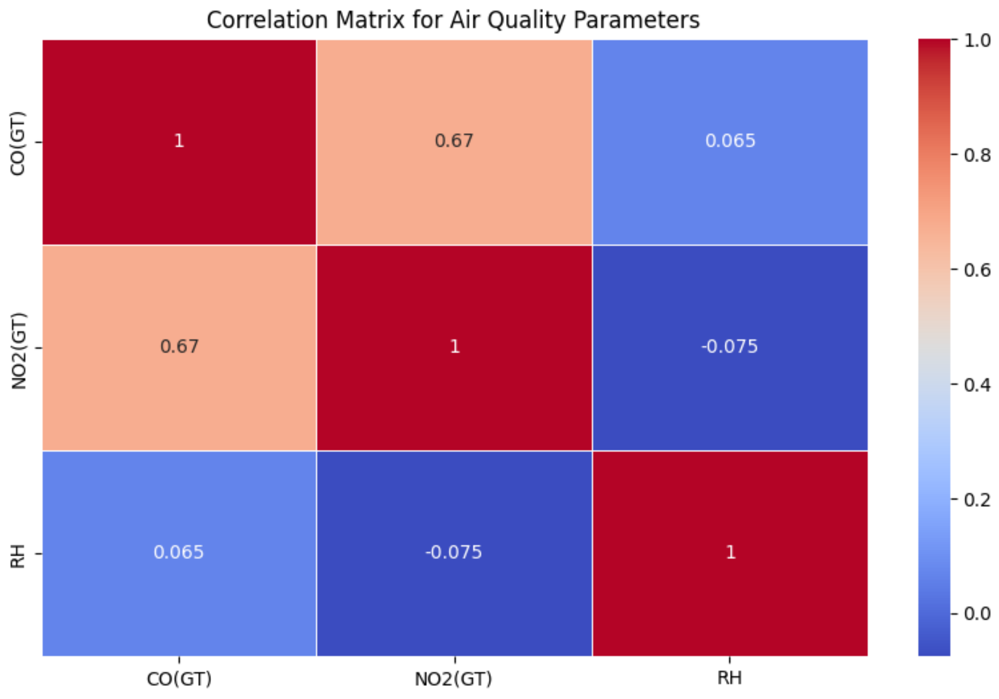
Candidate: 14



- RH: The time series of Relative Humidity does not reveal any apparent relation to the pollutant concentrations. Its variations are seemingly more irregular and more likely to be linked directly with meteorological factors rather than sources of pollution.

Correlation Analysis

To explore the relationships between the three variables, a correlation matrix was made, and the results were visualized using a heatmap.



	CO(GT)	NO2(GT)	RH
CO(GT)	1	0.67	0.065
NO2(GT)	0.67	1	-0.075
RH	0.065	-0.075	1

Observations from the Correlation Matrix:

CO and NO2: The strongest correlation (0.67) indicating that they two are connected, which could be from vehicle emissions or industrial activities.

RH and CO: An positive correlation (0.065), suggesting relative connection between humidity and carbon monoxide levels.

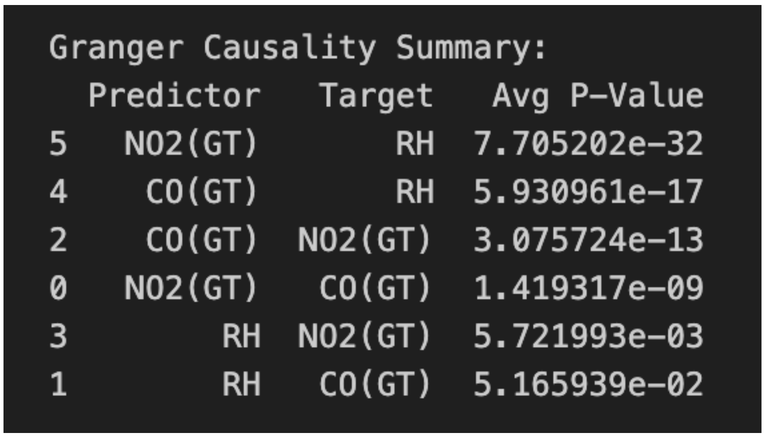
RH and NO2: A weak negative correlation (-0.075), further confirming that NO2 variations are largely independent of relative humidity.

These results highlight the strong interdependence of CO and NO2, with RH being more independent of the gaseous pollutants. The would suggest that predictive modeling efforts for CO and NO2 must consider their interconnected variability, while RH may require separate modeling considerations.

Task 2

Multivariate vs. Univariate Modeling

The Air Quality data set includes three main variables: CO(GT), NO2(GT), and RH, either represented separately as a univariate time series or combined as multivariate. As far as the choice of the best modeling approach is concerned, dependencies of variables were investigated using Granger Causality Tests and supported by the Correlation Analysis.



```
Granger Causality Summary:
  Predictor   Target   Avg P-Value
5   NO2(GT)       RH  7.705202e-32
4    CO(GT)       RH  5.930961e-17
2    CO(GT)  NO2(GT)  3.075724e-13
0   NO2(GT)   CO(GT)  1.419317e-09
3        RH  NO2(GT)  5.721993e-03
1        RH   CO(GT)  5.165939e-02
```

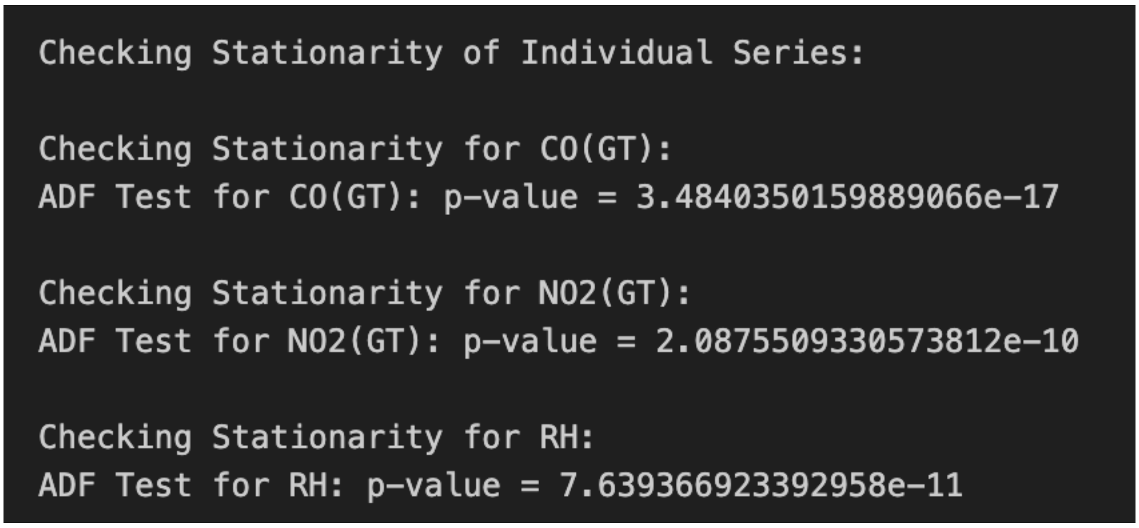
The Granger Causality Test revealed a relationships between CO and NO2, suggesting these two variables influence each other over time. This finding supports the use of a multivariate approach, such as a Vector Autoregressive (VAR) model, which has again supported the appropriateness of adopting a multivariate approach like the VAR model to capture interdependencies among variables. . On the other hand, RH presented very weak links and did not have any major causing effect on either CO or NO2. RH can be represented separately if needed.

- CO (GT) and NO2 (GT) have strong causal interactions, with an average p-value of 3.076 for CO causing NO2, and 1.419 for NO2 causing CO. This causal relationship implies that both variables have an impact on one another over time, which supports the use of a Vector Autoregressive (VAR) model in multivariate modeling.

- RH showed a weaker causal influence on CO (p-value = 5.166) and NO2 (p-value = 5.722), and vice versa. While the causality is statistically significant, the weak association suggests RH may not be valuable in understanding or predicting CO and NO2 concentrations.

Stationarity Testing

Stationarity is one of the basic assumptions in most models of a time series. The ADF tests were conducted on each variable to check for stationarity, and the findings proved that these three series were nonstationary with trend and seasonal components.



```
Checking Stationarity of Individual Series:

Checking Stationarity for CO(GT):
ADF Test for CO(GT): p-value = 3.4840350159889066e-17

Checking Stationarity for NO2(GT):
ADF Test for NO2(GT): p-value = 2.0875509330573812e-10

Checking Stationarity for RH:
ADF Test for RH: p-value = 7.639366923392958e-11
```

CO (GT):

- ADF p-value = 3.4840
- Since the p-value is much less than **0.05**, we reject the null hypothesis of the ADF test, which means the series is stationary.

NO2 (GT):

- ADF p-value = 2.0875
- Again, the p-value is less than **0.05**, so we reject the null hypothesis. The series is stationary.

RH:

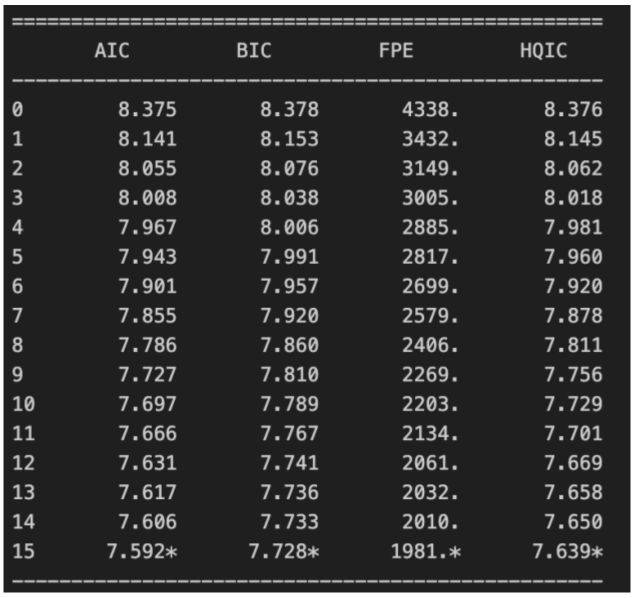
- ADF p-value = 7.6393
- With a p-value less than **0.05**, the series is stationary.

Task 3 – Analysis, modeling and predict*ion*

VAR Model Development and Order Selection

a Vector Autoregressive model has been developed to model the interdependencies between the three variables: CO (GT), NO2 (GT), and RH. First, the optimal lag order of the model was selected based on the statistical criteria, which include Akaike Information Criterion and

Bayesian Information Criterion. According to the VAR order selection, the optimal lag order is 15 because it minimized AIC, BIC, and other criteria.



```
==================================================
       AIC         BIC         FPE         HQIC
--------------------------------------------------
0        8.375       8.378       4338.       8.376
1        8.141       8.153       3432.       8.145
2        8.055       8.076       3149.       8.062
3        8.008       8.038       3005.       8.018
4        7.967       8.006       2885.       7.981
5        7.943       7.991       2817.       7.960
6        7.901       7.957       2699.       7.920
7        7.855       7.920       2579.       7.878
8        7.786       7.860       2406.       7.811
9        7.727       7.810       2269.       7.756
10       7.697       7.789       2203.       7.729
11       7.666       7.767       2134.       7.701
12       7.631       7.741       2061.       7.669
13       7.617       7.736       2032.       7.658
14       7.606       7.733       2010.       7.650
15      7.592*      7.728*      1981.*      7.639*
--------------------------------------------------
```

The results highlight lag 15 as the optimal choice, as indicated by the minimum AIC value.

Fitting the VAR Model

The VAR model was fitted to the training dataset using the selected lag order of 15. The results for the three equations (CO, NO2, and RH) insights that would be important to find more about the relationships between the variables:

CO (GT) Equation:

1. Lagged CO(GT) Values: In this series, autoregressive features are very prominent with large coefficients contributing from higher-order lags; that will indicate that the past values of CO(GT) are very important while deciding their future values.
2. The presence of different lags of NO2(GT) which impacts CO(GT) denotes that there is causality between the variables. For instance:
   - The coefficients for Lag 1 and Lag 2 of NO2(GT) are positive, indicating that a prior increase in NO2(GT) will be followed by an increased value of CO(GT).
3. Impact of Relative Humidity: The impact of the relative humidity on carbon monoxide GT varies, and few of the lags significantly contribute. Among those, lag 3 has a small positive influence.

NO2 (GT) Equation:

1. Lagged NO2 dependence: The equation hence underlines a marked autoregressive pattern; hence, the value of NO2 is to be forecasted based on its one-step lagged values.
2. CO-GT: Lagged values of CO-GT greatly contribute to NO2 GT, with wide coefficients at specific lags. The interaction between CO and NO2 is also underlined over time.
3. The impact of relative humidity, on the other hand, is much weaker and noisier than those of NO2 and CO, although some lags of the former are statistically significant.

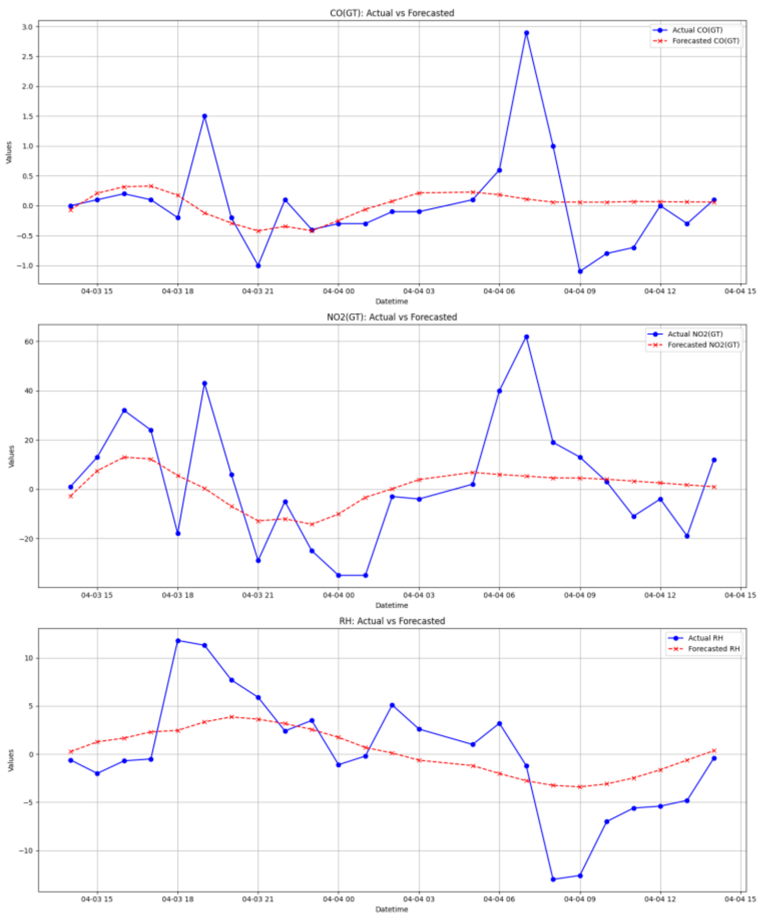
RH Equation:

1. Autoregressive Dependence: RH mainly depends on its past values since it portrays major coefficients for a wide range of lags. This result addresses the relative independence of RH in the VAR set against fluctuations in CO and NO2.
2. Influence of CO(GT) and NO2(GT): At the same time, many lags of CO(GT) and NO2(GT) are weaker, yet they also drive RH. These relationships indeed seem to be sparser and less predictable compared to the history of RH itself.

These results align with the findings from the Granger Causality Tests, which suggested strong interdependence between CO and NO2 and weaker influence from RH.

Forecasting and Evaluation

The model is used to forecast the next 24 periods in the test sets. Forecasts of CO (GT), NO2 (GT), and RH versus real measured values were plotted. The performance measures MAE and RMSE were calculated in order to check the performance regarding predictions.

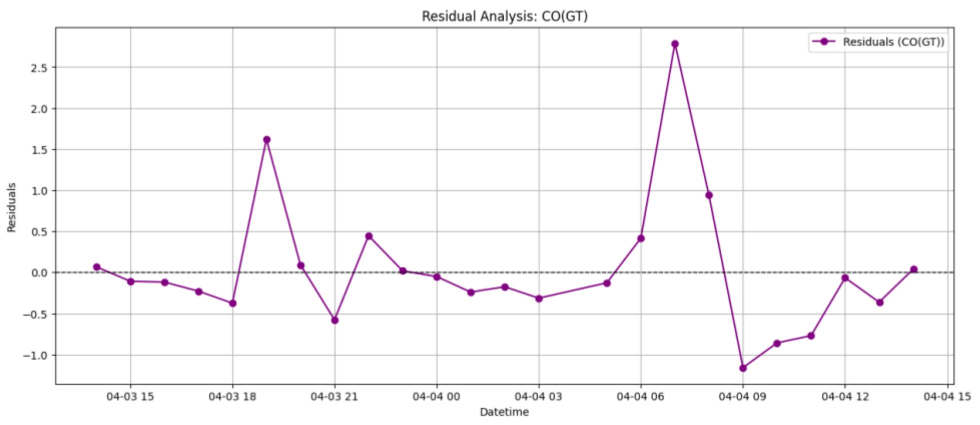


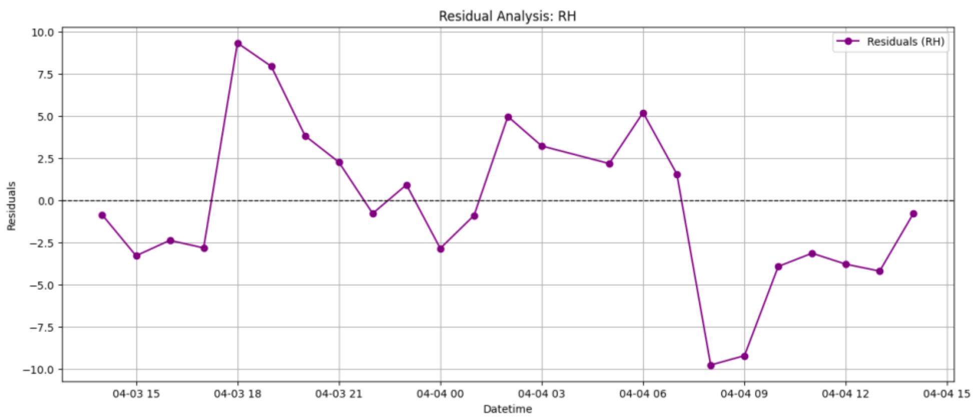
Forecast Visualization

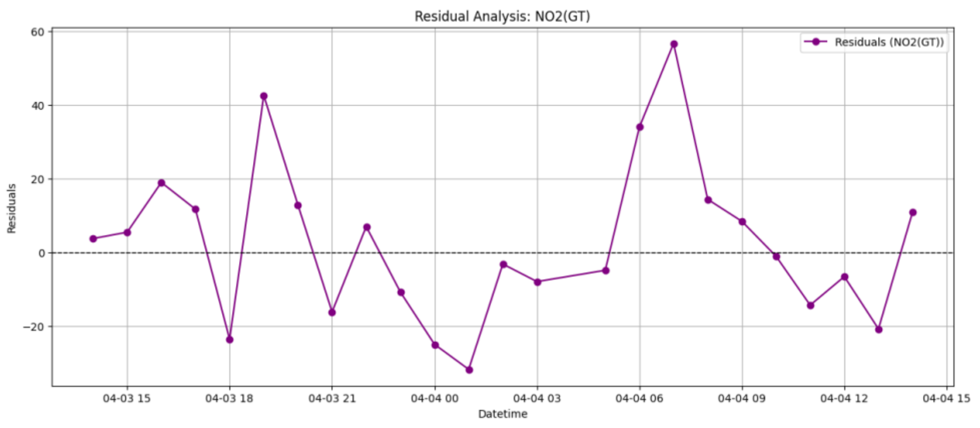
Observed and projected values for each variable were then plotted to show how well the model was able to capture the general trend and pattern of the data. Forecasts of both CO (GT) and NO2 (GT) followed measured values fairly well but with significant scatter on the more extreme values. The RH forecasts were in slightly higher disagreement, most likely because of its weaker relations with the other variables.

Residual Analysis

Residual analysis was done to check model assumptions and its performance. Residual plots of each variable were random noises around zero without large systematic biases. The residual correlation matrix was very low for the residuals of CO, NO2, and RH, which depicts that the relationship amongst themselves is pretty well captured by the VAR model itself.







Insights and Implications

Besides, the model VAR captured the proper dynamic interaction between CO and NO2, which is jointly of endogenous relationship representation in the process. Though the influence of RH was less strong with the other two series, yet it was adequately modeled out to make the overall analysis robust.

Some difficulties of the model to predict extreme values were seen, especially in RH and could support further development by including variables, such as temperature or wind speed.

As expected in any time series forecasting, the accuracy of forecasts decreased slightly for longer time horizons.

This was a powerful VAR model considering it used just lagged values and relations among variables to provide quite an accurate short-run forecast, especially for the level of CO and NO2 in this highly air-quality-dependent dataset. The effort underlines the importance of multivariate time series modeling in the perspective of environmental data for great insight into air quality monitoring and management.

Bonus Task - Analysis, modeling and prediction

This task was selected to explore whether simpler, variable-specific models could provide better predictions compared to the multivariate VAR model. The ARIMA models are also one of the most widely used in time-series forecasting because of their flexibility and accuracy in modeling a pattern in one variable. Unlike the VAR, focusing on an interaction of variables, ARIMA describes the dynamics of one series. This therefore provides a good avenue to focus the analysis on trend, seasonality, and noise.

The bonus task focused on applying ARIMA (AutoRegressive Integrated Moving Average) models individually to the time series data for CO(GT), NO2(GT), and RH. In contrast to the VAR model, which examines relationships among more than one variable, ARIMA is a kind of univariate modeling. This exercise aimed at verifying whether ARIMA could outperform the results obtained with the application of the VAR model in the individual forecast of the analyzed variables.

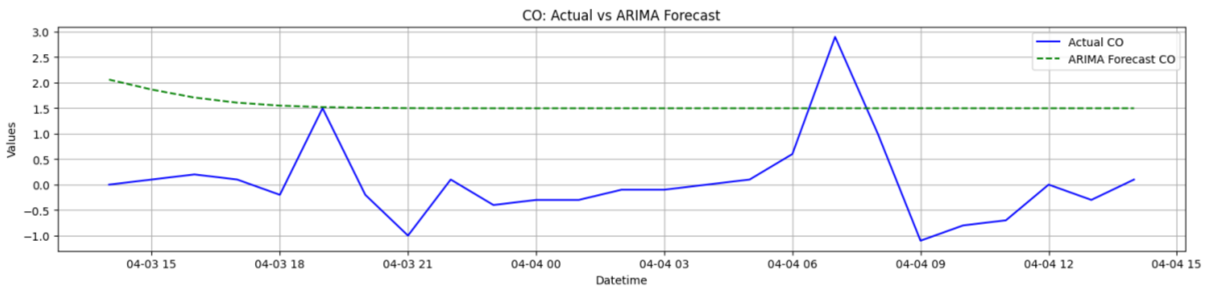
Model Summary and Parameters

1. **CO (GT): ARIMA(3,1,1)**
    - **AR (p):** 3
    - **Differencing (d):** 1
    - **MA (q):** 1
    - **AIC:** 16066.454

   - Significant parameters included the first three autoregressive terms and a single moving average term. The Ljung-Box test confirmed the residuals were white noise, indicating a good model fit.
2. **NO2 (GT): ARIMA(2,1,2)**
   - **AR (p):** 2
   - **Differencing (d):** 1
   - **MA (q):** 2
   - **AIC:** 61388.199
   - Both AR and MA terms were significant, and the Jarque-Bera test suggested non-normality in the residuals, likely due to extreme values or outliers.
3. **RH: ARIMA(4,1,2)**
   - **AR (p):** 4
   - **Differencing (d):** 1
   - **MA (q):** 2
   - **AIC:** 39764.355
   - The model captured the seasonality and trends effectively, but the high kurtosis indicated potential overfitting or the presence of anomalies.

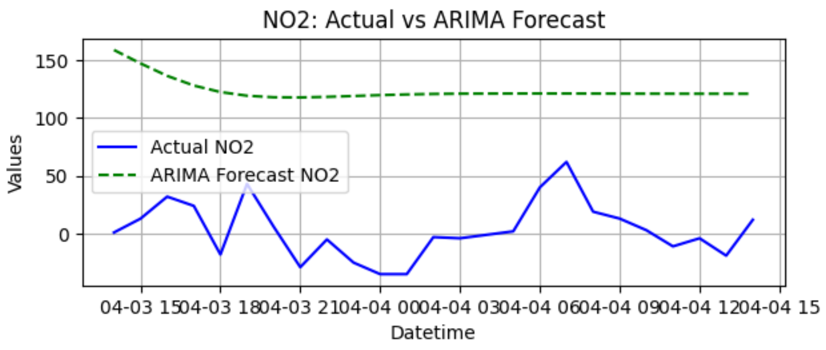
Forecast Results and Observations

1. **CO (GT):**



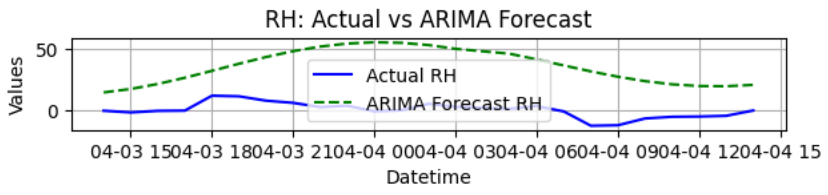
   - General trends were smoother for the ARIMA model forecast than the actual data, although it does capture these up and down trends, it does not show sharp peaks or fluctuations.
   - This mismatch may arise from ARIMA's limitations in handling sudden variability in time series data.

2. **NO2 (GT):**



- The forecast for NO2 showed a consistent downward drift, deviating significantly from the actual observed peaks and troughs. This would indicate that the ARIMA model struggled to accurately represent the volatile behavior of NO2 over time.

3. **RH**:



- hese indeed captured very regular seasonal trends in the RH forecasts, whereas ones from the data were highly spiky.
- This could also suggest that the ARIMA model may not have fully captured the influence of external factors, such as weather conditions.

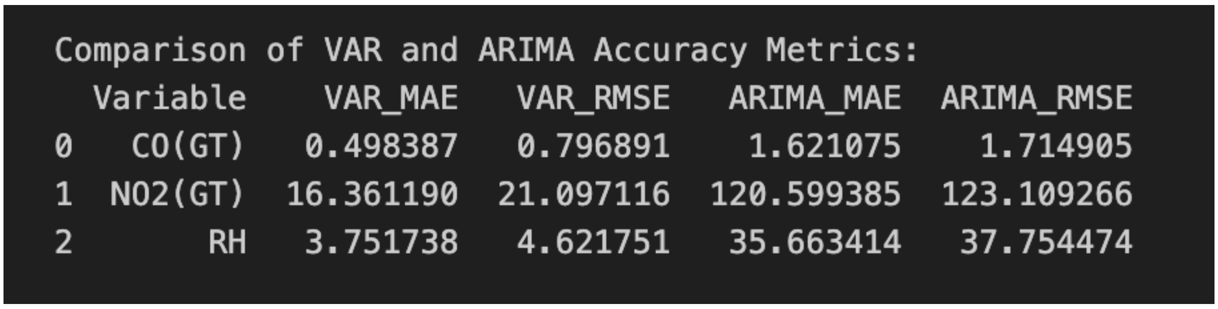
Strengths and Limitations of ARIMA Models

1. **Strengths**:
   - The ARIMA models provided reliable overall trends, especially for CO (GT) and RH.
   - Parameter selection via auto_arima was efficient, identifying the best configurations to minimize the AIC and ensure statistical adequacy.
2. **Limitations**:
   - The univariate nature of ARIMA ignores relationships between variables like for example, CO and NO2, limiting its ability to capture interdependencies.
   - Forecasts smoothed out extreme fluctuations, resulting in inaccuracies for volatile variables like NO2.

While the ARIMA models captured the general trend with understandable results, some shortcomings remained concerning intrinsic volatility in the dataset and the interactive nature among the variables. Further analysis of a multivariate approach will go higher precisions in the forecast and better reflect real-life relationships.

Task 4 – Results and Evaluation

This section provides an analysis of the performance of two different forecast methods, namely Vector Autoregression (VAR) and ARIMA-AutoRegressive Integrated Moving Average, in their ability to yield efficient forecasts for key air quality indicators such as CO (GT), NO2 (GT), and RH. These models were followed up with the assessment of their performances with the MAE and RMSE.



```
Comparison of VAR and ARIMA Accuracy Metrics:
  Variable    VAR_MAE   VAR_RMSE   ARIMA_MAE  ARIMA_RMSE
0   CO(GT)   0.498387   0.796891    1.621075    1.714905
1  NO2(GT)  16.361190  21.097116  120.599385  123.109266
2       RH   3.751738   4.621751   35.663414   37.754474
```

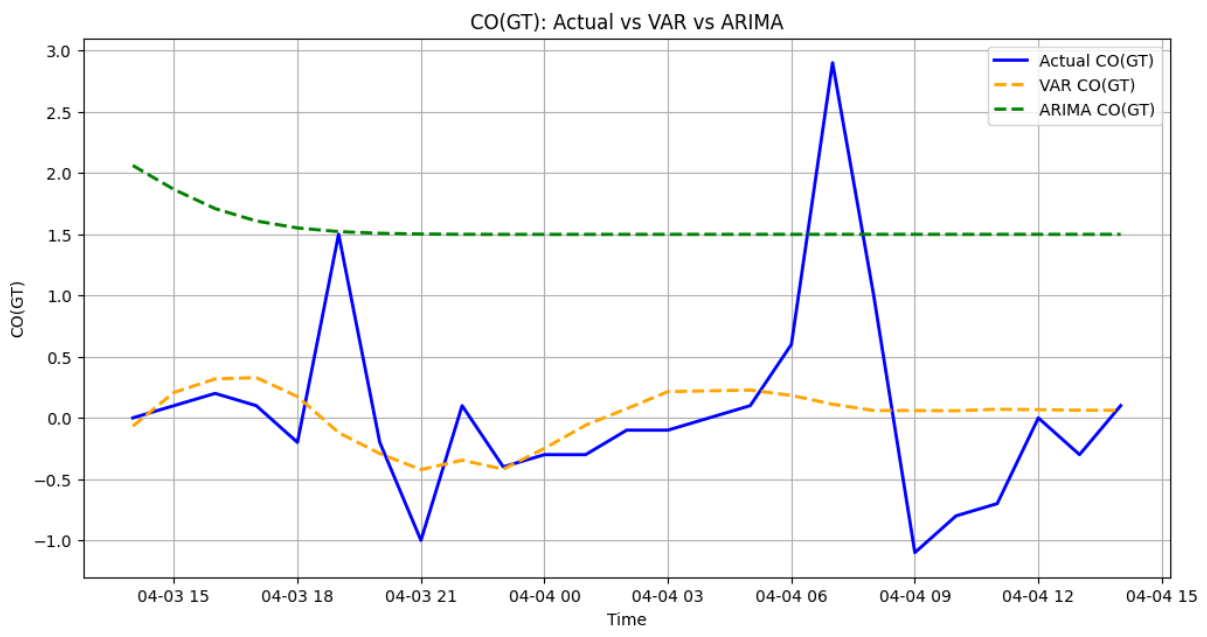
Analysis of Results

1. **CO (GT):**
    - The VAR model outperforms ARIMA significantly, with much lower MAE (0.4984 vs. 1.6211) and RMSE (0.79 vs. 1.71). This suggests that VAR had more ability to catch the short-run dynamics and relationships of variables influencing the levels of CO.
2. **NO2 (GT):**
    - While similar to CO, the VAR model proved to be more accurate, with much lower MAE and RMSE than ARIMA: 16.36 versus 120.59 and 21.0971 versus 123.1093, respectively.

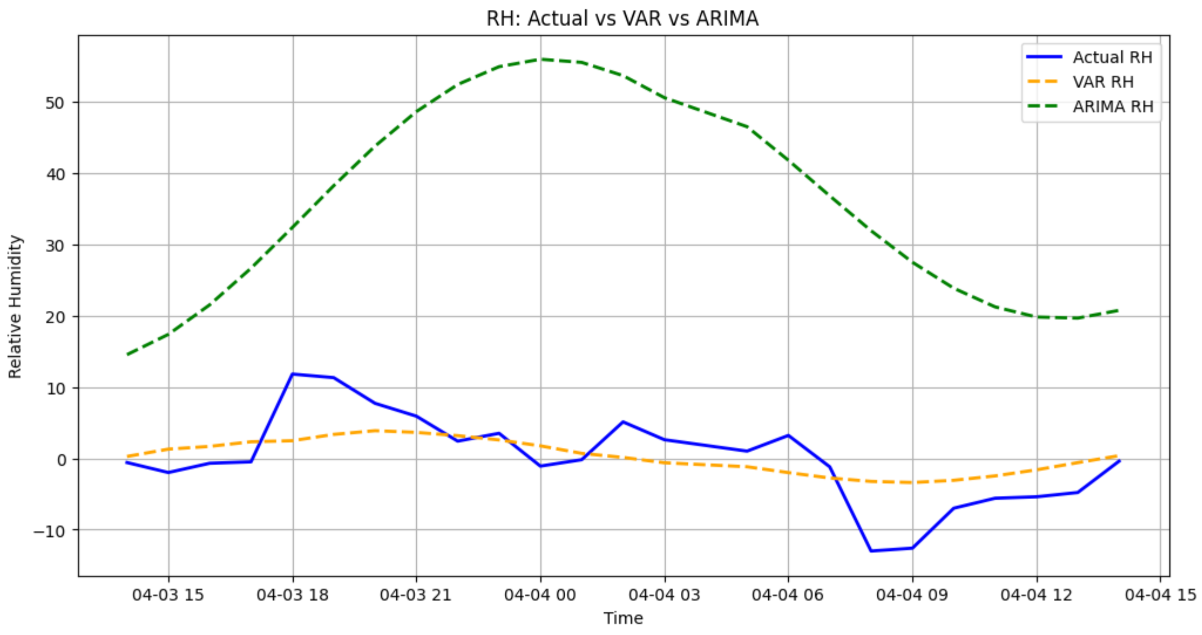
  - Large discrepancies in the forecasts of ARIMA could be a reason for not considering the strong cause-effect relationships of NO2 with other main factors like CO and RH.
3. **RH (Relative Humidity):**
  - The VAR outperformed the ARIMA model at an MAE of 3.7517 and RMSE of 4.6218, against the ARIMA MAE of 35.6634 and RMSE of 37.7545.
  - ARIMA's weaker performance for RH highlights its limited capacity to handle the volatility and inter-variable influences present in the dataset.

Graph Analysis: VAR vs. ARIMA Predictions

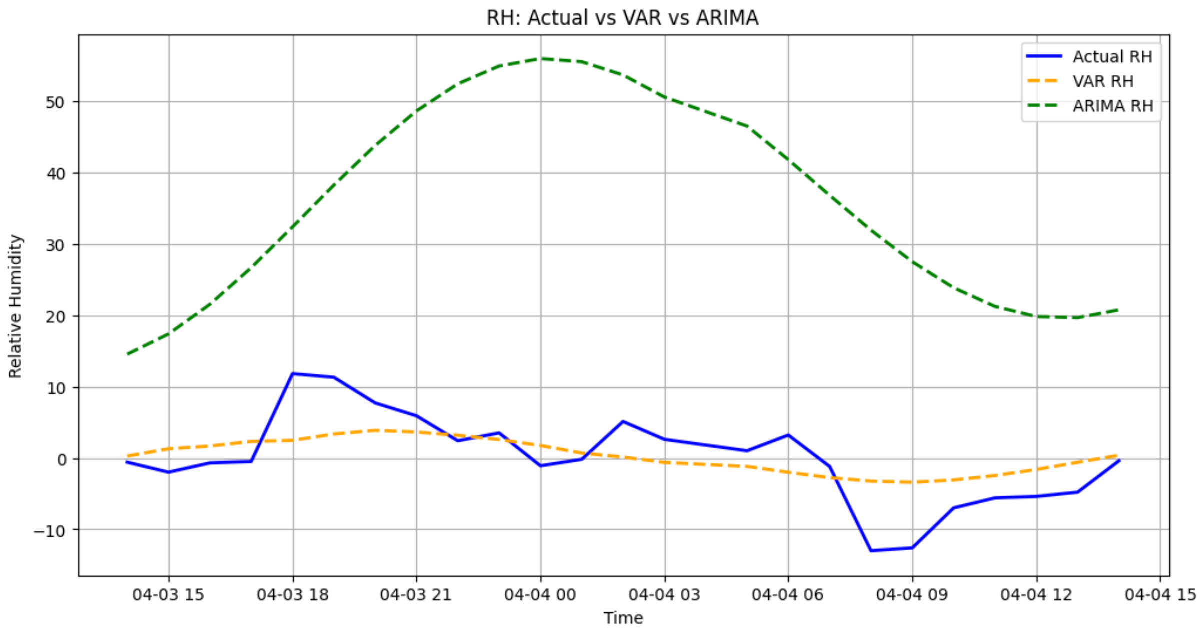
The graphs display the actual values (blue line), VAR predictions (orange dashed line), and ARIMA predictions (green dashed line):



1. **CO(GT)**: While VAR predictions capture the variations better, the actual valuesfollow them quite closely. Contrasting with that, ARIMA looks static and unable to adapt to the variations. That reflects the limitation of ARIMA when modeling variables influenced by dynamic interrelationships.



2. **NO2(GT)**: The model, though showing minor deviations, sets the trend of NO2 quitewell in the VAR model. ARIMA predictions remain constant and do notshow the sudden change seen in the actual NO2 levels.



3. **RH**: In terms of relative humidity, VAR gives better results for the real values, while ARIMA responds poorly to changes , sticking with a very basic form offorecasting.

Visual Comparison of Forecasts

The next forecast observations also support the fact that the VAR model strength will be enhanced:

- The VAR forecasts closely align with the observed values, capturing both trends and fluctuations in the data.
- In the other hand, ARIMA yielded an ultra-smoothed forecast which failed to capture many pronounced peaks in the actual data.

Interdependencies Matter:

- The relationships between the variables, such as strong causality present between CO and NO2, are exploited by the VAR model to give better forecasting.
-
- In contrast, ARIMA failed to capture these interdependencies and, through its univariate nature, had higher errors.

Short-Term vs. Long-Term Accuracy:

- VAR showed a better short-run accuracy, especially on volatile variables like NO2.

- ARIMA forecast showed smooth results, which may be helpful in analyzing the long-term trend yet not that good for the short-term.

Model Suitability:

- When the analysis deals with many interrelated time series variables, as many analyses of air quality do, VAR would be more appropriate since it models cross-variable interactions. Nevertheless, ARIMA could be useful in independent and univariate considerations and lacks robustness at that level to represent multivariate settings.

The comparative analysis showed that the VAR model performed consistently better in making forecasts of all three air quality parameters than ARIMA. VAR accounted for the inter-variable dynamics in all cases with much-reduced MAE and RMSE. The findings underpin that, in the cases in which variables may be interrelated, one should make consideration of multivariate approaches to forecasts by time series. Future refinements in the parameters of VAR or hybrid modelling may further help improve the accuracy in forecasts.

Conclusion

This study focuses on the application of time series analysis and forecasting techniques to air quality data, based on three key variables: carbon monoxide (CO) (GT), nitrogen dioxide (NO2) (GT), and relative humidity (RH). This encompasses strict data pre-processing, stationarity checking, causal analysis, and model implementation-both univariate (ARIMA) and multivariate (VAR). Each one of these activities contributed to a deep exploration of the dataset and identification of the best modeling approach.

The initial exploration highlighted the interdependencies among the variables, with CO and NO2 showing strong bi-directional causal relationships, whereas RH had a relatively weak causal link. The stationarity tests indicated that all the variables needed differencing to make them stationary, a prerequisite for any time series modeling. This helped in getting proper model development and forecasting.

Although ARIMA models perform well in detecting overall trends, the model faced some limitations in volatility response and intervariable relationships. As a result, the ARIMA predictions had a very smooth trend that led to increased error, in particular for NO2 and RH. Conversely, the VAR model made better utilization of the relationship among those variables to realize far more accurate forecasts of the concerned factors, as reflected by the small MAE and RMSE of all variables from the VAR model.

The comparison undertaken between VAR and ARIMA gave reason for the adoption of multivariate approaches when time series are interlinked within a dataset. The strength of VAR, capable of modeling cross-variable influences, outperformed ARIMA in both short-run accuracy and responsiveness to variation. In fact, the same was evidenced by the visualization of the forecast, where VAR predictions were closer to the observed values.

This analysis, therefore, concluded that multivariate approaches were superior, such as using VAR to make such interlinked dataset forecasts. But ARIMA is good to go for univariate time series but not in cases involving multiple interrelated variables. There are studies that will be covered by integrating hybrid models, which combine the strengths of ARIMA and VAR, or using machine learning techniques for further improvement in predictive performance to gain more depth understanding of the dynamics of air quality.

Sources

Nikki-priyaHIT. (n.d.). *GitHub - Nikki-PriyaHIT/Air-Quality-Prediction-Using-ARIMA-Model*. GitHub. https://github.com/nikki-priyaHIT/Air-Quality-Prediction-Using-ARIMA-Model?tab=readme-ov-file

Haripriya4work. (n.d.). *GitHub - haripriya4work/Air-Quality-Prediction: Air Quality Prediction using time series forecasting in python with models LSTM and ARIMA. Successfully predicted test values.* GitHub. https://github.com/haripriya4work/Air-Quality-Prediction

MachineLearningPlus (n.d.) - *ARIMA Model – Complete Guide to Time Series Forecasting in Python* https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/

Binh-Bk. (n.d.). *GitHub - binh-bk/air-quality-analysis: Jupyter notebooks and Python code for analyzing air quality (fine particle, PM2.5)*. GitHub. https://github.com/binh-bk/air-quality-analysis

Babcock, J. (2016). *Mastering Predictive Analytics with Python*. Packt Publishing.

Abbott, D. (2014). Applied Predictive Analytics: *Principles and Techniques for the Professional Data Analyst*. Wiley.

GeeksforGeeks. (2024, May 21). *Residual analysis*. GeeksforGeeks. https://www.geeksforgeeks.org/residual-analysis/

Gupta, A. (2024, November 21). *A complete guide to get a grasp of time series analysis*. Simplilearn.com. https://www.simplilearn.com/tutorials/statistics-tutorial/what-is-time-series-analysis

Candidate: 14

Air pollution: Everything you need to know. (2024, March 26). https://www.nrdc.org/stories/air-pollution-everything-you-need-know#effects

Katy. (2024, November 25). Performing “Granger Causality” with Python: Detailed Examples. *Medium*. https://medium.com/codex/performing-granger-causality-with-python-detailed-examples-3bca3fb1e1d2

Source code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from statsmodels.tsa.stattools import grangercausalitytests, adfuller, kpss
from statsmodels.tsa.api import VAR
from sklearn.metrics import mean_absolute_error, mean_squared_error
from pmdarima import auto_arima

df = pd.read_csv('/Users/hodanielkhuu/Library/CloudStorage/OneDrive-HøyskolenKristiania/School/3 år/PGR304 Predictive Analytics/Exam/DataSet_4_Exam/AirQualityUCI.csv', sep=';', decimal=',')

# Combine Date and Time columns
df['Datetime'] = df['Date'] + ' ' + df['Time']

# Explicitly specify the format of the datetime
df['Datetime'] = pd.to_datetime(df['Datetime'], format='%d/%m/%Y %H.%M.%S')
```

Candidate: 14

```
# Set as index
df.set_index('Datetime', inplace=True)

# Verify the changes
print(df.index)
print(df.head())

plt.style.use('default')

# Plot for CO (GT)
plt.figure(figsize=(12, 4))
plt.plot(data.index, data['CO(GT)'], label='CO (GT)', color='b')
plt.legend()
plt.title('Air Quality - CO (GT)')
plt.xlabel('DateTime')
plt.ylabel('Value')
plt.grid(True)
plt.show()

# Plot for NO2 (GT)
plt.figure(figsize=(12, 4))
plt.plot(data.index, data['NO2(GT)'], label='NO2 (GT)', color='r')
plt.legend()
plt.title('Air Quality - NO2 (GT)')
plt.xlabel('DateTime')
plt.ylabel('Value')
plt.grid(True)
plt.show()

# Plot for Relative Humidity (RH)
plt.figure(figsize=(12, 4))
plt.plot(data.index, data['RH'], label='Relative Humidity', color='g')
plt.legend()
plt.title('Air Quality - Relative Humidity (RH)')
plt.xlabel('DateTime')
plt.ylabel('Value')
plt.grid(True)
plt.show()


data = df[['CO(GT)', 'NO2(GT)', 'RH']]

# Handle Missing Values
print(f"Missing Values Before Cleaning:\n{data.isna().sum()}\n")
data.replace(-200, np.nan, inplace=True)
data.dropna(inplace=True)
print(f"Missing Values After Cleaning:\n{data.isna().sum()}\n")
data.dropna(inplace=True)
data.interpolate(method='time', inplace=True)  # Interpolate missing values to preserve
continuity

print("\nDataset Information:")
data.info()
print("\nFirst 5 Rows of Data:")
print(data.head())

# Statistical Summary
print("\nStatistical Summary of Dataset:")
print(data.describe())

correlations = data.corr()
print(correlations)

# Heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix for Air Quality Parameters')
plt.show()



# ADF - Stationarity Tests
def check_stationarity(timeseries, title):
    print(f"\nChecking Stationarity for {title}:")
    # Augmented Dickey-Fuller test
    adf_result = adfuller(timeseries)
    print(f"ADF Test for {title}: p-value = {adf_result[1]}")

# Check stationarity for each time series
print("\nChecking Stationarity of Individual Series:")
check_stationarity(data['CO(GT)'], 'CO(GT)')
check_stationarity(data['NO2(GT)'], 'NO2(GT)')
check_stationarity(data['RH'], 'RH')




def granger_causality_summary(data, max_lag=12):
    summary = []
    for target in data.columns:
        for predictor in data.columns:
            if target != predictor:
                test_result = grangercausalitytests(data[[target, predictor]].dropna(), max_lag,
verbose=False)
                p_values = [test_result[lag][0]['ssr_ftest'][1] for lag in range(1, max_lag + 1)]
                # Average p-value across lags
                avg_p_value = np.mean(p_values)
                summary.append({
                    'Predictor': predictor,
                    'Target': target,
                    'Avg P-Value': avg_p_value
                })
    return pd.DataFrame(summary).sort_values(by='Avg P-Value')
```

Candidate: 14

```
# Generate and Display Granger Causality Summary
gc_summary = granger_causality_summary(data, max_lag=12)
print("\nGranger Causality Summary:")
print(gc_summary)



train = data[:-24]
test = data[-24:]

# Ensure there are no NaN or infinite values in the training data
train = train[["CO(GT)", "NO2(GT)", "RH"]].dropna()
test = test[["CO(GT)", "NO2(GT)", "RH"]].dropna()

print("\nTrain Set Size:", train.shape)
print("Test Set Size:", test.shape)



model = VAR(train)
order_selection = model.select_order(maxlags=15)
print("\nSelected Order of VAR Model:")
print(order_selection.summary())

# Fit Model Using Selected Order
var_model = model.fit(order_selection.aic)
print("\nSummary of Fitted VAR Model:")
print(var_model.summary())



var_forecast = var_model.forecast(train.values[-var_model.k_ar:], steps=24)
var_forecast_df = pd.DataFrame(var_forecast, columns=train.columns, index=test.index)

# Define Evaluation Function
def evaluate_forecast(true_values, forecast_values, label):
    mae = mean_absolute_error(true_values, forecast_values)
    rmse = np.sqrt(mean_squared_error(true_values, forecast_values))
    print(f"{label} - MAE: {mae:.2f}, RMSE: {rmse:.2f}")

# Evaluate forecasts for each variable
evaluate_forecast(test['CO(GT)'], var_forecast_df['CO(GT)'], 'CO(GT)')
evaluate_forecast(test['NO2(GT)'], var_forecast_df['NO2(GT)'], 'NO2(GT)')
evaluate_forecast(test['RH'], var_forecast_df['RH'], 'RH')

# Plotting Actual vs Forecast
plt.figure(figsize=(15, 18))

# CO(GT)
plt.subplot(3, 1, 1)
plt.plot(test.index, test['CO(GT)'], label='Actual CO(GT)', color='blue', marker='o')
plt.plot(var_forecast_df.index, var_forecast_df['CO(GT)'], label='Forecasted CO(GT)',
linestyle='--', color='red', marker='x')
plt.legend()
plt.title('CO(GT): Actual vs Forecasted')
plt.xlabel('Datetime')
plt.ylabel('Values')
plt.grid()

# NO2(GT)
plt.subplot(3, 1, 2)
plt.plot(test.index, test['NO2(GT)'], label='Actual NO2(GT)', color='blue', marker='o')
plt.plot(var_forecast_df.index, var_forecast_df['NO2(GT)'], label='Forecasted NO2(GT)',
linestyle='--', color='red', marker='x')
plt.legend()
plt.title('NO2(GT): Actual vs Forecasted')
plt.xlabel('Datetime')
plt.ylabel('Values')
plt.grid()
```

Candidate: 14

```
# RH
plt.subplot(3, 1, 3)
plt.plot(test.index, test['RH'], label='Actual RH', color='blue', marker='o')
plt.plot(var_forecast_df.index, var_forecast_df['RH'], label='Forecasted RH', linestyle='--',
color='red', marker='x')
plt.legend()
plt.title('RH: Actual vs Forecasted')
plt.xlabel('Datetime')
plt.ylabel('Values')
plt.grid()

plt.tight_layout()
plt.show()

def plot_residuals(actual, predicted, title):
    residuals = actual.values - predicted.values
    plt.figure(figsize=(15, 6))
    plt.plot(actual.index, residuals, label=f'Residuals ({title})', color='purple', marker='o')
    plt.axhline(y=0, color='black', linestyle='--', linewidth=1)
    plt.title(f'Residual Analysis: {title}')
    plt.xlabel('Datetime')
    plt.ylabel('Residuals')
    plt.legend()
    plt.grid()
    plt.show()

# Plot residuals for each variable
plot_residuals(test['CO(GT)'][-24:], var_forecast_df['CO(GT)'], 'CO(GT)')
plot_residuals(test['NO2(GT)'][-24:], var_forecast_df['NO2(GT)'], 'NO2(GT)')
plot_residuals(test['RH'][-24:], var_forecast_df['RH'], 'RH')


co_arima = auto_arima(data['CO(GT)'], seasonal=False, trace=True)
no2_arima = auto_arima(data['NO2(GT)'], seasonal=False, trace=True)
```

```
rh_arima = auto_arima(data['RH'], seasonal=False, trace=True)

# Forecast for each model (24 steps)
co_forecast_arima = co_arima.predict(n_periods=24)
no2_forecast_arima = no2_arima.predict(n_periods=24)
rh_forecast_arima = rh_arima.predict(n_periods=24)

# ARIMA forecasts for each variable
arima_forecasts = {
    'CO(GT)': co_forecast_arima,
    'NO2(GT)': no2_forecast_arima,
    'RH': rh_forecast_arima
}

arima_forecast_df = pd.DataFrame(arima_forecasts, index=test.index)



# Print ARIMA Model Summary
print("\nSummary of ARIMA Models:")
print(co_arima.summary())
print(no2_arima.summary())
print(rh_arima.summary())

plt.figure(figsize=(15, 10))

# CO(GT) Plot
plt.subplot(3, 1, 1)
plt.plot(test.index[-24:], test['CO(GT)'][-24:], label='Actual CO', color='blue')
plt.plot(arima_forecast_df.index[-24:], co_forecast_arima, label='ARIMA Forecast CO',
linestyle='--', color='green')
plt.legend()
plt.title('CO: Actual vs ARIMA Forecast')
plt.xlabel('Datetime')
plt.ylabel('Values')
plt.grid()

# NO2(GT) Plot
plt.subplot(2, 1, 2)
plt.plot(test.index[-24:], test['NO2(GT)'][-24:], label='Actual NO2', color='blue')
plt.plot(arima_forecast_df.index[-24:], no2_forecast_arima, label='ARIMA Forecast NO2',
linestyle='--', color='green')
plt.legend()
plt.title('NO2: Actual vs ARIMA Forecast')
plt.xlabel('Datetime')
plt.ylabel('Values')
plt.grid()

# RH Plot
plt.subplot(4, 1, 2)
plt.plot(test.index[-24:], test['RH'][-24:], label='Actual RH', color='blue')
plt.plot(arima_forecast_df.index[-24:], rh_forecast_arima, label='ARIMA Forecast RH',
linestyle='--', color='green')
plt.legend()
plt.title('RH: Actual vs ARIMA Forecast')
plt.xlabel('Datetime')
plt.ylabel('Values')
plt.grid()

plt.tight_layout()
plt.show()

# Plot for CO(GT)
plt.figure(figsize=(12, 6))
plt.plot(test.index, test["CO(GT)"], label="Actual CO(GT)", color="blue", linewidth=2)
plt.plot(test.index, var_forecast_df["CO(GT)"], label="VAR CO(GT)", color="orange",
linestyle="--", linewidth=2)
plt.plot(test.index, arima_forecasts["CO(GT)"], label="ARIMA CO(GT)", color="green",
linestyle="dashed", linewidth=2)
```

```
plt.title("CO(GT): Actual vs VAR vs ARIMA")
plt.xlabel("Time")
plt.ylabel("CO(GT)")
plt.legend()
plt.grid()
plt.show()

# Repeat for NO2(GT)
plt.figure(figsize=(12, 6))
plt.plot(test.index, test["NO2(GT)"], label="Actual NO2(GT)", color="blue", linewidth=2)
plt.plot(test.index, var_forecast_df["NO2(GT)"], label="VAR NO2(GT)", color="orange",
linestyle="--", linewidth=2)
plt.plot(test.index, arima_forecasts["NO2(GT)"], label="ARIMA NO2(GT)", color="green",
linestyle="dashed", linewidth=2)
plt.title("NO2(GT): Actual vs VAR vs ARIMA")
plt.xlabel("Time")
plt.ylabel("NO2(GT)")
plt.legend()
plt.grid()
plt.show()

# Repeat for RH
plt.figure(figsize=(12, 6))
plt.plot(test.index, test["RH"], label="Actual RH", color="blue", linewidth=2)
plt.plot(test.index, var_forecast_df["RH"], label="VAR RH", color="orange", linestyle="--",
linewidth=2)
plt.plot(test.index, arima_forecasts["RH"], label="ARIMA RH", color="green",
linestyle="dashed", linewidth=2)
plt.title("RH: Actual vs VAR vs ARIMA")
plt.xlabel("Time")
plt.ylabel("Relative Humidity")
plt.legend()
plt.grid()
plt.show()
```

```
metrics = []

for column in test.columns:
    # VAR Metrics
    var_mae = mean_absolute_error(test[column], var_forecast_df[column])
    var_rmse = np.sqrt(mean_squared_error(test[column], var_forecast_df[column]))

    # ARIMA Metrics
    arima_mae = mean_absolute_error(test[column], arima_forecasts[column])
    arima_rmse = np.sqrt(mean_squared_error(test[column], arima_forecasts[column]))

    # Append results to metrics list
    metrics.append({
        "Variable": column,
        "VAR_MAE": var_mae,
        "VAR_RMSE": var_rmse,
        "ARIMA_MAE": arima_mae,
        "ARIMA_RMSE": arima_rmse
    })

# Convert metrics to DataFrame
metrics_df = pd.DataFrame(metrics)

# Display the results
print("\nComparison of VAR and ARIMA Accuracy Metrics:")
print(metrics_df)
```