**AIR QUALITY ANALYSIS USING**

**PYTHON**

**AIR QUALITY ANALYSIS**

**G.GOWRI [812121106008]**

**Introduction:**

Air quality analysis is a critical field of environmental science and public health that focuses on assessing and monitoring the composition of the Earth's atmosphere to determine the presence and concentration of various pollutants and contaminants. It plays a vital role in understanding and addressing the impact of air pollution on human health, ecosystems, and the environment. This introduction provides an overview of the key aspects and importance of air quality analysis.

**Overview of the process:**

The following is an overview of the process of building a air quality analysis model by performing different activities like feature engineering, model training and evaluation.

**Model training:**

Training the model with Prophet is really easy. The team copied the mechanism used is scikit packages: fit() and predict()

## Creating a model:

1. introduce to basic setup of folder, install pandas , matplotlib , seaborn (using pip for Python package), Anaconda is a good choice if you are using Windows (or even Mac, Linux). ...
2. basic use of those tools (clean, explore, plot, interpret)
3. work with a CSV file from Airnow.gov.

**Feature engineering:**

Feature engineering is a crucial step in any data analysis or machine learning project, including air quality analysis. It involves creating new features or transforming existing ones to better represent the underlying patterns in the data. In the context of air quality analysis, the goal is to extract relevant information from various data sources, such as sensors, weather data, and geographical information, to improve the accuracy of predictive models or gain better insights. Here are some feature engineering ideas for air quality analysis:

**Temporal Features:**

Time of Day: Extract hour and day of the week information from the timestamp to account for diurnal and weekly variations.

Time Lags: Create lag features to capture historical trends and autocorrelation in air quality data.

**Meteorological Features:**

Weather Data: Include weather conditions like temperature, humidity, wind speed, and precipitation, as these can influence air quality.

Wind Direction: Convert wind direction to categorical variables (e.g., N, S, E, W) or create polar coordinates.

Seasonal Indicators: Encode seasons or climatic seasons (e.g., summer, winter) based on the calendar date.

**Geospatial Features:**

Location Information: If you have data from multiple monitoring stations, encode station locations or distances to specific landmarks.

Geographical Clusters: Group stations or areas into clusters based on proximity, and use cluster labels as features.

**Historical Features:**

Rolling Statistics: Compute rolling mean, median, standard deviation, or other statistical measures over a specific time window to capture short-term trends.

Historical Max and Min: Track the historical maximum and minimum values of air quality indicators.

**Categorical Features:**

Day of the Week: Convert the day of the week to a categorical feature.

Public Holidays: Include a binary variable indicating whether a particular day is a public holiday.

**Interactions:**

Create interaction terms between relevant features. For example, the interaction between temperature and humidity can affect air quality.

**Cyclical Encoding:**

For cyclical features like time, use techniques like circular encoding (e.g., sine and cosine transforms) to handle periodic patterns.

**Outlier Detection:**

Create binary indicators for extreme values or outliers in the air quality data. These can be used to capture unusual events.

**Feature Scaling and Normalization:**

Standardize or normalize features to ensure that they are on a similar scale, especially if you are using algorithms sensitive to feature scaling, like k-means or SVM.

**Dimensionality Reduction:**

Use dimensionality reduction techniques such as Principal Component Analysis (PCA) or t-SNE to reduce the number of features while preserving essential information.

**Textual Features:**

If you have textual data related to air quality events or reports, consider using natural language processing techniques to extract relevant information from text.

**External Data:**

Integrate external data sources like traffic data, population density, industrial activities, or land use data if they are known to impact air quality.

Remember that the choice of features and feature engineering techniques depends on the specific characteristics of your dataset and the goals of your air quality analysis. Careful feature selection and engineering can significantly enhance the performance and interpretability of your models.

**Model evaluation:**

Evaluating a model for air quality analysis is crucial to ensure that it provides accurate and reliable predictions or insights. Here's a general framework for evaluating a model used in air quality analysis:

1. Data Preparation and Splitting:

- Collect and preprocess your air quality data. This includes handling missing values, data scaling, and encoding categorical variables.

- Split your data into training, validation, and test sets to evaluate the model's performance.

2. Choose Evaluation Metrics:

- Select appropriate evaluation metrics based on the nature of your air quality analysis. Common metrics include Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and R-squared (R2).

- Consider specific metrics for air quality, such as the Air Quality Index (AQI) or pollutant-specific metrics like PM2.5 levels.

3. Model Selection:

- Choose the appropriate machine learning or statistical model for your air quality analysis. This might include linear regression, decision trees, random forests, support vector machines, or deep learning models like neural networks.

4. Training and Validation:

- Train your model on the training data.

- Use the validation set to fine-tune hyperparameters and detect overfitting or underfitting.

5. Cross-Validation:

- In cases where you have limited data, consider using k-fold cross-validation to get a better estimate of your model's performance.

6. Model Evaluation:

- Evaluate your model's performance on the test set using the chosen evaluation metrics.

- Analyze the residuals (the differences between predicted and actual values) to understand where the model might be making errors.

7. Visualizations:

- Create visualizations, such as scatter plots, time series plots, or histograms, to visualize the model's predictions against actual data.

8. Feature Importance:

- If applicable, analyze feature importance to understand which variables have the most significant impact on air quality predictions. This can help in feature selection or data interpretation.

9. Domain Expert Review:

- Involve domain experts to assess the model's outputs and understand the practical implications of the results. They can provide valuable insights and help identify any model biases.

10. Model Interpretability:

- Depending on the model used, consider using techniques like SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) to interpret the model's predictions.

11. Model Deployment and Monitoring:

- If your model is for real-time air quality monitoring, deploy it in a production environment and set up monitoring to ensure it continues to perform well over time. Make necessary updates as needed.

12. Documentation and Reporting:

- Document the entire modeling process, including data sources, preprocessing, model selection, and evaluation results. Share these findings and insights with stakeholders and the wider community.

13. Continuous Improvement:

- Air quality models may need regular updates to adapt to changing environmental conditions. Continuously gather new data, retrain the model, and refine the analysis process.

Remember that the specific evaluation process may vary based on the objectives of your air quality analysis, the data available, and the complexity of the models used. It's important to choose an appropriate model, fine-tune it, and thoroughly validate its performance to ensure reliable air quality predictions.

**Visualization:**

Visualizing air quality analysis is essential for understanding and communicating data effectively. Here are some common ways to visualize air quality data:

1.Time Series Plots: Time series plots are useful for displaying changes in air quality over time. You can use line charts to show the variations in pollutants (e.g., PM2.5, PM10, NO2) over hours, days, or months.

2.Heatmaps: Heatmaps can be used to show spatial variations in air quality. Color-coding the map based on pollutant concentration levels can quickly highlight areas with poor air quality.

3.Bar Charts: Bar charts are useful for comparing air quality parameters between different locations or over different time periods. You can use horizontal or vertical bars to represent pollutant concentrations.

4. Pie Charts: Pie charts are useful for showing the composition of different pollutants in the air. You can create a pie chart to represent the percentage of each pollutant in the total air quality.

5. Box Plots: Box plots provide a summary of the distribution of air quality data, including median, quartiles, and outliers. They are useful for identifying variations and anomalies.

6.Scatter Plots: Scatter plots are valuable for examining relationships between air quality variables. For example, you can create a scatter plot to show the correlation between temperature and ozone levels.

7. Histograms: Histograms can be used to visualize the distribution of a single air quality parameter. They are especially helpful for understanding the frequency of different concentration ranges.

8. Geospatial Maps: Use geospatial maps to display air quality data over a geographic area. Geographic Information Systems (GIS) can help create interactive maps that allow users to explore air quality across different locations.

9. Animated Visualizations: Animation can be used to show how air quality changes over time. For instance, you can create an animation that illustrates the movement of pollutants over the course of a day or a year.

10. Radar Charts: Radar charts can be useful for comparing air quality parameters across different criteria. For example, you can use a radar chart to compare air quality at different monitoring stations based on multiple pollutants.

11.3D Plots: In cases where you have data in three dimensions (e.g., time, location, and pollutant concentration), 3D plots can provide a comprehensive view of the data.

12. Dashboard: Interactive dashboards, often created with tools like Tableau or Power BI, can combine multiple visualization types to provide a comprehensive view of air quality data. Users can filter, explore, and gain insights from the data interactively.

13. Word Clouds: Word clouds can be used to highlight frequently mentioned words or phrases related to air quality in textual data, such as social media comments or news articles.

14. Color-Coded Maps: Use color gradients on maps to represent different levels of air quality. For example, green can indicate good air quality, while red can indicate poor air quality.

15. Violin Plots: Violin plots combine aspects of box plots and kernel density estimation to provide a more detailed view of the distribution of air quality data.

The choice of visualization depends on your data, objectives, and audience. It's often useful to use a combination of these visualization techniques to provide a comprehensive understanding of air quality analysis. Additionally, you can use various data visualization software and programming libraries (e.g., Matplotlib, ggplot2, D3.js) to create these visualizations.

**PROGRAM:**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load your pollution data into a Pandas DataFrame (assuming it's in a CSV file)

data = pd.read\_csv("pollution\_data.csv")

# Data preprocessing (cleaning and grouping)

# Assuming the DataFrame has columns: 'City', 'SO2', 'NO2', 'RSPM\_PM10'

# Replace with your actual column names

data = data.dropna() # Remove rows with missing values

grouped\_data = data.groupby('City').mean() # Calculate average values for each city

# Data visualization

plt.figure(figsize=(12, 6))

# Bar chart for SO2 levels

plt.subplot(131)

sns.barplot(x=grouped\_data.index, y=grouped\_data['SO2'])

plt.title("Average SO2 Levels")

# Bar chart for NO2 levels

plt.subplot(132)

sns.barplot(x=grouped\_data.index, y=grouped\_data['NO2'])

plt.title("Average NO2 Levels")

# Bar chart for RSPM/PM10 levels

plt.subplot(133)

sns.barplot(x=grouped\_data.index, y=grouped\_data['RSPM\_PM10'])

plt.title("Average RSPM/PM10 Levels")

plt.tight\_layout()

plt.show()

**Conclusion:**

Drawing a conclusion from an air quality analysis requires careful consideration of the data and the context in which it was collected. Here's a general outline of how to draw conclusions from an air quality analysis.