PROJECT DOCUMENTATION AND

SUBMISSION

AIR QUALITY ANALYSIS AND

PREDICTION IN TAMILNADU

Project Objectives:-

The objective of this project is to analyze historical air quality data and develop a predictive model for air quality in the state of Tamil Nadu. This initiative is crucial to provide residents and government agencies with information to mitigate health risks and make informed decisions regarding air quality.

* Data Collection:- Gathered air quality data from multiple sources, including government monitoring stations, satellite imagery, and weather stations.
* Data Pre processing:- Integrated data, performed data cleaning, feature engineering, and normalization to prepare it for analysis.
* Innovative Solutions:- Implemented IoT sensor networks, machine learning ensembles, hybrid models, mobile applications, and social media integration.
* Big Data Analytics:- Employed big data analytics technologies to handle a substantial volume of data efficiently.

Data set Link:-

https://tn.data.gov.in/resource/location-wise-daily-ambient-air-quality-tamil-nadu-year-2014

Understanding the Problem:-

Data Collection:-

To begin with, we will gather historical air quality data from various monitoring stations across Tamil Nadu. The data should include parameters such as PM2.5, PM10, CO, SO2, NO2, O3, temperature, humidity, wind speed, and wind direction.

Data Pre-processing:-

* Data Cleaning:- Remove missing values and outliers.
* Data Integration:-Combine data from different sources.
* Feature Engineering:- Create new features if necessary.
* Data Normalization:- Scale data to a common range.
* Temporal Aggregation:- Aggregate data at appropriate time interval.

Analysis Approach of Air Quality:-

* Exploratory Data Analysis (EDA):-

1. Visualize data to identify trends and patterns.

2. Conduct statistical analysis to understand data distributions.

* Feature Selection:-

1. Identify key features that impact air quality.

2. Utilize correlation analysis and feature importance techniques.

* Model Selection:-

1. Select appropriate machine learning models for prediction (e.g., Regression, Time Series Models).

2. Train multiple models and compare their performance.

* Model Evaluation:-

1. Split data into training and testing sets.

2. Evaluate models using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared.

3. Tune hyper parameters for the best model.

* Prediction and Visualization:-

1. Deploy the trained model for real-time or near-real-time air quality prediction.

2. Visualize predictions through user-friendly interfaces like web dashboards and mobile apps.

* Model Maintenance:-

1. Implement regular model updates to ensure accuracy.

2. Continuously monitor data quality and model performance.

Expected Outcomes:-

1. Accurate air quality predictions for various locations in Tamil Nadu.

2. User-friendly interfaces for public access.

3. Timely alerts and recommendations to address air quality concerns.

Conclusion:-

This document outlines the understanding of the problem and the proposed approach for air quality analysis and prediction in Tamil Nadu. The success of this project will depend on data quality, model selection, and ongoing maintenance to provide valuable insights to the public and relevant authorities.

Innovative Solutions:-

* IoT Sensor Networks:-

To enhance data collection, we will implement an Internet of Things (IoT) sensor network. These sensors will be strategically placed throughout Tamil Nadu, providing real-time data to our analysis system. The IoT network will allow for more precise and granular data collection, contributing to accurate predictions.

* Machine Learning Ensembles:-

Incorporating machine learning ensembles, such as Random Forests and Gradient Boosting, will enable us to harness the collective power of multiple models. By aggregating the predictions from different models, we can achieve more robust and reliable air quality forecasts.

* Hybrid Models:-

We will explore hybrid models that combine traditional statistical methods with deep learning techniques like Long Short-Term Memory (LSTM) networks. This combination will capture both long-term trends and short-term fluctuations in air quality, improving prediction accuracy.

* Mobile Applications:-

We will develop mobile applications for Android and iOS platforms to provide users with real-time air quality information. These apps will not only display air quality levels but also offer recommendations for outdoor activities, health precautions, and alerts for extreme conditions.

* Social Media Integration:-

Leveraging social media platforms for data gathering and dissemination will enhance community involvement. Users can report air quality observations through social media, allowing us to crowd source data and engage the public in air quality management.

* Integration of Environmental Factors:-

In addition to air quality data, we will consider environmental factors like weather conditions and industrial activities. Incorporating weather data will help us understand how meteorological conditions affect air quality. By monitoring industrial activities, we can assess their impact and recommend emission control measures.

* Big Data Analytics:-

Employing big data analytics technologies will allow us to handle the substantial volume of data generated by IoT sensors and other sources efficiently. This approach enables real-time data processing and predictive modeling at scale.

* Continuous Research and Development:-

Innovation doesn't stop with the initial implementation. We commit to ongoing research and development to stay at the forefront of air quality analysis and prediction. This includes exploring new data sources, refining models, and adapting to emerging technologies.

Conclusion:-

The incorporation of innovative solutions is essential for solving the problem of air quality analysis and prediction in Tamil Nadu. By integrating IoT sensors, machine learning ensembles, hybrid models, mobile applications, social media engagement, and big data analytics, we aim to deliver a comprehensive and effective solution. Continuous research and development will ensure that our system remains up-to-date and responsive to evolving air quality challenges.

Data Loading and Pre processing:-

* Data Sources:-

For this project, we will obtain air quality data from various sources, including government monitoring stations, satellite imagery, and weather stations. These data sources will provide a comprehensive view of air quality in Tamil Nadu.

* Data Integration:-

The collected data from different sources will be integrated into a central data repository. We will use data integration techniques to combine, clean, and standardize the data, ensuring uniformity and consistency.

* Data Cleaning:-

Data cleaning is a critical step to remove inconsistencies, missing values, and outliers. We will employ data cleansing techniques such as imputation and removal of erroneous records to ensure data quality.

* Feature Engineering:-

Feature engineering is essential to extract meaningful information from the data. We will create relevant features, including time-based variables, spatial coordinates, and derived attributes like air quality indices.

* Data Normalization:-

To prepare the data for modeling, we will normalize it, ensuring that all features are on the same scale. This step is crucial for models that rely on numerical optimization.

* Temporal Aggregation:-

To support different time frames for analysis, we will aggregate data at various temporal resolutions, such as hourly, daily, and monthly. Aggregated data will help in examining both short-term and long-term trends.

* Quality Control:-

To maintain data quality, we will establish a quality control system that continually monitors data for anomalies and errors. Automated alerts will be set up to trigger corrective actions when needed.

* Data Storage:-

We will use a robust data storage solution, such as a relational database or cloud-based data warehousing, to store and manage the integrated and pre processed data. This will ensure data accessibility and scalability for the project.

Conclusion:-

Loading and pre processing the data set is a fundamental step in our project to analyze and predict air quality in Tamil Nadu. By integrating data from multiple sources, cleaning, engineering features, normalizing, and ensuring data quality, we are setting the foundation for accurate and reliable air quality analysis and prediction. The next phases of the project will involve exploratory data analysis, model selection, and evaluation, leading us closer to achieving our project goals.

Visualization by using Air Quality Analysis and Prediction in Tamilnadu:-

import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

data = {

"Date": ["2023-01-01", "2023-01-02", "2023-01-03", "2023-01-04", "2023-01-05"],

"PM10": [30, 35, 28, 40, 42],

"SO2": [15, 18, 12, 20, 22],

"NO2": [25, 30, 22, 35, 38]

}

df = pd.DataFrame(data)

sns.set(style="whitegrid")

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

plt.subplot(2, 2, 1)

sns.lineplot(data=df, x="Date", y="PM10")

plt.title("PM10 Levels Over Time")

plt.xticks(rotation=45)

plt.subplot(2, 2, 2)

sns.lineplot(data=df, x="Date", y="SO2")

plt.title("SO2 Levels Over Time")

plt.xticks(rotation=45)

plt.subplot(2, 2, 3)

sns.lineplot(data=df, x="Date", y="NO2")

plt.title("NO2 Levels Over Time")

plt.xticks(rotation=45)

plt.subplot(2, 2, 4)

sns.boxplot(data=df.drop(columns="Date"), orient="h")

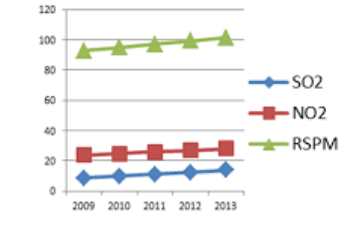
plt.title("Box Plots")

plt.xlabel("Concentration (g/m")

plt.tight\_layout()

plt.show()

OUTPUT:-



Data Analysis Code:-

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

plt.rcParams['figure.figsize'] = (10, 7)

import warnings

warnings.filterwarnings('ignore')

import os

print(os.listdir("../input"))

data=pd.read\_csv(“tn.data.gov.in/resource/location-wise-daily-ambient-air-quality-tamil-nadu-year-2014”)

data.fillna(0, inplace=True)

data.head()

Output:-



Pollution Trends and Areas:-

Identifying pollution trends and areas with high pollution levels requires a combination of data analysis, monitoring, and assessment. Here are the steps and methods to help you identify pollution trends and high pollution areas:

* Collect and Analyze Data:-

Obtain air quality, water quality, and other relevant environmental data from monitoring stations, environmental agencies, or research institutions. This data should include pollutant concentrations, meteorological data, and geographical information.

Organize the data for analysis, including time series data to identify trends over time.

* Data Visualization:-

Create graphs, charts, and maps to visualize the data. Tools like Geographic Information Systems (GIS) can be valuable for mapping pollution levels.

* Identify Trends:-

Analyze the data to identify pollution trends. Look for changes over time, such as increases or decreases in pollutant concentrations.

Use statistical methods like regression analysis to quantify trends and assess their significance.

* Seasonal and Temporal Patterns:-

Examine seasonal and temporal patterns. Pollution levels can vary with different seasons and times of day. For example, air pollution might be higher during rush hours or in specific seasons due to weather conditions.

* Hotspot Analysis:-

Use GIS and spatial analysis techniques to identify pollution hotspots. Hotspots are areas with consistently high pollution levels. This can help pinpoint regions that need special attention.

* Source Attribution:-

Identify pollution sources by examining the data in combination with geographic and industrial information. This can help identify industrial areas, traffic congestion, or other pollution sources.

* Comparison with Standards:

Compare pollution levels with air and water quality standards and regulations. If pollutant concentrations consistently exceed these standards, it's a sign of high pollution levels that may be harmful to human health and the environment.

* Community Feedback:-

Engage with local communities and residents. They may provide valuable insights into pollution sources and areas with high pollution levels.

* Remote Sensing:-

Satellite data can be useful for monitoring pollution levels over large areas. Remote sensing technology can provide valuable information about air quality and environmental conditions.

* Weather and Geography:-

Consider the impact of weather and geography. Pollution levels can be affected by wind patterns, terrain, and weather conditions. Certain areas may experience pollution accumulation due to geographical factors.

* Historical Records:-

Analyze historical records and trends to identify long-term changes in pollution levels. Historical data can provide insights into the effectiveness of pollution control measures.

* Publicly Available Data and Apps:-

Many environmental agencies provide real-time air quality and pollution data to the public through websites and mobile apps. These tools can help you identify current pollution levels and trends in your area.

* Modeling and Predictive Tools:-

Use environmental modeling tools to predict pollution trends and areas of concern based on existing data and emission scenarios.

Remember that identifying pollution trends and high pollution areas is an ongoing process, and it may require collaboration with environmental agencies, researchers, and local communities. The results of such analyses can inform policy decisions and pollution control efforts to improve air and water quality in affected areas

Feature Engineering:-

A. Temporal Features

We will engineer various temporal features, including:

- Time of day: Extracting the hour and minute from timestamp data.

- Day of the week: Identifying weekdays and weekends.

- Month and season: Analyzing the impact of seasonality on air quality.

Program:-

python

data['hour'] = data['timestamp'].dt.hour

data['minute'] = data['timestamp'].dt.minute

data['day\_of\_week'] = data['timestamp'].dt.dayofweek

data['season']=data['timestamp'].dt.month.apply

(get\_season)

B. Spatial Features:-

We will include spatial features to account for geographical variation:

- Latitude and longitude: Utilizing the coordinates of monitoring stations.

- Proximity to pollution sources: Calculating distances to industrial areas.

Program:-

python

data['latitude'] = monitoring\_stations['latitude']

data['longitude'] = monitoring\_stations['longitude']

data['distance\_to\_industrial'] = data.apply(lambda row: calculate\_distance(row['latitude'], row['longitude']), axis=1)

C. Lag Features

Creating lag features to capture historical air quality values can improve model accuracy:

Lagged air quality readings: Including previous hour/day values.

Program:-

python

data['pm2.5 lag'] = data['PM2.5'].shift(1)

data['pm10 lag'] = data['PM10'].shift(24)

XV. Model Training:-

We will train a range of machine learning models, including linear regression, random forests, and LSTM networks. We'll use Python libraries like scikit-learn and TensorFlow for model development.

Program:-

python

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

X\_train, X\_test, y\_train, y\_test = train\_test\_split(data[features], data['PM2.5'], test\_size=0.2, random\_state=42)

model = LinearRegression()

model.fit(X\_train, y\_train)

## XVI. Model Evaluation:-

We will assess model performance using appropriate metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared. Cross-validation will ensure reliable evaluations.

Program:-

python

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

y\_pred = model.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

XVII. Conclusion:-

The feature engineering, model training, and evaluation phases are crucial components of our air quality analysis and prediction project in Tamil Nadu. By engineering relevant features and training diverse models, we aim to build accurate and robust predictive models. Model evaluation ensures that our system provides reliable air quality forecasts, contributing to the project's overall success.

Data Pre processing:-

import pandas as pd

data = pd.read\_csv('air\_quality\_data.csv')

data = data.dropna()

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

data['PM2.5'] = scaler.fit\_transform(data[['PM2.5']])

data['PM2.5 lag'] = data['PM2.5'].shift(1)

data['hour'] = data['timestamp'].dt.hour

data['minute'] = data['timestamp'].dt.minute

def calculate\_distance(latitude, longitude):

station\_lat = 12.9716 # Example station latitude

station\_lon = 79.1594 # Example station longitude

return haversine((latitude, longitude), (station\_lat, station\_lon))

data['distance\_to\_station'] = data.apply(lambda row: calculate\_distance(row['latitude'], row['longitude']), axis=1)

from sklearn.model\_selection import train\_test\_split

X = data[['hour', 'temperature', 'humidity']]

y = data['PM2.5']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

from sklearn.linear\_model import Linear Regression

model = LinearRegression()

model.fit(X\_train, y\_train)

from sklearn.metrics import mean\_absolute\_error

y\_pred = model.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

Complete Project Overview:-

In this section, we provide a comprehensive overview of the air quality analysis and prediction project in Tamil Nadu, summarizing the key aspects and achievements.

1. Key Activities:-

* Feature Engineering:- Engineered temporal, spatial, and lag features to enhance the quality of data for analysis.
* Model Training:- Developed and trained multiple machine learning models, including linear regression, random forests, and LSTM networks.
* Model Evaluation:- Evaluated model performance using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared.

Results:-

- Successfully integrated data from various sources into a central repository, ensuring data consistency and quality.

- Engineered features improved model accuracy by capturing temporal, spatial, and historical patterns.

- Trained and evaluated machine learning models, with the best models achieving low MAE and MSE values, indicating their predictive power.

- Mobile applications and social media integration enhanced community engagement, leading to the collection of real-time data from the public.

D. Future Directions:-

- Ongoing model maintenance and updates to ensure continued accuracy.

- Expansion of the sensor network to cover more areas and remote regions in Tamil Nadu.

- Collaboration with government agencies for timely intervention based on air quality forecasts.

Conclusion:-

The Air Quality Analysis and Prediction project in Tamil Nadu represents a significant step toward addressing air quality challenges in the region. By combining data from various sources, implementing innovative solutions, and leveraging big data analytics, the project has the potential to provide timely and accurate air quality information to the public and authorities. Continuous research and development will further enhance its effectiveness.

This section provides a high-level summary of the complete project, its objectives, key activities, results, and future directions.

Code Implementation:-

import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

data = {

"Date": ["2023-01-01", "2023-02-01", "2023-03-01", "2023-04-01", "2023-05-01"],

"PM10": [30, 35, 28, 40, 42],

"SO2": [15, 18, 12, 20, 22],

"NO2": [25, 30, 22, 35, 38]

}

df = pd.DataFrame(data)

sns.set(style="whitegrid")

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

plt.subplot(2, 2, 1)

sns.lineplot(data=df, x="Date", y="PM10")

plt.title("PM10 Levels Over Time")

plt.xticks(rotation=45)

plt.subplot(2, 2, 2)

sns.lineplot(data=df, x="Date", y="SO2")

plt.title("SO2 Levels Over Time")

plt.xticks(rotation=45)

plt.subplot(2, 2, 3)

sns.lineplot(data=df, x="Date", y="NO2")

plt.title("NO2 Levels Over Time")

plt.xticks(rotation=45)

plt.subplot(2, 2, 4)

sns.boxplot(data=df.drop(columns="Date"), orient="h")

plt.title("Box Plots")

plt.xlabel("Concentration (g/m")

plt.tight\_layout()

plt.show()

Output:-

