Air Quality Index Prediction Using Machine Learning

Introduction: *Air Quality and Its Impact on Health:*

Air quality is an indispensable component of environmental health, intimately tied to the well-being of individuals and communities. It encompasses the condition of the air we breathe, the presence of various pollutants within it, and the overall impact of these elements on our lives. The assessment and management of air quality are paramount for safeguarding public health, informing environmental policies, and enabling individuals to make informed decisions about their daily activities.1

The quality of the air we breathe is not an abstract concern; it has tangible and far-reaching effects on our health and the environment. Pollutants in the atmosphere, such as gases and particulate matter, can have detrimental consequences when their concentrations exceed safe levels. These pollutants can originate from various sources, including industrial processes, transportation, and natural phenomena.

The Importance of Monitoring and Predicting Air Quality:

1. Health Implications: Poor air quality is linked to a range of health problems, from respiratory issues like asthma and bronchitis to cardiovascular diseases and even premature death. Vulnerable populations, such as children and the elderly, are particularly at risk.

2. Environmental Impact: Beyond human health, air pollution contributes to environmental degradation. It can harm ecosystems, damage vegetation, and lead to the acidification of soil and water bodies.

3. Policy Formulation: Accurate assessments of air quality are indispensable for policymakers. They provide the necessary data to develop effective environmental regulations and standards aimed at reducing pollution and safeguarding public health.

4. Personal Decision: Making On a daily basis, individuals make choices about outdoor activities, exercise, and travel. Access to real-time information about air quality empowers people to make informed decisions to protect their health.

Our Data-Driven Project- Predicting the Air Quality Index (AQI):

In the pursuit of healthier air and better-informed communities, our project takes a data-driven approach. We embark on a journey that leverages the power of data analysis and machine learning to predict a vital metric: the Air Quality Index (AQI).

The AQI is a comprehensive indicator that distills the complex web of air quality data into a single, easy-to-understand value. It offers insights into the overall quality of the air we breathe, considering various pollutants simultaneously. The AQI is a valuable tool for individuals, policymakers, and researchers alike.

Throughout the course of this article, we will navigate through the following key phases:

Section 0: Importing necessary libraries and Project setup:

- Pandas: Data manipulation and analysis.
- Numpy: Numerical computing with arrays.
- Seaborn/Matplotlib: Data visualization.
- Scikit-Learn (sklearn): Machine learning tools.
- XGBoost: High-accuracy machine learning algorithm.

```
#Importing necessary libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn import metrics
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
from sklearn.metrics import accuracy_score,confusion_matrix
from prettytable import PrettyTable
```

Section 1: Dataset Exploration and Visualization - Uncovering Insights Through Data Analysis:

- Data Overview: We begin by providing an overview of the dataset's size and structure.
- Data Inspection: We delve into the types of data it contains, identifying potential areas of interest and concern.
- Data Visualization: We employ visualization techniques to reveal patterns and relationships within the data, shedding light on which regions or types of locations exhibit higher pollutant levels.

The code provides insights into air quality data.

```
# Data Visualization
sns.pairplot(data=df)
```

```
df['state'].value_counts()
# viewing the count of values present in state column

plt.figure(figsize=(15,6))
plt.xticks(rotation=90)
df.state.hist()
plt.xlabel('state')
plt.ylabel('Frequencies')
plt.plot()
# The visualization shows us the count of states present in the dataset.

df['type'].value_counts()
# viewing the count of values present in type column

plt.figure(figsize=(15,6))
plt.xticks(rotation=90)
df.type.hist()
plt.xlabel('Type')
plt.ylabel('Frequencies')
plt.plot()
# The visualization shows us the count of types present in the dataset.

df['agency'].value_counts()
# viewing the count of values present in agency column

plt.figure(figsize=(15,6))
plt.xticks(rotation=90)
df.agency.hist()
plt.xlabel('Agency')
plt.ylabel('Frequencies')
plt.plot()
# The visualization shows us the count of agency present in the dataset.

plt.figure(figsize=(10,10))
plt.xticks(rotation=90)
sns.barplot(x='state',y='so2',data=df)
# This visualization shows the name of the state having higher so2 levels in the air.

df[['so2','state']].groupby(['state']).mean().sort_values(by='so2').plot.bar(color='purple')
plt.show()
# sorting states in increasing value of so2

plt.figure(figsize=(10,10))
plt.xticks(rotation=90)
sns.barplot(x='state',y='no2',data=df)
```

```
# This visualization shows the name of the state having higher no2 levels in the air.
df[['no2','state']].groupby(['state']).mean().sort_values(by='no2').plot.bar(color='purple')
plt.show()
# sorting states in increasing value of so2

plt.figure(figsize=(10,10))
plt.xticks(rotation=90)
sns.barplot(x='state',y='rspm',data=df)
# This visualization shows the name of the state having higher rspm levels in the air.

df[['rspm','state']].groupby(['state']).mean().sort_values(by='rspm').plot.bar(color='purple')
plt.show()
# sorting states in increasing value of so2

plt.figure(figsize=(10,10))
plt.xticks(rotation=90)
sns.barplot(x='state',y='spm',data=df)
# This visualization shows the name of the state having higher spm levels in the air.

plt.figure(figsize=(10,10))
plt.xticks(rotation=90)
sns.barplot(x='state',y='pm2_5',data=df)
```

Section 2: Data Preprocessing - Preparing the Data for Model Building:

Handling Missing Values: We address missing data to ensure that the dataset is ready for model development.

```
#Reading the dataset
df=pd.read_csv('data.csv',encoding='unicode_escape')

# Dropping unnecessary columns
df.drop(['agency'],axis=1,inplace=True)
df.drop(['stn_code'],axis=1,inplace=True)
df.drop(['date'],axis=1,inplace=True)
df.drop(['sampling_date'],axis=1,inplace=True)
df.drop(['location_monitoring_station'],axis=1,inplace=True)

# null value imputation for catagorical data
df['location']=df['location'].fillna(df['location'].mode()[0])
df['type']=df['type'].fillna(df['type'].mode()[0])

# null values are replaced with zeros for the numerical data
df.fillna(0, inplace=True)
```

Section 3: Calculating Individual Pollutant Indices - Unveiling the Impact of Key Pollutants:

For key pollutants like Sulphur Dioxide (SO2), Nitrogen Dioxide (NO2), Respirable Suspended Particulate Matter (RSPM), and Suspended Particulate Matter (SPM), we calculate individual pollutant indices. These indices help us understand the individual contributions of pollutants to air quality.

```
# calculating the individual pollutant index for so2(Sulphur dioxide)
def cal_SOi(so2):
    si=0
    if (so2<=40):
        si=so2*(50/40)
    elif(so2>40 and so2<=80):
        si=50+(so2-40)*(50/40)
    elif(so2>80 and so2<=380):
        si=100+(so2-80)*(100/300)
    elif(so2>380 and so2<=800):
        si=200+(so2-380)*(100/420)
    elif(so2>800 and so2<1600):
        si=300+(so2-800)*(100/800)
    elif(so2>1600):
        si=400+(so2-1600)*(100/800)
    return si
df['SOi']=df['so2'].apply(cal_SOi)
data= df[['so2','SOi']]

# calculating the individual pollutant index for no2(Nitrogen dioxide)
def cal_NOi(no2):
    ni=0
    if (no2<=40):
        ni=no2*(50/40)
    elif(no2>40 and no2<=80):
        ni=50+(no2-40)*(50/40)
    elif(no2>80 and no2<=380):
        ni=100+(no2-80)*(100/300)
    elif(no2>380 and no2<=800):
        ni=200+(no2-380)*(100/420)
    elif(no2>800 and no2<1600):
        ni=300+(no2-800)*(100/800)
    elif(no2>1600):
        ni=400+(no2-1600)*(100/800)
    return ni
df['NOi']=df['no2'].apply(cal_NOi)
data= df[['no2','NOi']]

# calculating the individual pollutant index for rspm(respirable suspended particulate matter)
def cal_RSPMI(rspm):
    rpi=0
    if (rpi<=30):
        rpi=rpi*50/30
```

```
    elif (rpi>30 and rpi<=60):
        rpi=50+(rpi-30)*50/30
    elif (rpi>60 and rpi<=90):
        rpi=100+(rpi-60)*100/30
    elif (rpi>90 and rpi<=120):
        rpi=200+(rpi-90)*100/30
    elif (rpi>120 and rpi<=250):
        rpi=300+(rpi-120)*(100/130)
    else:
        rpi=400+(rpi-250)*(100/130)
    return rpi
df['Rpi']=df['rspm'].apply(cal_RSPMI)
data= df[['rspm','Rpi']]

# calculating the individual pollutant index for spm(suspended particulate matter)
def cal_SPMI(spm):
    spi=0
    if (spm<=50):
        spi=spm*50/50
    elif (spm>50 and spm<=100):
        spi=50+(spm-50)*(50/50)
    elif (spm>100 and spm<=250):
        spi=100+(spm-100)*(100/150)
    elif (spm>250 and spm<=350):
        spi=200+(spm-250)*(100/100)
    elif (spm>350 and spm<=430):
        spi=300+(spm-350)*(100/80)
    else:
        spi=400+(spm-430)*(100/430)
    return spi
df['SPMI']=df['spm'].apply(cal_SPMI)
data= df[['spm','SPMI']]
```

Section 4: Calculating the Air Quality Index (AQI) - The Comprehensive Measure of Air Quality:

We introduce the concept of the AQI and develop a function to calculate it based on the individual pollutant indices. This enables us to assess air quality holistically.

```
#Calculating the air quality index
def cal_aqi(si,ni,rpi,spi):
    aqi=0
    if(si>ni and si>rpi and si>spi):
        aqi=si
    if(ni>si and ni>rpi and ni>spi):
        aqi=ni
    if(rpi>si and rpi>ni and rpi>spi):
        aqi=rpi
```

```
    if(spi>si and spi>ni and spi>rpi):
        aqi=spi
    return aqi

df['AQI']=df.apply(lambda x:cal_aqi(x['SOi'],x['NOi'],x['Rpi'],x['SPMI']),axis=1)
data= df[['state','SOi','NOi','Rpi','SPMI','AQI']]
```

Section 5: Categorizing AQI - Making AQI Understandable:

To enhance interpretability, we categorize the AQI into different ranges, making it more accessible for individuals and policymakers to gauge air quality conditions at a glance.

Using threshold values to classify a particular values as good, moderate, poor, unhealthy, very unhealthy and hazardous

```
# using threshold values to classify a particular values as good, moderate, poor, unhealthy,
very unhealthy and hazardous
def AQI_Range(x):
    if x<=50:
        return "Good"
    elif x>50 and x<=100:
        return "Moderate"
    elif x>100 and x<=200:
        return "Poor"
    elif x>200 and x<=300:
        return "Unhealthy"
    elif x>300 and x<=400:
        return "Very Unhealthy"
    elif x>400:
        return "Hazardous"

df['AQI_Range']=df['AQI'] .apply(AQI_Range)
df.head()

# we only select columns like soi, noi, rspi, spi
X=df[['SOi', 'NOi', 'Rpi', 'SPMI']]
Y=df['AQI']
X.head()
```

Section 6: Machine Learning Model Training - Harnessing the Power of Regression Models:

We delve into the world of machine learning, selecting regression models like Linear Regression, Decision Tree Regressor, and Random Forest Regressor to predict AQI. We discuss data splitting, model training, and evaluation using metrics like RMSE and R-squared.

```
# Splitting the data into training and testing data
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=70)
```

```
RF=RandomForestRegressor().fit(X_train,Y_train)

#predicting train
train_preds1=RF.predict(X_train)
#predicting test
test_preds1=RF.predict(X_test)

RMSE_train=(np.sqrt(metrics.mean_squared_error(Y_train,train_preds1)))
RMSE_test=(np.sqrt(metrics.mean_squared_error(Y_test,test_preds1)))

print("RMSE TrainingData = ",str(RMSE_train))
print("RMSE TestData = ",str(RMSE_test))
print('-'*50)
print('RSquared value on train:',RF.score(X_train,Y_train))
print('RSquared value on test:',RF.score(X_test,Y_test))
```

Section 7: Making Predictions - Empowering Real-Time Air Quality Monitoring:

With trained models, we can make real-time predictions of AQI by inputting individual pollutant indices. This capability is invaluable for monitoring and forecasting air quality in specific locations.

```
# Using a regression model to predict AQI value
predicted_aqi = RF.predict([[45,23.23,57,106]])
print("Predicted AQI:", predicted_aqi[0])

print(AQI_Range(predicted_aqi))
```

Section 8: Saving the Model - Ensuring Future Accessibility

To preserve our insights, we emphasize the importance of saving trained models to files. This facilitates their reuse without the need for retraining, ensuring that our data-driven approach remains accessible and relevant.

```
import joblib
joblib.dump(RF,'RF_regression.pkl')
```

References:

GitHub Repository:- Link to GitHub Repository where the project's source code and documentation are hosted.

Link to Repository: https://github.com/Tejas-warade/Air-Quality-Prediction-India/tree/main

Dataset:- In this Project "india-air-quality-data" dataset is used from kaggle.com.

Link to DataSet: https://www.kaggle.com/datasets/shrutibhargava94/india-air-quality-data

Installation and Setup:- To set up and run your project locally. Clone the github repo ,install all the dependencies and python libraries, and run app.py file and go localhost 127.0.0.1 url or the provided url.

Tutorial to setup: https://gregarious-maamoul-2110f2.netlify.app/

Contributing: If you want others to contribute to project submit issues, feature requests, or pull requests.

Note: Remember to keep your project repository well-maintained and up-to-date. Regularly review and respond to issues and pull requests from contributors to foster a collaborative community around your project.

Conclusion:

In conclusion, this article aims to illuminate the pivotal role of data analysis and machine learning in understanding, monitoring, and predicting air quality. It underscores the significance of informed decision-making in environmental health and policy formulation. As we embark on this journey through the realms of data-driven environmental science, we aim to contribute to a future where the air we breathe is cleaner, safer, and healthier for all.