**Air quality analysis and prediction in tamilnadu coding**

**import pandas as pd**

**import numpy as np**

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

**from sklearn.ensemble import RandomForestRegressor**

**from sklearn.metrics import mean\_squared\_error**

**import matplotlib.pyplot as plt**

**# Load your air quality data (assuming you have a CSV file with columns: 'Date', 'PM2.5', and possibly other features)**

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

**# Preprocess the data (you may need to handle missing values and perform feature engineering)**

**# For simplicity, let's assume you have already cleaned and preprocessed the data**

**# Extract features and target variable**

**X = data.drop(['Date', 'PM2.5'], axis=1) # Features (excluding Date and PM2.5)**

**y = data['PM2.5'] # Target variable (PM2.5)**

**# Split the data into training and testing sets**

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

**# Train a Random Forest regressor (you can try other regression models as well)**

**model = RandomForestRegressor(n\_estimators=100, random\_state=42)**

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

**# Make predictions**

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

**# Evaluate the model (use appropriate evaluation metrics for regression)**

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

**print(f'Mean Squared Error (MSE): {mse}')**

**# Visualize the actual vs. predicted values**

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

**plt.plot(y\_test.index, y\_test.values, label='Actual PM2.5')**

**plt.plot(y\_test.index, y\_pred, label='Predicted PM2.5', linestyle='--', marker='o', markersize=4)**

**plt.xlabel('Date')**

**plt.ylabel('PM2.5 Concentration')**

**plt.legend()**

**plt.title('Air Quality Prediction in Tamil Nadu')**

**plt.grid(True)**

**plt.show()**

* Load your air quality data from a CSV file, assuming it contains columns 'Date' and 'PM2.5' (you can add other relevant features).
* Preprocess your data as needed, including handling missing values and feature engineering.
* Split the data into training and testing sets. The **train\_test\_split** function is used for this purpose.
* Train a Random Forest regressor to predict PM2.5 concentrations. You can experiment with other regression models.
* Evaluate the model using the Mean Squared Error (MSE), which is a common metric for regression problems.
* Visualize the actual vs. predicted PM2.5 values.