

Anna University Regional Campus Coimbatore

Anna University: Chennai-600 025

DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING



IBM Naan Mudhalvan Phase 4 Submission

Development Part-II

**Title: AIR QUALITY ANALYSIS AND
PREDICTION IN TAMILNADU**

Name : PRAVEEN S

Register Number: 710021106027

Department :B.E.ECE

Sem/year :V/III

Air Quality Analysis and Prediction in Tamil Nadu

Objective:

The Objective of the project is to analyze and visualize the air quality data from the various monitoring stations in Tamil Nadu. The given dataset contains the measurements of the various gases that release into the atmosphere. Some of the gases that are given in the dataset are Sulphur Dioxide(SO₂), Nitrogen Dioxide(NO₂) and Respirable particulate matter and these are measured in different cities, villages, towns. This Project aim is to gain the insight of air pollution trends, estimate the RSPM/PM₁₀ levels based on SO₂ and NO₂ levels

Description of dataset:

The link for the chosen dataset is mentioned below,

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

The above dataset contains the combined version of air quality of Tamil Nadu. This contains the district wise dataset for the prediction of air quality parameter in the state of Tamil Nadu. This data was released by the Ministry of Environment and Forests and Central Pollution Control Board of India under the National Data Sharing and Accessibility Policy

OVERVIEW OF THE PROCESS

The following steps are the overview of the process of building an air quality analysis and prediction model.

STEP1: Choosing a dataset

Choosing the proper dataset for implementing the project

STEP2: Data Pre-processing

In data pre-processing we have selected data for the analysis of air quality in the various district of Tamil Nadu. Each of the dataset was cleaned by removing null values of the chosen dataset. Microsoft Excel Software was used to remove unnecessary, irrelevant and erroneous data.

STEP3: Splitting of the dataset

The chosen datasets are split into training and test data. These are used to train the model and then test it against the original data. The values predicted by the machine learning algorithm and to predict accuracy of the data.

STEP4: Training the dataset

Empirical studies show the best results which are obtained if 80% of the data is used for training. Random sampling is used as a way to divide the data into train and test sections. It is widely accepted and is very popular.

STEP5: Testing the dataset

Empirical studies show the best results that are obtained if the remaining 20% of the data is used for testing. Random sampling is used as a way to divide the data into train and test sections. It is widely accepted and is very popular.

STEP6: Feature Scaling

The data should be normalized in order to make the dataset more flexible and more consistent. Standard Scalar from Scikit-Learn Library has been used to do so. It normalizes the features by deleting the mean and scaling the unit variance

STEP7: Applying various Machine learning techniques

After the normalization, we need to apply the various machine learning technique for analysing the data. Some of the machine learning technique random forest regression, support vector regression which are used to analysis the air quality index.

STEP8: Applying ML technique-random forest regression

Random forest is a supervised machine learning algorithm that is used for classification and regression problems. It creates decision trees from several samples, using the majority vote for classification and the average in the case of regression. A random forest produces precise predictions that are easy to understand. Effective handling of large datasets is possible.

STEP 9: Calculation of evaluation metric for each ML techniques

The metrics used for the proposed work are R-SQUARE, MSE, RMSE, MAE, and the accuracy of various algorithm.

STEP 10: Determine the efficient Visualization techniques

Visualizations play a crucial role in conveying insights from air quality data analysis. Here are some visualization methods and techniques that can be employed in the “**Air Quality Analysis and Prediction in Tamil Nadu**”

CODE:

Loading the pre-processed dataset:

```
import pandas as pd
df=pd.read_csv('preprocessed_airquality.csv')
```

One Hot Encoding:

For categorical variables where no such ordinal relationship exists, the integer encoding is not enough.

In fact, using the one hot encoding technique and allowing the model to assume a natural ordering between categories may result in poor performance or unexpected results. This can be applied to the integer representation.

```
pd.get_dummies(df,columns=['City/Town/Village/Area','Type of
Location'])

dist=(df['City/Town/Village/Area'])
distset=set(dist)
dd=list(distset)
dict0fwords={dd[i]:i for i in range(0,len(dd))}
df['City/Town/Village/Area']=df['City/Town/Village/Area'].map(dict0fwords)
```

```
dist=(df['Type of Location'])
distset=set(dist)
dd=list(distset)
dict0fwords={dd[i]:i for i in range(0,len(dd))}
```

```
df['Type of Location']=df['Type of Location'].map(dict0fwords)
```

```
[ ] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2879 entries, 0 to 2878
Data columns (total 5 columns):
#   Column              Non-Null Count  Dtype
---  -
0   City/Town/Village/Area 2879 non-null  int64
1   Type of Location      2879 non-null  int64
2   SO2                   2879 non-null  float64
3   NO2                   2879 non-null  float64
4   RSPM/PM10             2879 non-null  float64
dtypes: float64(3), int64(2)
memory usage: 112.6 KB
```

Model Training:

- Choose a machine learning algorithm
- There are number of different machine learning algorithms that can be used for air quality analysis such as linear regression, KNN, Lasso Regression and Random Forests.

Model Evaluation:

- Model Evaluation is the process of assessing the performance of a machine learning model on unseen data. This is important to ensure that the model will generalize well to the new data.
- There are number of different metrics that can be used to evaluate the performance of air quality analysis and prediction model
- Some of the most common metrics include:
 - **Mean Squared Error (MSE):** This metric measures the average squared difference between the different areas/cities in Tamil Nadu and the various pollutants such as SO2, NO2 and RSPM/PM10.
 - **Root Mean Squared Error (RMSE):** This metric is the square root of the MSE
 - **Mean Absolute Error (MAE):** This metric measures the average absolute difference between the different areas/cities in Tamil Nadu and the various pollutants such as SO2, NO2 and RSPM/PM10.
 - **R-Squared:** This metric measures how well the model explains the variation in the pollutants in the different areas.

Linear Regression:

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

```
X=df[['City/Town/Village/Area']].values
y=df[['SO2','NO2','RSPM/PM10']].values
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,
shuffle=False)
```

```
lr_model=LinearRegression()
lr_model.fit(X_train,y_train)
```

Connecting to a runtime to enable file browsing.

{x}



```
[ ] X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,shuffle=False)
```

```
lr_model=LinearRegression()
lr_model.fit(X_train,y_train)
```

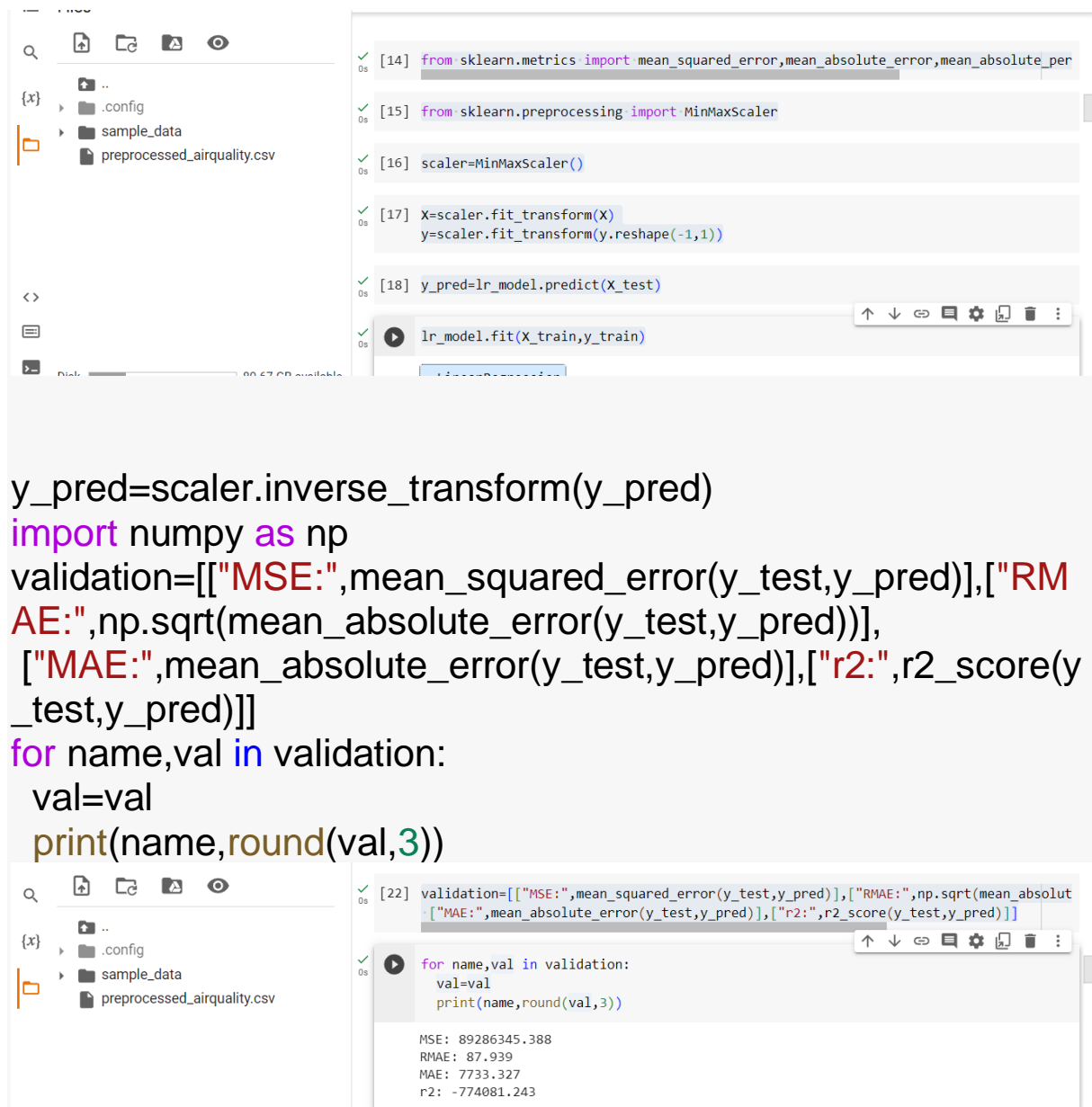
LinearRegression
LinearRegression()

```
from sklearn.metrics import
mean_squared_error,mean_absolute_error,mean_absolute_percentage_error,r2_score
```

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
```

```
X=scaler.fit_transform(X)
y=scaler.fit_transform(y.reshape(-1,1))
```

```
y_pred=lr_model.predict(X_test)
lr_model.fit(X_train,y_train)
```



```
[14] from sklearn.metrics import mean_squared_error, mean_absolute_error, mean_absolute_per
[15] from sklearn.preprocessing import MinMaxScaler
[16] scaler=MinMaxScaler()
[17] X=scaler.fit_transform(X)
    y=scaler.fit_transform(y.reshape(-1,1))
[18] y_pred=lr_model.predict(X_test)

lr_model.fit(X_train,y_train)

y_pred=scaler.inverse_transform(y_pred)
import numpy as np
validation=[["MSE:",mean_squared_error(y_test,y_pred)],["RM
AE:",np.sqrt(mean_absolute_error(y_test,y_pred))],
["MAE:",mean_absolute_error(y_test,y_pred)],["r2:",r2_score(y
_test,y_pred)]]
for name,val in validation:
    val=val
    print(name,round(val,3))

[22] validation=[["MSE:",mean_squared_error(y_test,y_pred)],["RMAE:",np.sqrt(mean_absolut
["MAE:",mean_absolute_error(y_test,y_pred)],["r2:",r2_score(y_test,y_pred)]]

for name,val in validation:
    val=val
    print(name,round(val,3))

MSE: 89286345.388
RMAE: 87.939
MAE: 7733.327
r2: -774081.243
```

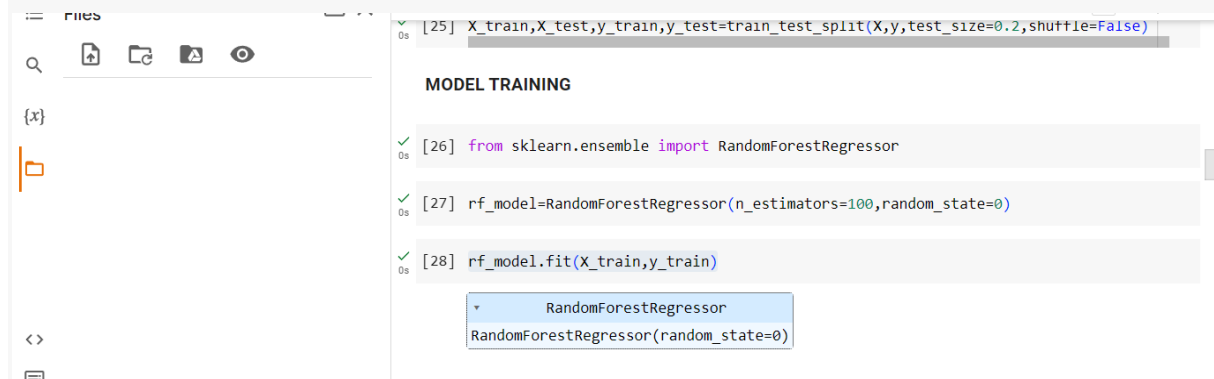
RANDOM FORESTS:

```
X=df[['City/Town/Village/Area']].values
y=df[['SO2','NO2','RSPM/PM10']].values
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,
shuffle=False)
```

Model Training

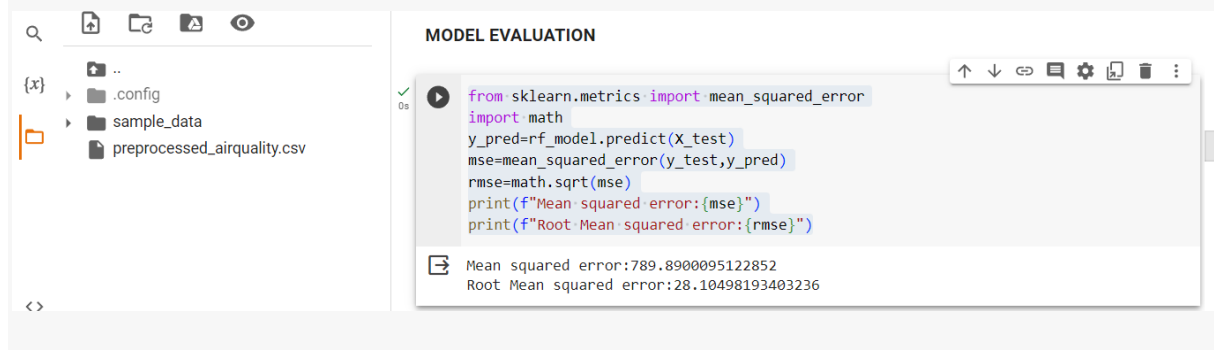
```
from sklearn.ensemble import RandomForestRegressor
rf_model=RandomForestRegressor(n_estimators=100,random
_state=0)
```

```
rf_model.fit(X_train,y_train)
```



Model Evaluation

```
from sklearn.metrics import mean_squared_error
import math
y_pred=rf_model.predict(X_test)
mse=mean_squared_error(y_test,y_pred)
rmse=math.sqrt(mse)
print(f"Mean squared error:{mse}")
print(f"Root Mean squared error:{rmse}")
```

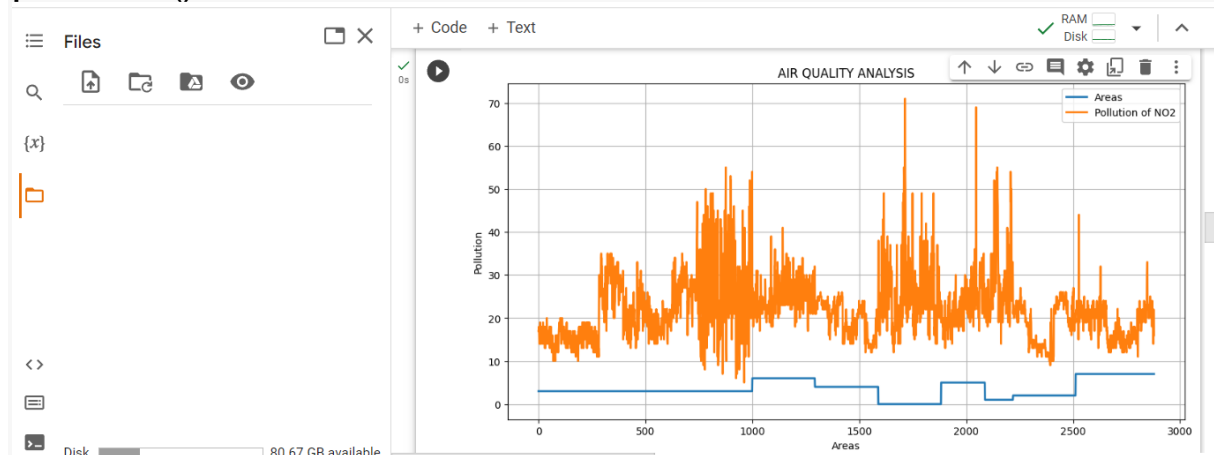


Visualization of Random Forest

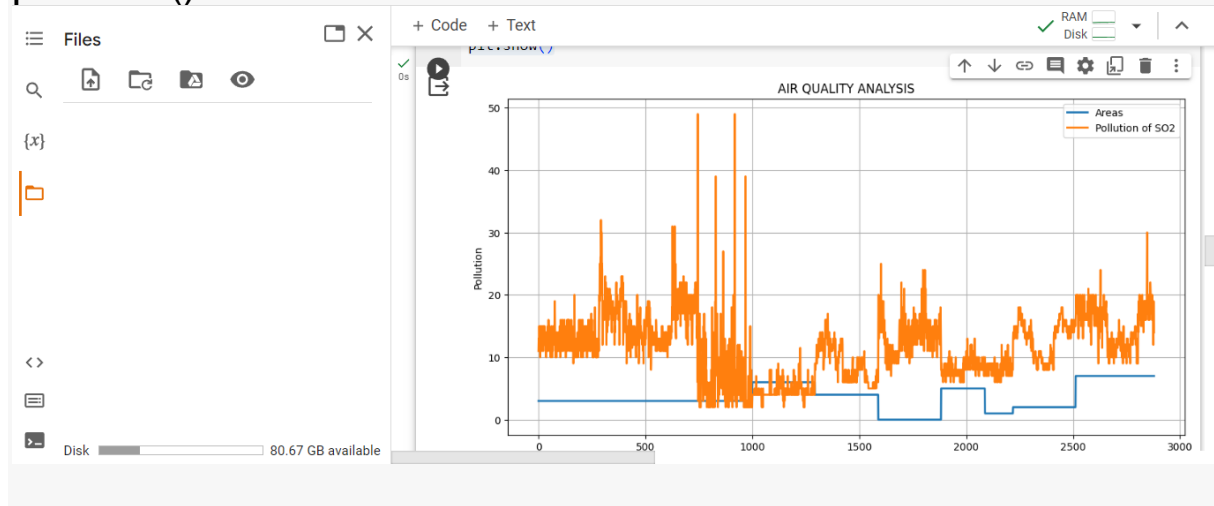
```
import matplotlib.pyplot as plt
data_range=df.index[-len(y_test):]
plt.figure(figsize=(12,6))
plt.plot(df['City/Town/Village/Area'],label='Areas',linewidth=2)
plt.plot(df['NO2'],label='Pollution of NO2',linewidth=2)
plt.title("AIR QUALITY ANALYSIS")
plt.xlabel('Areas')
plt.ylabel('Pollution')
plt.legend()
```



```
plt.grid()
plt.show()
```



```
import matplotlib.pyplot as plt
data_range=df.index[-len(y_test):]
plt.figure(figsize=(12,6))
plt.plot(df['City/Town/Village/Area'],label='Areas',linewidth=2)
plt.plot(df['SO2'],label='Pollution of SO2',linewidth=2)
plt.title("AIR QUALITY ANALYSIS")
plt.xlabel('Areas')
plt.ylabel('Pollution')
plt.legend()
plt.grid()
plt.show()
```

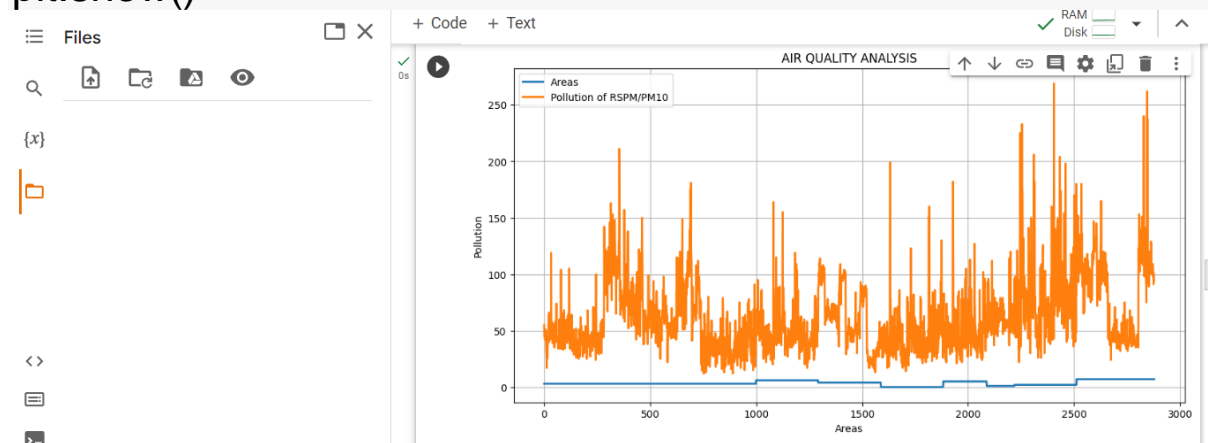


```
import matplotlib.pyplot as plt
data_range=df.index[-len(y_test):]
```

```

plt.figure(figsize=(12,6))
plt.plot(df['City/Town/Village/Area'],label='Areas',linewidth=2)
plt.plot(df['RSPM/PM10'],label='Pollution of
RSPM/PM10',linewidth=2)
plt.title("AIR QUALITY ANALYSIS")
plt.xlabel('Areas')
plt.ylabel('Pollution')
plt.legend()
plt.grid()
plt.show()

```



KNN:

```

X=df[['City/Town/Village/Area']].values
y=df[['SO2','NO2','RSPM/PM10']].values
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,
shuffle=False)
from sklearn.neighbors import KNeighborsRegressor
knn_model=KNeighborsRegressor(n_neighbors=5)
knn_model.fit(X_train,y_train)

```

```

[36] X=df[['City/Town/Village/Area']].values
     y=df[['SO2','NO2','RSPM/PM10']].values
     X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,shuffle=False)

[37] from sklearn.neighbors import KNeighborsRegressor

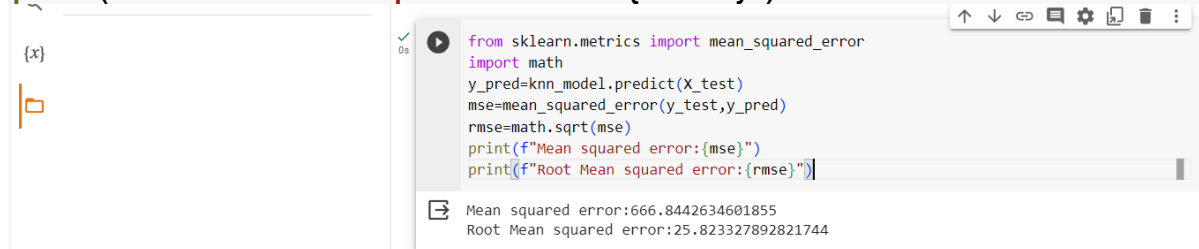
[38] knn_model=KNeighborsRegressor(n_neighbors=5)
     knn_model.fit(X_train,y_train)

```

```

from sklearn.metrics import mean_squared_error
import math
y_pred=knn_model.predict(X_test)
mse=mean_squared_error(y_test,y_pred)
rmse=math.sqrt(mse)
print(f"Mean squared error:{mse}")
print(f"Root Mean squared error:{rmse}")

```



```

from sklearn.metrics import mean_squared_error
import math
y_pred=knn_model.predict(X_test)
mse=mean_squared_error(y_test,y_pred)
rmse=math.sqrt(mse)
print(f"Mean squared error:{mse}")
print(f"Root Mean squared error:{rmse}")

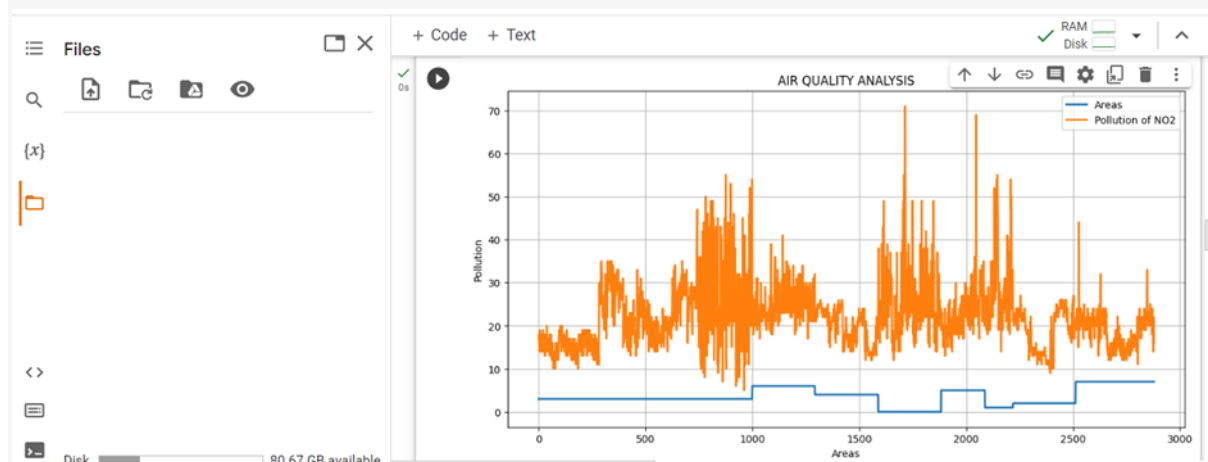
```

Mean squared error:666.8442634601855
Root Mean squared error:25.823327892821744

```

import matplotlib.pyplot as plt
data_range=df.index[-len(y_test):]
plt.figure(figsize=(12,6))
plt.plot(df['City/Town/Village/Area'],label='Areas',linewidth=2)
plt.plot(df['NO2'],label='Pollution of NO2',linewidth=2)
plt.title("AIR QUALITY ANALYSIS")
plt.xlabel('Areas')
plt.ylabel('Pollution')
plt.legend()
plt.grid()
plt.show()

```



```

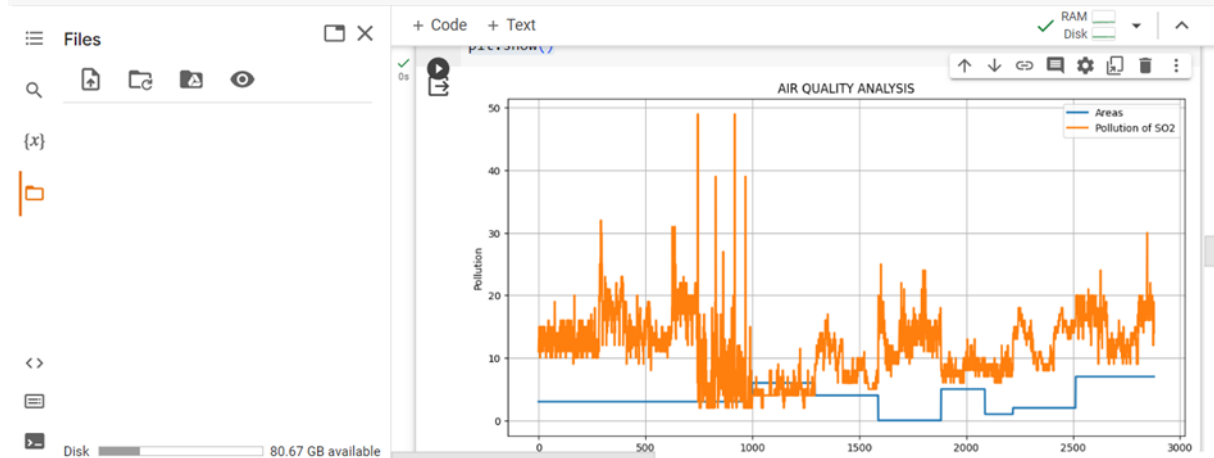
import matplotlib.pyplot as plt
data_range=df.index[-len(y_test):]
plt.figure(figsize=(12,6))

```

```

plt.plot(df['City/Town/Village/Area'],label='Areas',linewidth=2)
plt.plot(df['SO2'],label='Pollution of SO2',linewidth=2)
plt.title("AIR QUALITY ANALYSIS")
plt.xlabel('Areas')
plt.ylabel('Pollution')
plt.legend()
plt.grid()
plt.show()

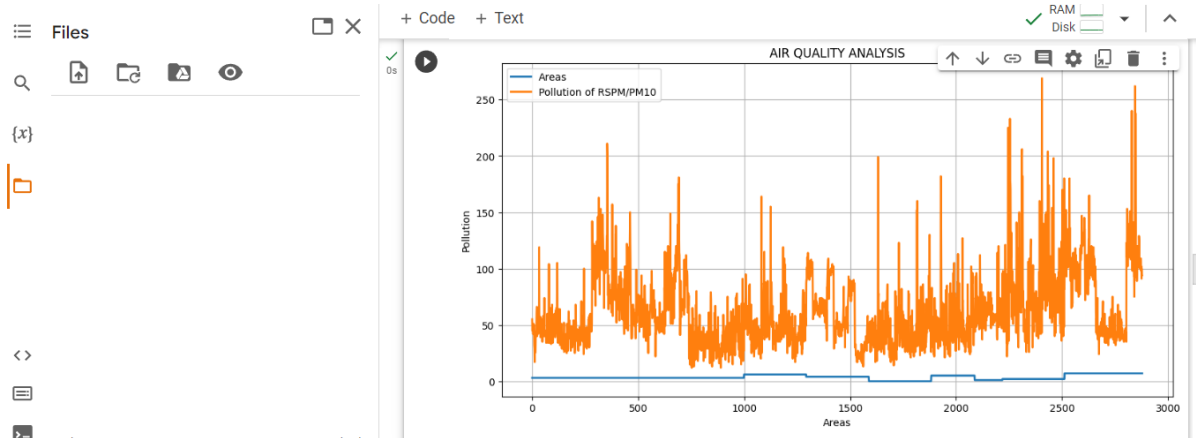
```



```

import matplotlib.pyplot as plt
data_range=df.index[-len(y_test):]
plt.figure(figsize=(12,6))
plt.plot(df['City/Town/Village/Area'],label='Areas',linewidth=2)
plt.plot(df['RSPM/PM10'],label='Pollution of
RSPM/PM10',linewidth=2)
plt.title("AIR QUALITY ANALYSIS")
plt.xlabel('Areas')
plt.ylabel('Pollution')
plt.legend()
plt.grid()
plt.show()

```



Import Libraries:

```
from sklearn.linear_model import Lasso,SGDRegressor,Ridge
from sklearn.svm import SVR
from sklearn.gaussian_process import
GaussianProcessRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split
import seaborn as sns
```

```
[43] from sklearn.linear_model import Lasso,SGDRegressor,Ridge
      from sklearn.svm import SVR
      from sklearn.gaussian_process import GaussianProcessRegressor
      from sklearn.tree import DecisionTreeRegressor
      from sklearn.model_selection import train_test_split
      import seaborn as sns

[44] X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,shuffle=False)
```

Model Training:

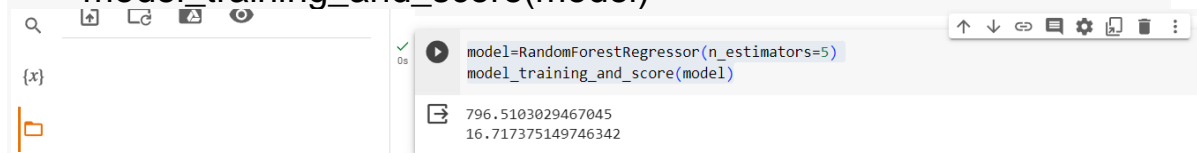
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,
shuffle=False)
ms=[]
ma=[]
mse=mean_squared_error
mae=mean_absolute_error
def model_training_and_score(model):
    model.fit(X_train,y_train)
    y_pred=np.nan_to_num(model.predict(X_test))
    print(mse(y_test,y_pred))
```

```
print(mae(y_test,y_pred))
ms.append(mse(y_test,y_pred))
ma.append(mae(y_test,y_pred))
```

Model Evaluation:

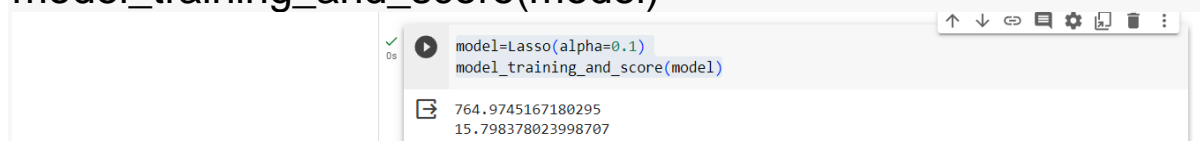
1. Random Forest

```
model=RandomForestRegressor(n_estimators=5)
model_training_and_score(model)
```



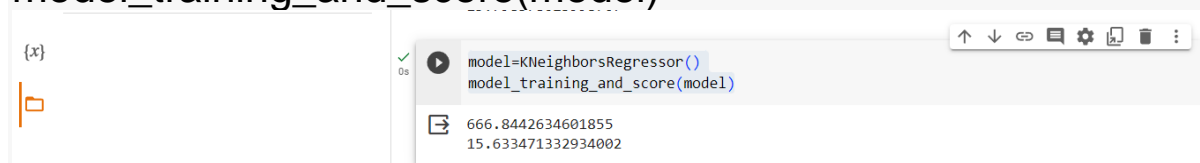
2.Lasso Regrssion

```
model=Lasso(alpha=0.1)
model_training_and_score(model)
```



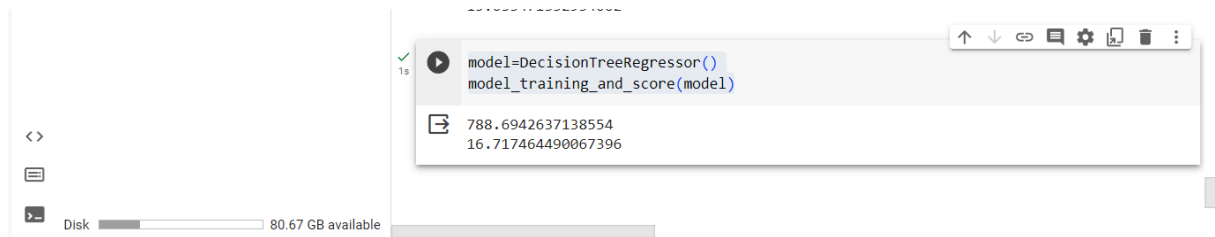
3.K Neighbors Regression

```
model=KNeighborsRegressor()
model_training_and_score(model)
```



4. Decision Tree Regression

```
model=DecisionTreeRegressor()
model_training_and_score(model)
```



```
model=DecisionTreeRegressor()  
model_training_and_score(model)
```

```
788.6942637138554  
16.717464490067396
```

CONCLUSION:

In conclusion, while data science techniques can provide valuable insights into **Air Quality Analysis and Prediction in Tamil Nadu**. Moreover, it determines the various pollutants that affect the different cities in Tamil Nadu and it will be used to prediction of air quality across the various regions of Tamil Nadu to control the air pollution across the state.