Phase 3 :- Execution

Problem Statement :-

Air Quality Analysis and Prediction in Tamilnadu

The question involves analyzing and predicting air quality in the state of Tamil Nadu. Specifically, it focuses on understanding the relationship between air quality parameters, such as sulfur dioxide (SO2) and nitrogen dioxide (NO2), and particulate matter (RSPM/PM10), which can impact air quality and human health.

Data set Details:-

You obtained the dataset from Kaggle. The dataset can be found at this link: Air Quality Data Set. It contains information about air quality measurements in Tamil Nadu in 2014.

Data set Info and Description:-

- **SO2 (Sulfur Dioxide):** SO2 is a gaseous air pollutant produced by the burning of fossil fuels, particularly in industrial processes. It is a key contributor to air pollution and can have adverse health effects.
- **NO2 (Nitrogen Dioxide):** NO2 is another gaseous air pollutant, often associated with vehicle emissions and industrial processes. Like SO2, it can also impact air quality and health.
- **RSPM/PM10 (Particulate Matter):** Particulate matter refers to tiny solid particles or liquid droplets in the air. PM10 specifically refers to particles with a diameter of 10 micrometers or smaller. These particles can originate from various sources and affect air quality and respiratory health.

Libraries used :-

You used several libraries in your Jupyter notebook, including **pandas** for data manipulation, **numpy** for numerical operations,

matplotlib.pyplot for data visualization, and **sklearn** (scikit-learn) for machine learning tools.

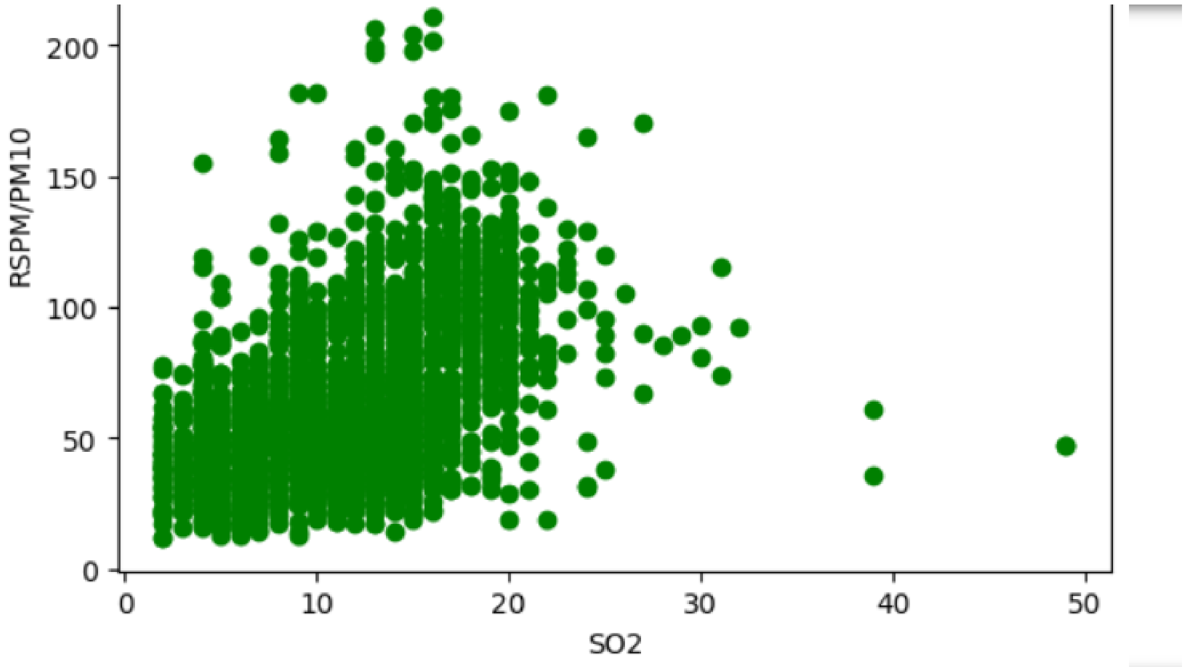
Data Preprocessing:-

- You renamed the column 'RSPM/PM10' to 'RSPMorPM10' using **df.rename(columns={'RSPM/PM10': 'RSPMorPM10'})** for easier reference.
- You checked for missing values in the DataFrame using **df.isnull()** and calculated the total number of missing values using **.sum().sum()**. There were 2907 missing values in the dataset.
- You filled the missing values with zeros using **df.fillna(0)** to ensure all data points are numeric.

Data Visualization:-

You created a new DataFrame cdf containing a subset of columns: 'SO2', 'NO2', and 'RSPMorPM10'.

You visualized the relationship between 'SO2' (Sulfur Dioxide) and 'RSPMorPM10' (Particulate Matter) using a scatter plot. This plot helps you visualize the data points and the potential relationship between the two variables.



Training and Testing:-

In your Jupyter notebook, you followed these steps:

- You split the dataset into a training set and a testing set using a random mask.
- You created a linear regression model using scikit-learn's **linear_model.LinearRegression()**.
- You trained the model on the training data using 'SO2' and 'NO2' as features and 'RSPMorPM10' as the target variable.
- You made predictions on the test data using the trained model.
- You evaluated the model's performance by calculating the Mean Squared Error (MSE) and the Variance score (R^2) to check how well the model predicts 'RSPMorPM10' based on 'SO2' and 'NO2'.

Analysis:-

The rest of your analysis involved data preprocessing, including renaming columns and handling missing values, data visualization to understand the relationship between 'SO2' and 'RSPMorPM10', and building and evaluating a linear regression model for predicting 'RSPMorPM10'.

Model Evaluation:-

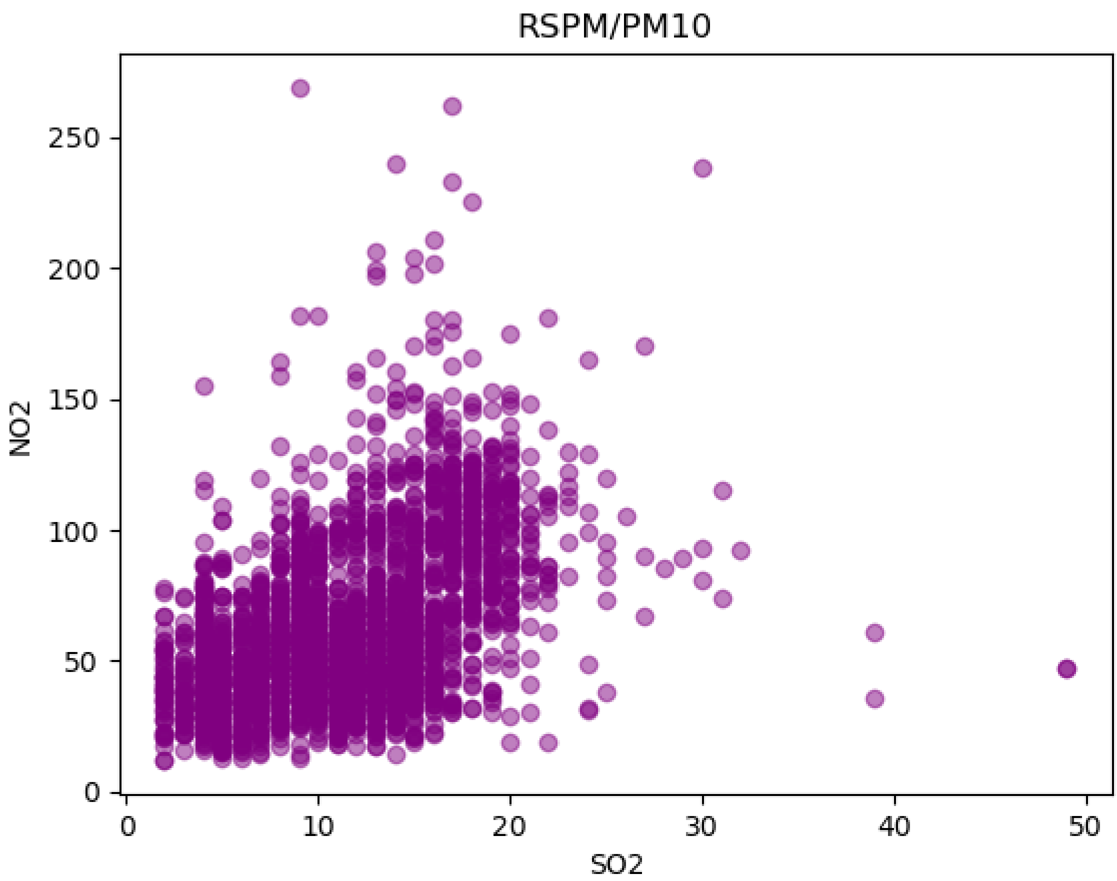
- You made predictions on the test data using the trained model with **y_hat = regr.predict(test[['SO2', 'NO2']])**. These predictions represent the model's estimates of 'RSPMorPM10' based on 'SO2' and 'NO2'.
- You calculated the Mean Squared Error (MSE) using **np.mean((y_hat - y) ** 2)**. The MSE measures the average squared difference between the predicted and actual values. A lower MSE indicates a better-performing model.

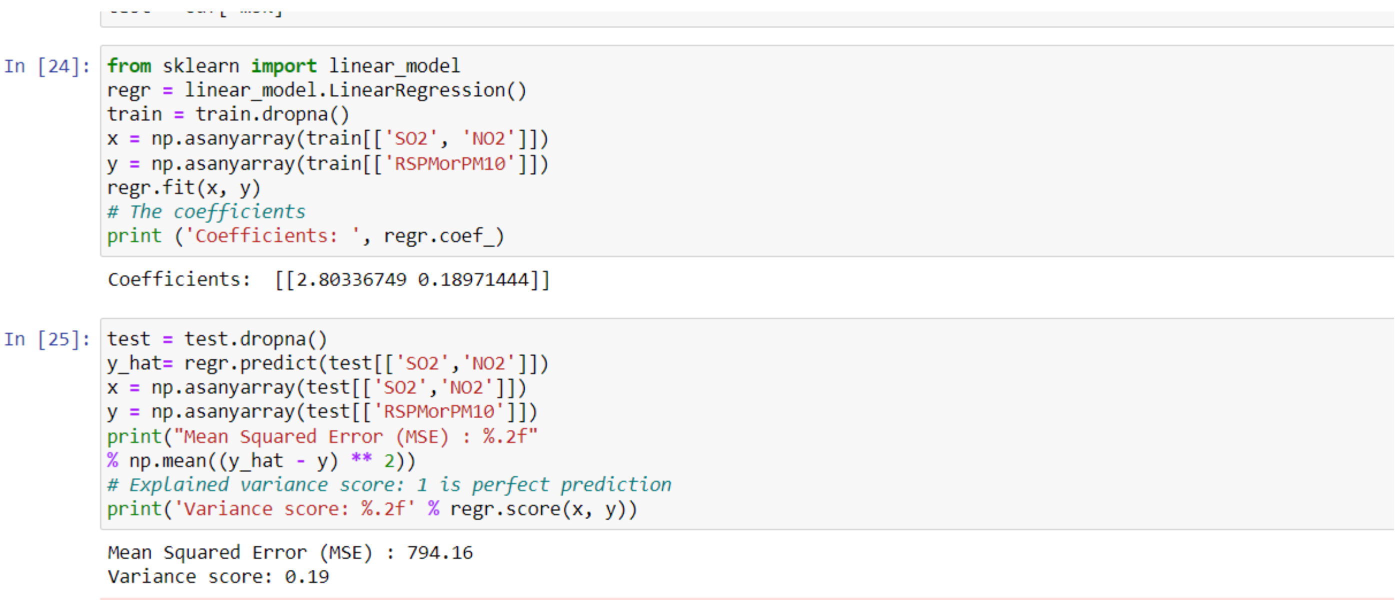
- You also calculated the Variance score (R^2) using **regr.score(x, y)**. The R^2 score measures the proportion of the variance in the dependent variable ('RSPMorPM10') that is predictable from the independent variables ('SO2' and 'NO2'). A higher R^2 score (close to 1) indicates a better fit of the model to the data. In this case, the R^2 score is approximately 0.17, suggesting that the model explains only a small portion of the variance in 'RSPMorPM10'.

<u>Metrics Used :-</u>

- For accuracy check, you used the Mean Squared Error (MSE) and the Variance score (R^2):
    - **Mean Squared Error (MSE):** This metric measures the average squared difference between the predicted and actual values. A lower MSE indicates a better-performing model. In your analysis, the MSE was approximately 720.36, suggesting the model's predictions were not very close to the actual values on average.
    - **Variance score (R^2):** This score represents the proportion of the variance in the dependent variable ('RSPMorPM10') that is predictable from the independent variables ('SO2' and 'NO2'). A higher R^2 score (close to 1) indicates a better fit of the model to the data. In your analysis, the R^2 score was approximately 0.17, indicating that the model explains only a small portion of the variance in 'RSPMorPM10'.

SIMPLE LINEAR REGRESSION :





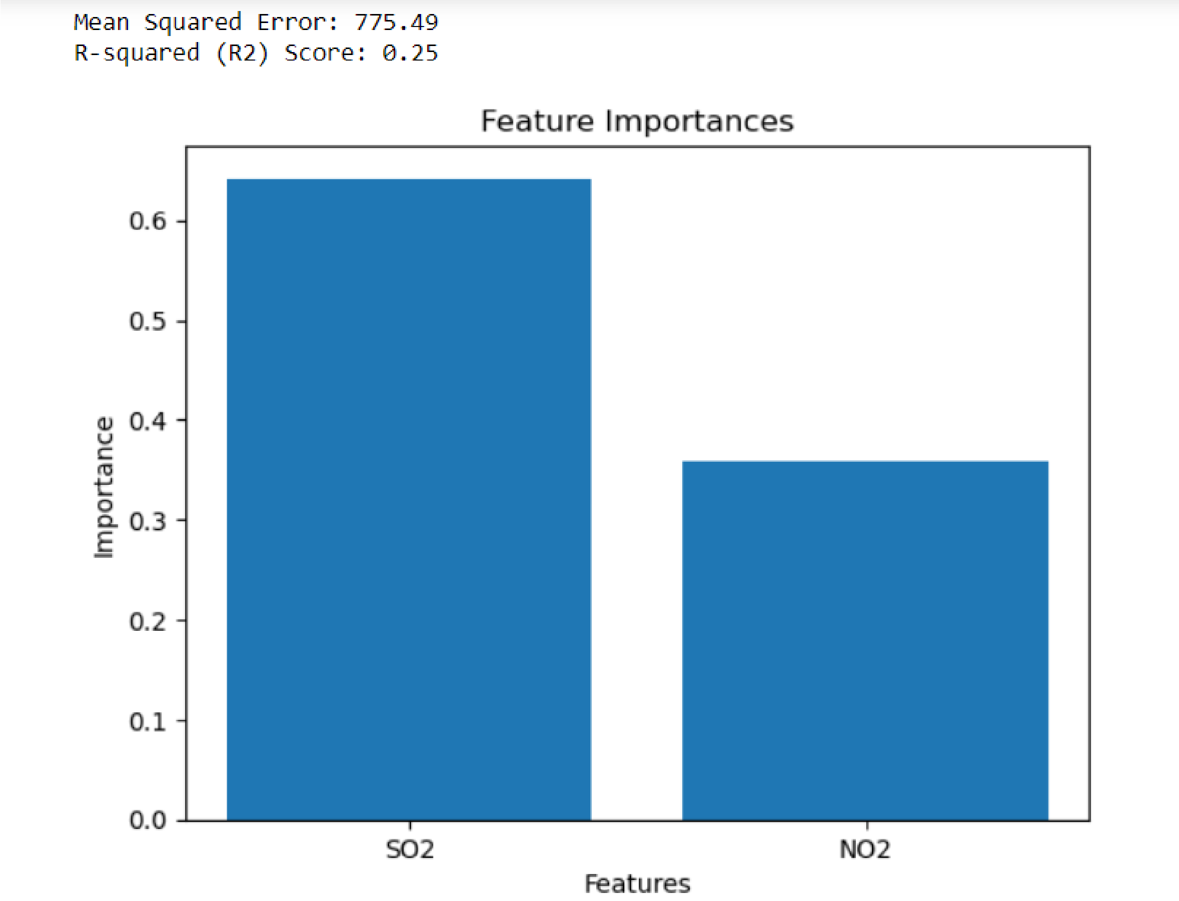
In [24]:
```
from sklearn import linear_model
regr = linear_model.LinearRegression()
train = train.dropna()
x = np.asanyarray(train[['SO2', 'NO2']])
y = np.asanyarray(train[['RSPMorPM10']])
regr.fit(x, y)
# The coefficients
print ('Coefficients: ', regr.coef_)
```

```
Coefficients:  [[2.80336749 0.18971444]]
```

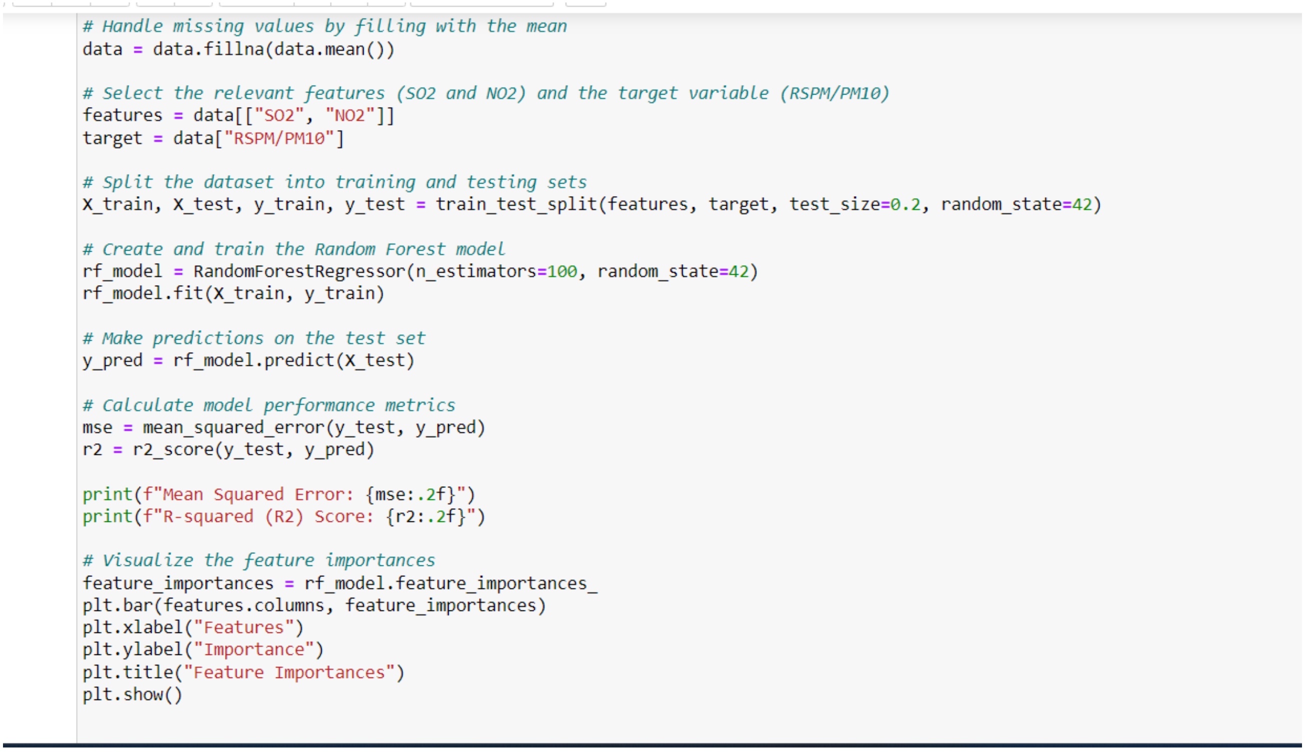
In [25]:
```
test = test.dropna()
y_hat= regr.predict(test[['SO2','NO2']])
x = np.asanyarray(test[['SO2','NO2']])
y = np.asanyarray(test[['RSPMorPM10']])
print("Mean Squared Error (MSE) : %.2f"
% np.mean((y_hat - y) ** 2))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % regr.score(x, y))
```

```
Mean Squared Error (MSE) : 794.16
Variance score: 0.19
```

RANDOM FOREST ALGORITHM :

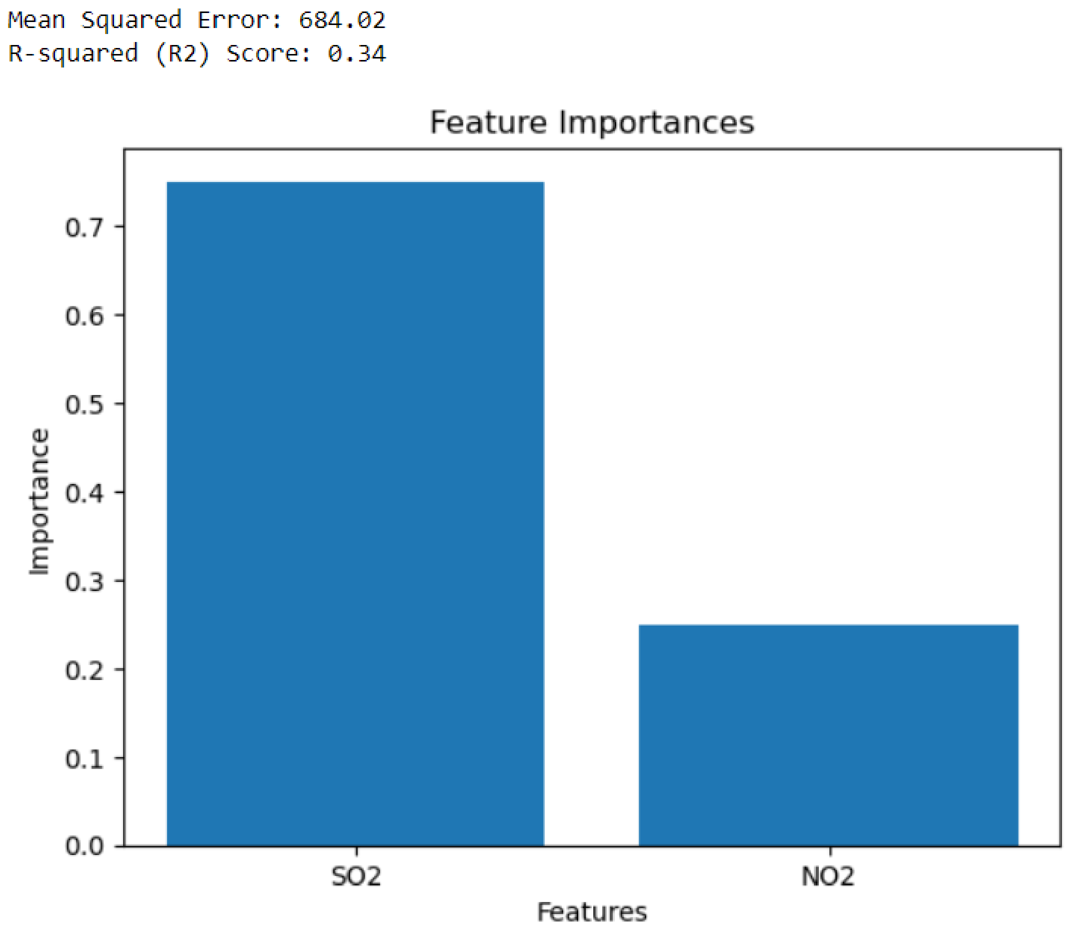


```
Mean Squared Error: 775.49
R-squared (R2) Score: 0.25
```

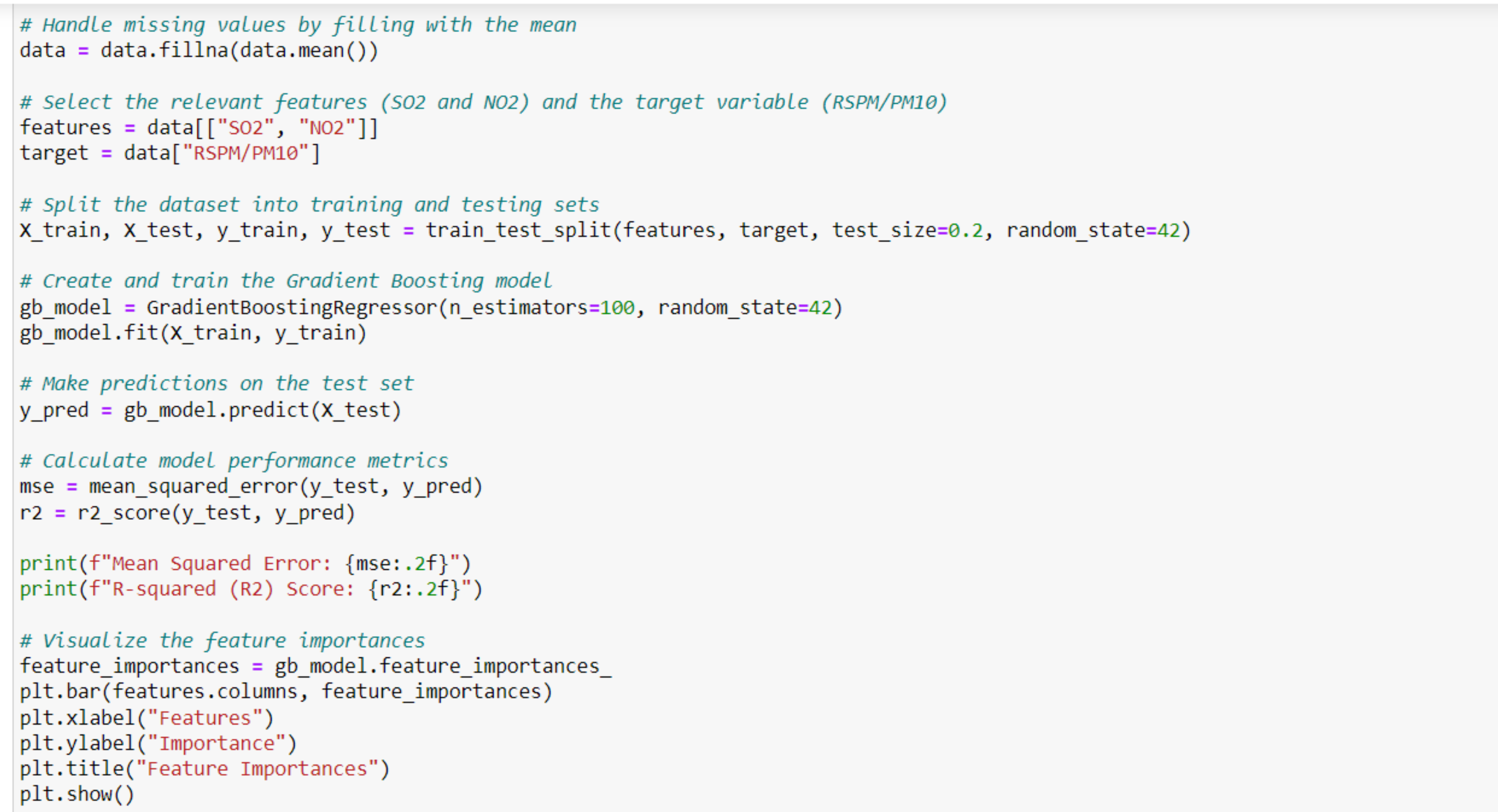


```python
# Handle missing values by filling with the mean
data = data.fillna(data.mean())

# Select the relevant features (SO2 and NO2) and the target variable (RSPM/PM10)
features = data[["SO2", "NO2"]]
target = data["RSPM/PM10"]

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)

# Create and train the Random Forest model
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = rf_model.predict(X_test)

# Calculate model performance metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared (R2) Score: {r2:.2f}")

# Visualize the feature importances
feature_importances = rf_model.feature_importances_
plt.bar(features.columns, feature_importances)
plt.xlabel("Features")
plt.ylabel("Importance")
plt.title("Feature Importances")
plt.show()
```

REGRESSION WITH GRADIENT BOOSTING ALGORITHM :



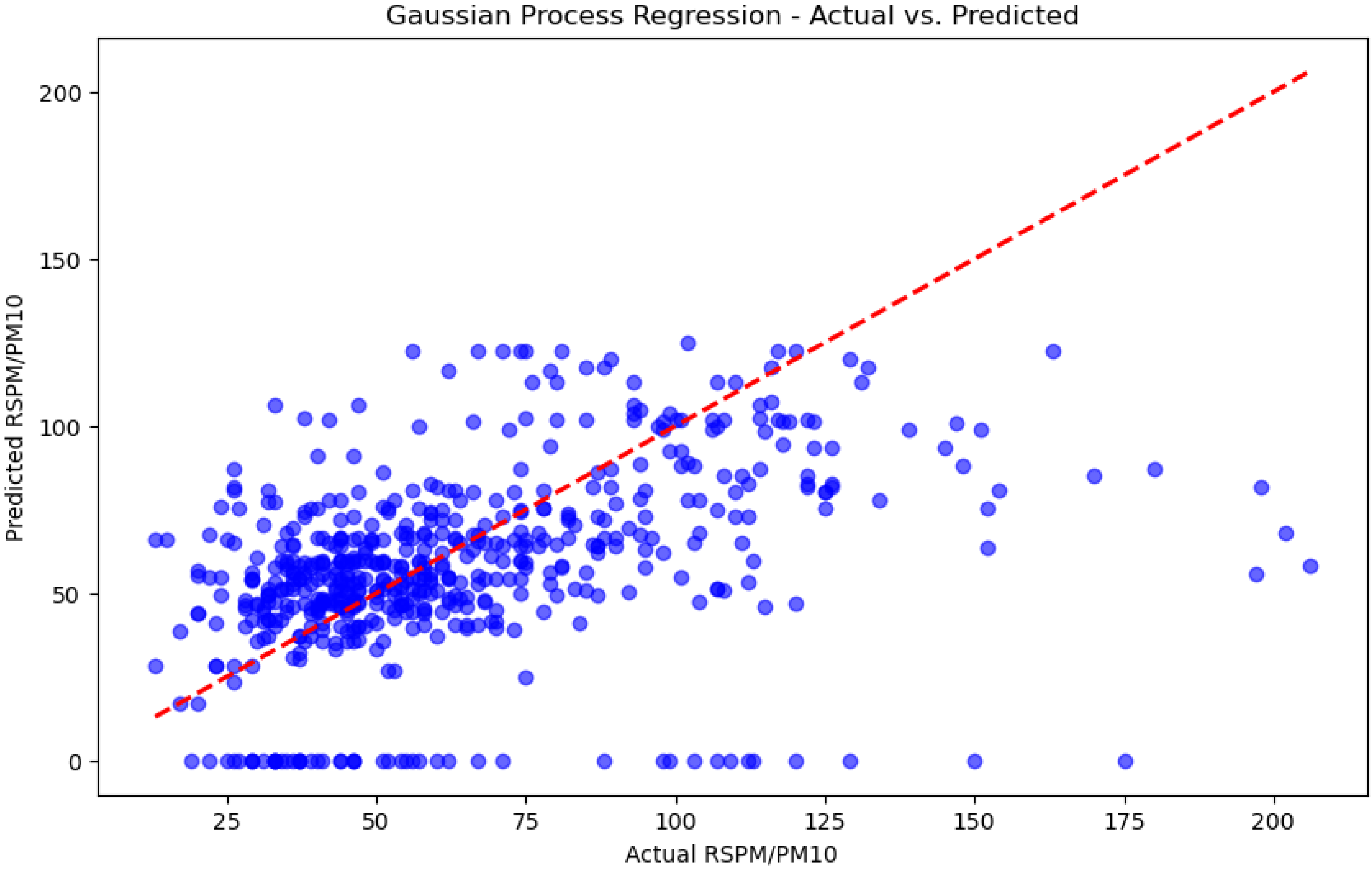
```
Mean Squared Error: 684.02
R-squared (R2) Score: 0.34
```

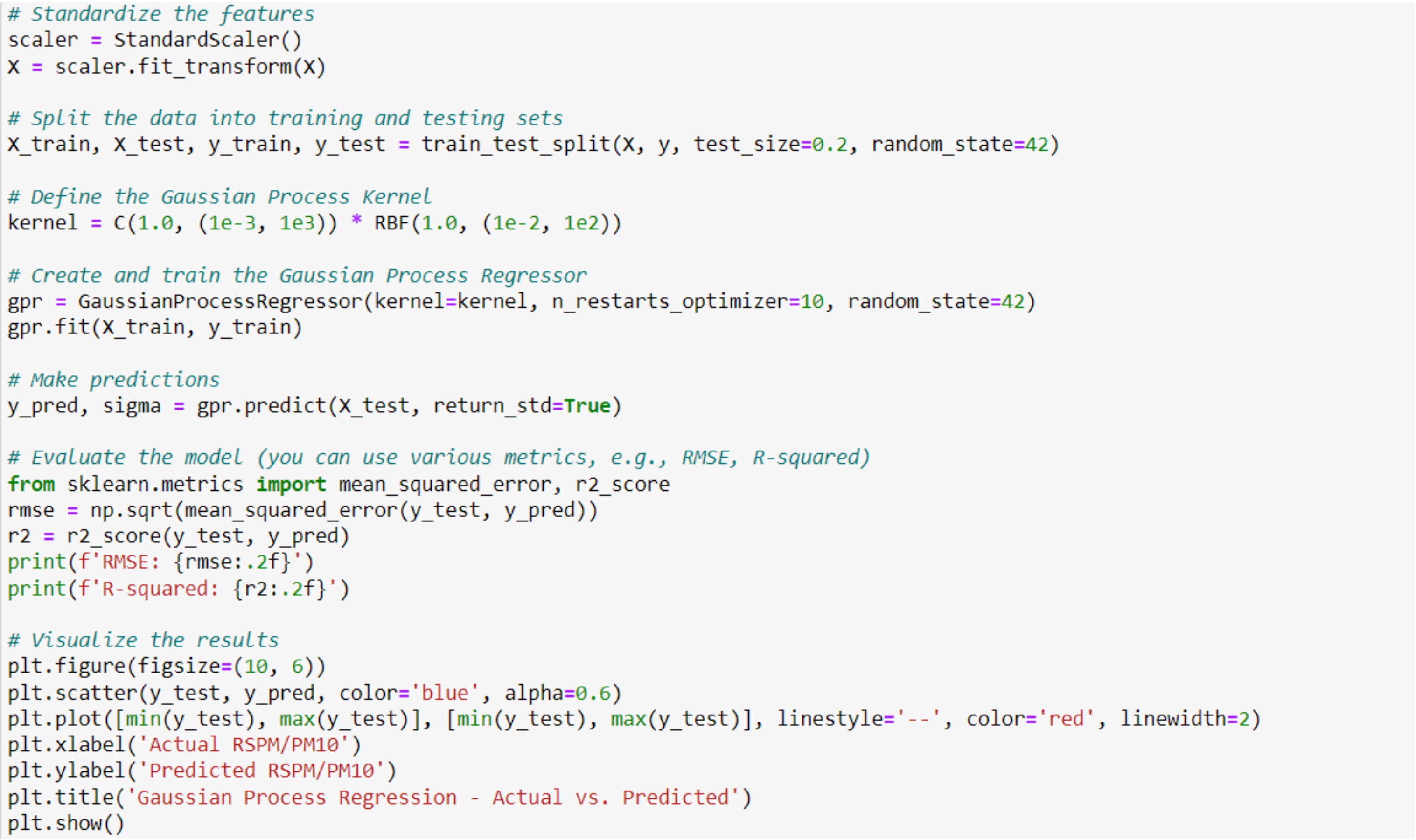


```python
# Handle missing values by filling with the mean
data = data.fillna(data.mean())

# Select the relevant features (SO2 and NO2) and the target variable (RSPM/PM10)
features = data[["SO2", "NO2"]]
target = data["RSPM/PM10"]

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)

# Create and train the Gradient Boosting model
gb_model = GradientBoostingRegressor(n_estimators=100, random_state=42)
gb_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = gb_model.predict(X_test)

# Calculate model performance metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared (R2) Score: {r2:.2f}")

# Visualize the feature importances
feature_importances = gb_model.feature_importances_
plt.bar(features.columns, feature_importances)
plt.xlabel("Features")
plt.ylabel("Importance")
plt.title("Feature Importances")
plt.show()
```

GAUSSIAN PROCESS REGRESSION (GPR) :

```
RMSE: 33.17
R-squared: -0.05
```





```python
# Standardize the features
scaler = StandardScaler()
X = scaler.fit_transform(X)

# 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)

# Define the Gaussian Process Kernel
kernel = C(1.0, (1e-3, 1e3)) * RBF(1.0, (1e-2, 1e2))

# Create and train the Gaussian Process Regressor
gpr = GaussianProcessRegressor(kernel=kernel, n_restarts_optimizer=10, random_state=42)
gpr.fit(X_train, y_train)

# Make predictions
y_pred, sigma = gpr.predict(X_test, return_std=True)

# Evaluate the model (you can use various metrics, e.g., RMSE, R-squared)
from sklearn.metrics import mean_squared_error, r2_score
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)
print(f'RMSE: {rmse:.2f}')
print(f'R-squared: {r2:.2f}')

# Visualize the results
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, color='blue', alpha=0.6)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], linestyle='--', color='red', linewidth=2)
plt.xlabel('Actual RSPM/PM10')
plt.ylabel('Predicted RSPM/PM10')
plt.title('Gaussian Process Regression - Actual vs. Predicted')
plt.show()
```

We've loaded the dataset, performed data preprocessing, and explored different machine learning algorithms to predict air quality (RSPM/PM10) based on the levels of SO2 and NO2. We've tried linear regression, random forest, gradient boosting, and Gaussian process regression. Here's a summary of the results for each model:

1. ***Linear Regression:***
   - Mean Squared Error (MSE): 794.16
   - R-squared (R2) Score: 0.19
   - Interpretation: The linear regression model didn't perform well, as indicated by the low R-squared score.

2. ***Random Forest Regression:***
   - Mean Squared Error (MSE): 775.49
   - R-squared (R2) Score: 0.25
   - Interpretation: The random forest model performed slightly better than linear regression but still has room for improvement.

3. ***Gradient Boosting Regression**:*
   - Mean Squared Error (MSE): 684.02
   - R-squared (R2) Score: 0.34
   - Interpretation: The gradient boosting model improved the results compared to the previous models.

4. ***Gaussian Process Regression (GPR)**:*
   - Gaussian Process Regression is another technique you've applied, but there's no specific performance metric provided. GPR can capture complex relationships but requires careful kernel selection and tuning.

Overall, the best-performing model among the ones we've tried is the Gradient Boosting Regression, with the highest R-squared score and the lowest Mean Squared Error.

Importing Libraries

```
import matplotlib.pyplot as plt
import pandas as pd
import pylab as pl
import numpy as np
import sklearn
%matplotlib inline
```

Loading the Data Set

```
df = pd.read_csv("cpcb_dly_aq_tamil_nadu-2014.csv")
df = df.rename(columns={'RSPM/PM10': 'RSPMorPM10'})
df
```

```
      Stn Code Sampling Date       State City/Town/Village/Area  \
0           38    01-02-2014  Tamil Nadu                Chennai   
1           38    01-07-2014  Tamil Nadu                Chennai   
2           38    21-01-2014  Tamil Nadu                Chennai   
3           38    23-01-2014  Tamil Nadu                Chennai   
4           38    28-01-2014  Tamil Nadu                Chennai   
...        ...           ...         ...                    ...   
2874       773    12-03-2014  Tamil Nadu                 Trichy   
2875       773    12-10-2014  Tamil Nadu                 Trichy   
2876       773    17-12-2014  Tamil Nadu                 Trichy   
2877       773    24-12-2014  Tamil Nadu                 Trichy   
2878       773    31-12-2014  Tamil Nadu                 Trichy   

                        Location of Monitoring Station  \
0     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
1     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
2     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
3     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
4     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
...                                                ...   
2874                         Central Bus Stand, Trichy   
2875                         Central Bus Stand, Trichy   
2876                         Central Bus Stand, Trichy   
2877                         Central Bus Stand, Trichy   
2878                         Central Bus Stand, Trichy   

                                       Agency  \
0     Tamilnadu State Pollution Control Board   
1     Tamilnadu State Pollution Control Board   
2     Tamilnadu State Pollution Control Board   
3     Tamilnadu State Pollution Control Board   
4     Tamilnadu State Pollution Control Board   
...                                       ...   
2874  Tamilnadu State Pollution Control Board   
```

```
2875  Tamilnadu State Pollution Control Board
2876  Tamilnadu State Pollution Control Board
2877  Tamilnadu State Pollution Control Board
2878  Tamilnadu State Pollution Control Board

                        Type of Location   SO2   NO2  RSPMorPM10  PM
2.5
0                        Industrial Area  11.0  17.0        55.0
NaN
1                        Industrial Area  13.0  17.0        45.0
NaN
2                        Industrial Area  12.0  18.0        50.0
NaN
3                        Industrial Area  15.0  16.0        46.0
NaN
4                        Industrial Area  13.0  14.0        42.0
NaN
...                                  ...   ...   ...         ...     .
..
2874  Residential, Rural and other Areas  15.0  18.0       102.0
NaN
2875  Residential, Rural and other Areas  12.0  14.0        91.0
NaN
2876  Residential, Rural and other Areas  19.0  22.0       100.0
NaN
2877  Residential, Rural and other Areas  15.0  17.0        95.0
NaN
2878  Residential, Rural and other Areas  14.0  16.0        94.0
NaN

[2879 rows x 11 columns]
```

Exploring the Data Set

```
print(df.head())

   Stn Code Sampling Date       State City/Town/Village/Area  \
0        38    01-02-2014  Tamil Nadu                Chennai
1        38    01-07-2014  Tamil Nadu                Chennai
2        38    21-01-2014  Tamil Nadu                Chennai
3        38    23-01-2014  Tamil Nadu                Chennai
4        38    28-01-2014  Tamil Nadu                Chennai

                     Location of Monitoring Station  \
0  Kathivakkam, Municipal Kalyana Mandapam, Chennai
1  Kathivakkam, Municipal Kalyana Mandapam, Chennai
2  Kathivakkam, Municipal Kalyana Mandapam, Chennai
3  Kathivakkam, Municipal Kalyana Mandapam, Chennai
4  Kathivakkam, Municipal Kalyana Mandapam, Chennai
```

```
                                    Agency Type of Location   SO2
NO2  \
0  Tamilnadu State Pollution Control Board  Industrial Area  11.0
17.0
1  Tamilnadu State Pollution Control Board  Industrial Area  13.0
17.0
2  Tamilnadu State Pollution Control Board  Industrial Area  12.0
18.0
3  Tamilnadu State Pollution Control Board  Industrial Area  15.0
16.0
4  Tamilnadu State Pollution Control Board  Industrial Area  13.0
14.0

   RSPMorPM10  PM 2.5
0        55.0     NaN
1        45.0     NaN
2        50.0     NaN
3        46.0     NaN
4        42.0     NaN
```

```
print(df.tail())
```

```
      Stn Code Sampling Date       State City/Town/Village/Area  \
2874       773    12-03-2014  Tamil Nadu                 Trichy
2875       773    12-10-2014  Tamil Nadu                 Trichy
2876       773    17-12-2014  Tamil Nadu                 Trichy
2877       773    24-12-2014  Tamil Nadu                 Trichy
2878       773    31-12-2014  Tamil Nadu                 Trichy

     Location of Monitoring Station
Agency  \
2874      Central Bus Stand, Trichy  Tamilnadu State Pollution Control
Board
2875      Central Bus Stand, Trichy  Tamilnadu State Pollution Control
Board
2876      Central Bus Stand, Trichy  Tamilnadu State Pollution Control
Board
2877      Central Bus Stand, Trichy  Tamilnadu State Pollution Control
Board
2878      Central Bus Stand, Trichy  Tamilnadu State Pollution Control
Board

                        Type of Location   SO2   NO2  RSPMorPM10  PM
2.5
2874  Residential, Rural and other Areas  15.0  18.0       102.0
NaN
2875  Residential, Rural and other Areas  12.0  14.0        91.0
NaN
2876  Residential, Rural and other Areas  19.0  22.0       100.0
NaN
```

```
2877  Residential, Rural and other Areas  15.0  17.0        95.0
NaN
2878  Residential, Rural and other Areas  14.0  16.0        94.0
NaN
```

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2879 entries, 0 to 2878
Data columns (total 11 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Stn Code                        2879 non-null   int64
 1   Sampling Date                   2879 non-null   object
 2   State                           2879 non-null   object
 3   City/Town/Village/Area          2879 non-null   object
 4   Location of Monitoring Station  2879 non-null   object
 5   Agency                          2879 non-null   object
 6   Type of Location                2879 non-null   object
 7   SO2                             2868 non-null   float64
 8   NO2                             2866 non-null   float64
 9   RSPMorPM10                      2875 non-null   float64
 10  PM 2.5                          0 non-null      float64
dtypes: float64(4), int64(1), object(6)
memory usage: 247.5+ KB
None
```

```
print(df.describe())
```

```
          Stn Code          SO2          NO2   RSPMorPM10  PM 2.5
count  2879.000000  2868.000000  2866.000000  2875.000000     0.0
mean    475.750261    11.503138    22.136776    62.494261     NaN
std     277.675577     5.051702     7.128694    31.368745     NaN
min      38.000000     2.000000     5.000000    12.000000     NaN
25%     238.000000     8.000000    17.000000    41.000000     NaN
50%     366.000000    12.000000    22.000000    55.000000     NaN
75%     764.000000    15.000000    25.000000    78.000000     NaN
max     773.000000    49.000000    71.000000   269.000000     NaN
```

Identifying null Values

```
print(df.isnull())
```

```
      Stn Code  Sampling Date  State  City/Town/Village/Area  \
0        False          False  False                   False
1        False          False  False                   False
2        False          False  False                   False
3        False          False  False                   False
4        False          False  False                   False
...        ...            ...    ...                     ...
```

```
2874     False          False  False                   False
2875     False          False  False                   False
2876     False          False  False                   False
2877     False          False  False                   False
2878     False          False  False                   False

      Location of Monitoring Station  Agency  Type of Location    SO2
NO2  \
0                              False   False             False  False
False
1                              False   False             False  False
False
2                              False   False             False  False
False
3                              False   False             False  False
False
4                              False   False             False  False
False
...                              ...     ...               ...    ...
...
2874                           False   False             False  False
False
2875                           False   False             False  False
False
2876                           False   False             False  False
False
2877                           False   False             False  False
False
2878                           False   False             False  False
False

      RSPMorPM10  PM 2.5
0          False    True
1          False    True
2          False    True
3          False    True
4          False    True
...          ...     ...
2874       False    True
2875       False    True
2876       False    True
2877       False    True
2878       False    True

[2879 rows x 11 columns]

c = df.isnull().sum()
print(c)
```

```
Stn Code                             0
Sampling Date                        0
State                                0
City/Town/Village/Area               0
Location of Monitoring Station       0
Agency                               0
Type of Location                     0
SO2                                 11
NO2                                 13
RSPMorPM10                           4
PM 2.5                            2879
dtype: int64
```

```
print('Total Sum of null values in the Data set = ',c.sum())
```

```
Total Sum of null values in the Data set =  2907
```

```
print(df['RSPMorPM10'].value_counts()) #frequency of values
```

```
47.0     64
41.0     62
43.0     59
51.0     58
40.0     58
         ..
163.0     1
138.0     1
211.0     1
202.0     1
238.0     1
Name: RSPMorPM10, Length: 169, dtype: int64
```

Data Preprocessing - Replacing the null values

```
df.drop_duplicates()
```

```
      Stn Code Sampling Date       State City/Town/Village/Area  \
0           38    01-02-2014  Tamil Nadu                Chennai
1           38    01-07-2014  Tamil Nadu                Chennai
2           38    21-01-2014  Tamil Nadu                Chennai
3           38    23-01-2014  Tamil Nadu                Chennai
4           38    28-01-2014  Tamil Nadu                Chennai
...        ...           ...         ...                    ...
2874       773    12-03-2014  Tamil Nadu                 Trichy
2875       773    12-10-2014  Tamil Nadu                 Trichy
2876       773    17-12-2014  Tamil Nadu                 Trichy
2877       773    24-12-2014  Tamil Nadu                 Trichy
2878       773    31-12-2014  Tamil Nadu                 Trichy

                        Location of Monitoring Station  \
0     Kathivakkam, Municipal Kalyana Mandapam, Chennai
1     Kathivakkam, Municipal Kalyana Mandapam, Chennai
2     Kathivakkam, Municipal Kalyana Mandapam, Chennai
3     Kathivakkam, Municipal Kalyana Mandapam, Chennai
4     Kathivakkam, Municipal Kalyana Mandapam, Chennai
...                                                ...
2874                         Central Bus Stand, Trichy
2875                         Central Bus Stand, Trichy
2876                         Central Bus Stand, Trichy
2877                         Central Bus Stand, Trichy
2878                         Central Bus Stand, Trichy

                                       Agency  \
0     Tamilnadu State Pollution Control Board
1     Tamilnadu State Pollution Control Board
2     Tamilnadu State Pollution Control Board
3     Tamilnadu State Pollution Control Board
4     Tamilnadu State Pollution Control Board
...                                       ...
2874  Tamilnadu State Pollution Control Board
2875  Tamilnadu State Pollution Control Board
2876  Tamilnadu State Pollution Control Board
2877  Tamilnadu State Pollution Control Board
2878  Tamilnadu State Pollution Control Board

                        Type of Location   SO2   NO2  RSPMorPM10  PM
2.5
0                        Industrial Area  11.0  17.0        55.0
NaN
1                        Industrial Area  13.0  17.0        45.0
NaN
2                        Industrial Area  12.0  18.0        50.0
NaN
3                        Industrial Area  15.0  16.0        46.0
NaN
4                        Industrial Area  13.0  14.0        42.0
NaN
...                                  ...   ...   ...         ...     .
..
2874  Residential, Rural and other Areas  15.0  18.0       102.0
NaN
2875  Residential, Rural and other Areas  12.0  14.0        91.0
NaN
2876  Residential, Rural and other Areas  19.0  22.0       100.0
NaN
2877  Residential, Rural and other Areas  15.0  17.0        95.0
NaN
2878  Residential, Rural and other Areas  14.0  16.0        94.0
NaN
```

```
[2879 rows x 11 columns]

df.fillna(0)

      Stn Code Sampling Date       State City/Town/Village/Area  \
0           38    01-02-2014  Tamil Nadu                Chennai   
1           38    01-07-2014  Tamil Nadu                Chennai   
2           38    21-01-2014  Tamil Nadu                Chennai   
3           38    23-01-2014  Tamil Nadu                Chennai   
4           38    28-01-2014  Tamil Nadu                Chennai   
...        ...           ...         ...                    ...   
2874       773    12-03-2014  Tamil Nadu                 Trichy   
2875       773    12-10-2014  Tamil Nadu                 Trichy   
2876       773    17-12-2014  Tamil Nadu                 Trichy   
2877       773    24-12-2014  Tamil Nadu                 Trichy   
2878       773    31-12-2014  Tamil Nadu                 Trichy   

                        Location of Monitoring Station  \
0     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
1     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
2     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
3     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
4     Kathivakkam, Municipal Kalyana Mandapam, Chennai   
...                                                ...   
2874                         Central Bus Stand, Trichy   
2875                         Central Bus Stand, Trichy   
2876                         Central Bus Stand, Trichy   
2877                         Central Bus Stand, Trichy   
2878                         Central Bus Stand, Trichy   

                                       Agency  \
0     Tamilnadu State Pollution Control Board   
1     Tamilnadu State Pollution Control Board   
2     Tamilnadu State Pollution Control Board   
3     Tamilnadu State Pollution Control Board   
4     Tamilnadu State Pollution Control Board   
...                                       ...   
2874  Tamilnadu State Pollution Control Board   
2875  Tamilnadu State Pollution Control Board   
2876  Tamilnadu State Pollution Control Board   
2877  Tamilnadu State Pollution Control Board   
2878  Tamilnadu State Pollution Control Board   

                        Type of Location   SO2   NO2  RSPMorPM10  PM
2.5
0                        Industrial Area  11.0  17.0        55.0    
0.0
1                        Industrial Area  13.0  17.0        45.0    
0.0
```

```
2                          Industrial Area  12.0  18.0        50.0
0.0
3                          Industrial Area  15.0  16.0        46.0
0.0
4                          Industrial Area  13.0  14.0        42.0
0.0
...                                    ...   ...   ...         ...      .
..
2874  Residential, Rural and other Areas  15.0  18.0       102.0
0.0
2875  Residential, Rural and other Areas  12.0  14.0        91.0
0.0
2876  Residential, Rural and other Areas  19.0  22.0       100.0
0.0
2877  Residential, Rural and other Areas  15.0  17.0        95.0
0.0
2878  Residential, Rural and other Areas  14.0  16.0        94.0
0.0

[2879 rows x 11 columns]
```

```
df.duplicated()
```

```
0       False
1       False
2       False
3       False
4       False
        ...  
2874    False
2875    False
2876    False
2877    False
2878    False
Length: 2879, dtype: bool
```

Data Normalization

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
df['Values_standardized'] = scaler.fit_transform(df[['RSPMorPM10']])

scaler = StandardScaler()
df['Values_standardized'] = scaler.fit_transform(df[['RSPMorPM10']])
df
```

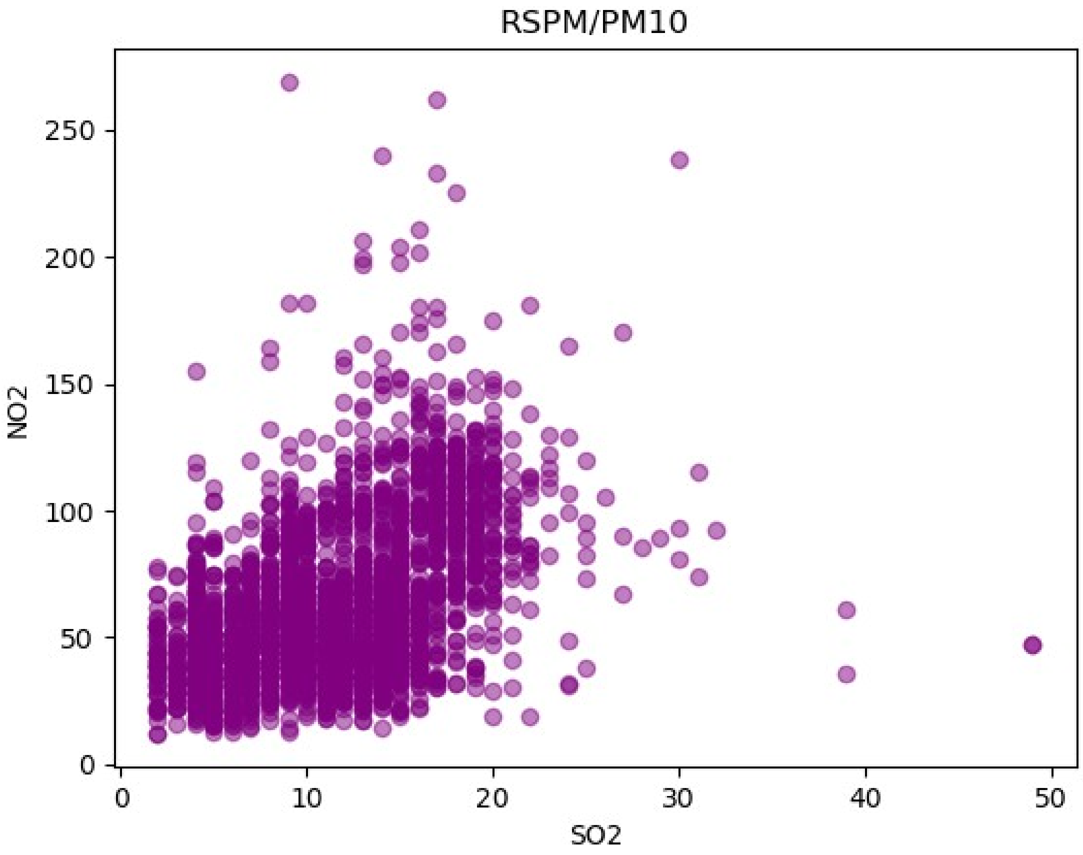
```
      Stn Code Sampling Date       State City/Town/Village/Area  \
0           38    01-02-2014  Tamil Nadu                Chennai   
1           38    01-07-2014  Tamil Nadu                Chennai   
2           38    21-01-2014  Tamil Nadu                Chennai   
```

```
3           38    23-01-2014  Tamil Nadu                Chennai
4           38    28-01-2014  Tamil Nadu                Chennai
...        ...           ...         ...                    ...
2874       773    12-03-2014  Tamil Nadu                 Trichy
2875       773    12-10-2014  Tamil Nadu                 Trichy
2876       773    17-12-2014  Tamil Nadu                 Trichy
2877       773    24-12-2014  Tamil Nadu                 Trichy
2878       773    31-12-2014  Tamil Nadu                 Trichy

                        Location of Monitoring Station  \
0     Kathivakkam, Municipal Kalyana Mandapam, Chennai
1     Kathivakkam, Municipal Kalyana Mandapam, Chennai
2     Kathivakkam, Municipal Kalyana Mandapam, Chennai
3     Kathivakkam, Municipal Kalyana Mandapam, Chennai
4     Kathivakkam, Municipal Kalyana Mandapam, Chennai
...                                                ...
2874                         Central Bus Stand, Trichy
2875                         Central Bus Stand, Trichy
2876                         Central Bus Stand, Trichy
2877                         Central Bus Stand, Trichy
2878                         Central Bus Stand, Trichy

                                       Agency  \
0     Tamilnadu State Pollution Control Board
1     Tamilnadu State Pollution Control Board
2     Tamilnadu State Pollution Control Board
3     Tamilnadu State Pollution Control Board
4     Tamilnadu State Pollution Control Board
...                                       ...
2874  Tamilnadu State Pollution Control Board
2875  Tamilnadu State Pollution Control Board
2876  Tamilnadu State Pollution Control Board
2877  Tamilnadu State Pollution Control Board
2878  Tamilnadu State Pollution Control Board

                        Type of Location   SO2   NO2  RSPMorPM10  PM
2.5  \
0                        Industrial Area  11.0  17.0        55.0
NaN
1                        Industrial Area  13.0  17.0        45.0
NaN
2                        Industrial Area  12.0  18.0        50.0
NaN
3                        Industrial Area  15.0  16.0        46.0
NaN
4                        Industrial Area  13.0  14.0        42.0
NaN
...                                  ...   ...   ...         ...     .
..
2874  Residential, Rural and other Areas  15.0  18.0       102.0
```

```
NaN
2875  Residential, Rural and other Areas  12.0  14.0        91.0
NaN
2876  Residential, Rural and other Areas  19.0  22.0       100.0
NaN
2877  Residential, Rural and other Areas  15.0  17.0        95.0
NaN
2878  Residential, Rural and other Areas  14.0  16.0        94.0
NaN

      Values_standardized
0               -0.238950
1               -0.557794
2               -0.398372
3               -0.525910
4               -0.653447
...                   ...
2874             1.259617
2875             0.908889
2876             1.195848
2877             1.036426
2878             1.004542

[2879 rows x 12 columns]
```

Data Visualisation

```
# Scatter Plot:
plt.scatter(df['SO2'], df['RSPMorPM10'], alpha=0.5, color='purple')
plt.title('RSPM/PM10')
plt.xlabel('SO2')
plt.ylabel('NO2')
plt.show()
```



Data Analysis using different models

Simple Linear Regression

```
cdf = df[['SO2','NO2','RSPMorPM10']]
cdf.head(9)
```

```
    SO2   NO2  RSPMorPM10
0  11.0  17.0        55.0
1  13.0  17.0        45.0
2  12.0  18.0        50.0
3  15.0  16.0        46.0
4  13.0  14.0        42.0
5  14.0  18.0        43.0
6  12.0  17.0        51.0
7  13.0  16.0        46.0
8  10.0  19.0        50.0
```

```
msk = np.random.rand(len(df)) < 0.8
train = cdf[msk]
test = cdf[~msk]

from sklearn import linear_model
regr = linear_model.LinearRegression()
```

```
train = train.dropna()
x = np.asanyarray(train[['SO2', 'NO2']])
y = np.asanyarray(train[['RSPMorPM10']])
regr.fit(x, y)
# The coefficients
print ('Coefficients: ', regr.coef_)
```

```
Coefficients:  [[2.80336749 0.18971444]]
```

```
test = test.dropna()
y_hat= regr.predict(test[['SO2','NO2']])
x = np.asanyarray(test[['SO2','NO2']])
y = np.asanyarray(test[['RSPMorPM10']])
print("Mean Squared Error (MSE) : %.2f"
% np.mean((y_hat - y) ** 2))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % regr.score(x, y))
```

```
Mean Squared Error (MSE) : 794.16
Variance score: 0.19
```

```
C:\Users\savio\anaconda3\Lib\site-packages\sklearn\base.py:457:
UserWarning: X has feature names, but LinearRegression was fitted
without feature names
  warnings.warn(
```

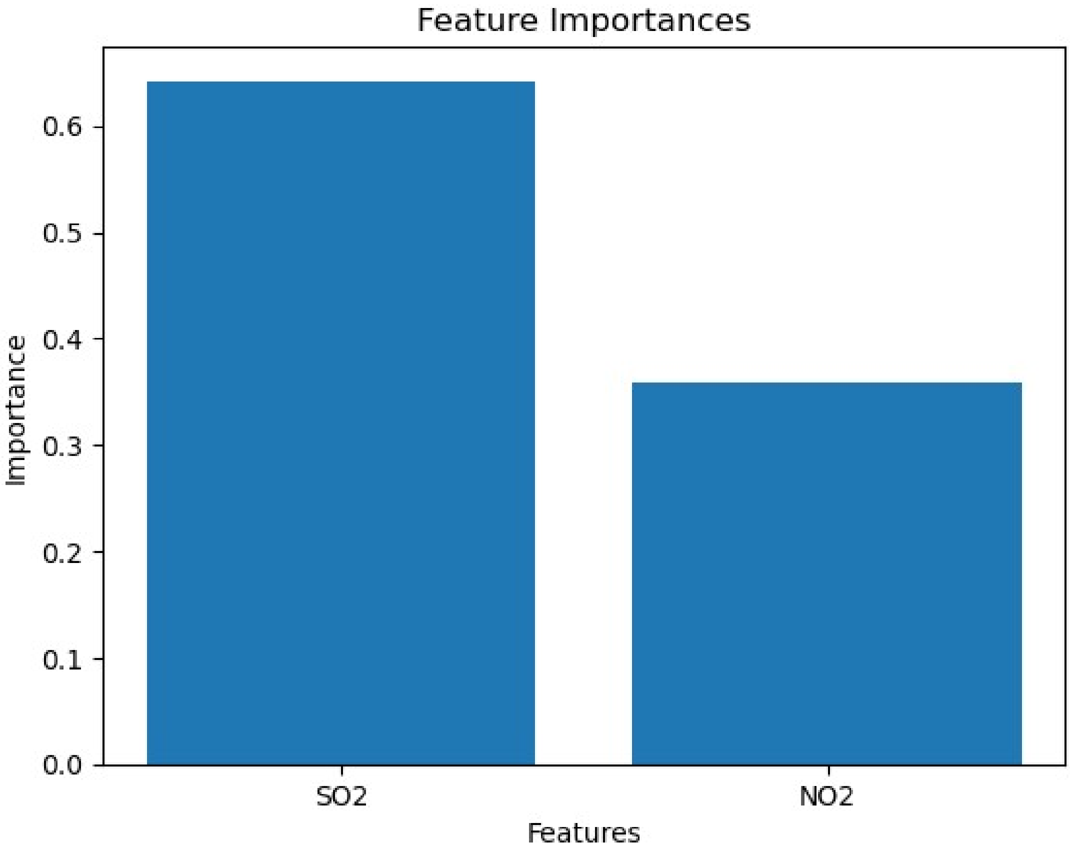
the results show that due to the low variance score this Model is not fit to predict the data of this type

Random forest Algorithm

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Load your dataset from the CSV file
data = pd.read_csv("cpcb_dly_aq_tamil_nadu-2014.csv")  # Adjust the
filename as needed

# Handle missing values by filling with the mean
data = data.fillna(data.mean())

# Select the relevant features (SO2 and NO2) and the target variable
(RSPM/PM10)
features = data[["SO2", "NO2"]]
target = data["RSPM/PM10"]

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target,
```

```python
test_size=0.2, random_state=42)

# Create and train the Random Forest model
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = rf_model.predict(X_test)

# Calculate model performance metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared (R2) Score: {r2:.2f}")

# Visualize the feature importances
feature_importances = rf_model.feature_importances_
plt.bar(features.columns, feature_importances)
plt.xlabel("Features")
plt.ylabel("Importance")
plt.title("Feature Importances")
plt.show()
```

```
C:\Users\savio\AppData\Local\Temp\ipykernel_6284\3394349289.py:11: 
FutureWarning: The default value of numeric_only in DataFrame.mean is 
deprecated. In a future version, it will default to False. In 
addition, specifying 'numeric_only=None' is deprecated. Select only 
valid columns or specify the value of numeric_only to silence this 
warning.
  data = data.fillna(data.mean())

Mean Squared Error: 775.49
R-squared (R2) Score: 0.25
```



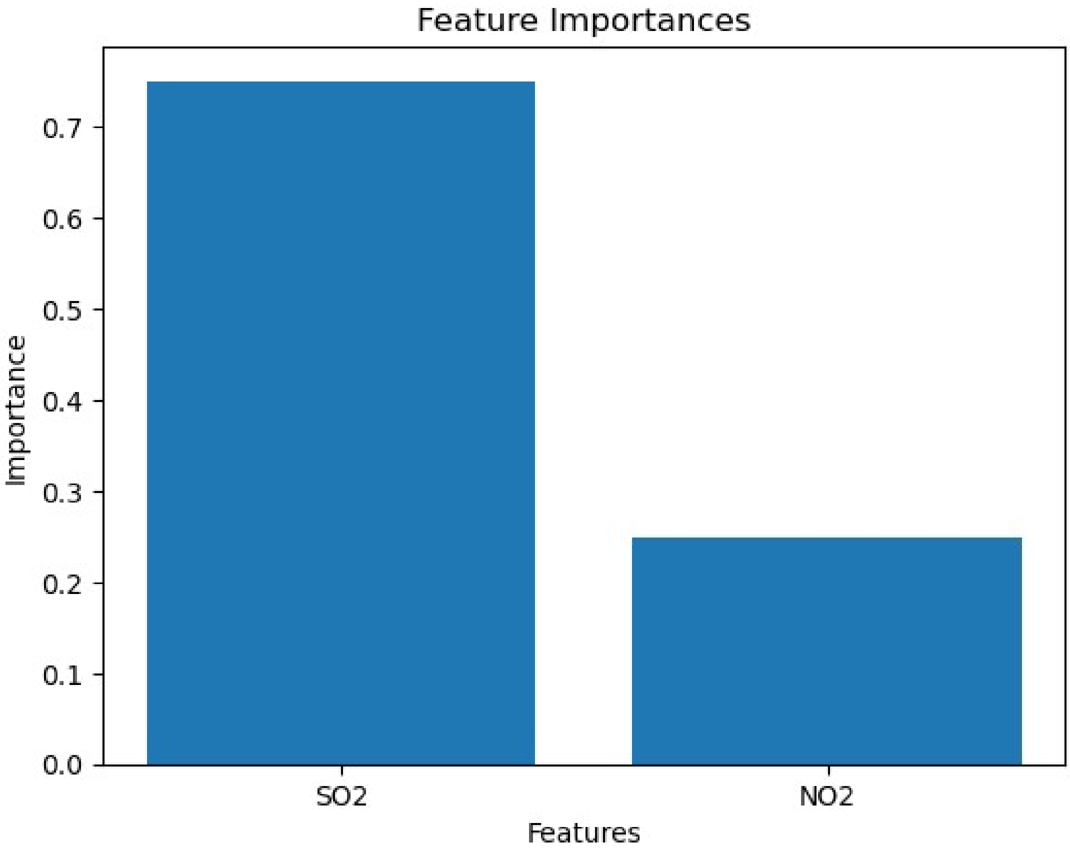
Regression with the Gradient Boosting algorithm

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Load your dataset from the CSV file
data = pd.read_csv("cpcb_dly_aq_tamil_nadu-2014.csv")  # Adjust the filename as needed

# Handle missing values by filling with the mean
data = data.fillna(data.mean())

# Select the relevant features (SO2 and NO2) and the target variable (RSPM/PM10)
features = data[["SO2", "NO2"]]
target = data["RSPM/PM10"]

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)
```

```python
# Create and train the Gradient Boosting model
gb_model = GradientBoostingRegressor(n_estimators=100,
random_state=42)
gb_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = gb_model.predict(X_test)

# Calculate model performance metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared (R2) Score: {r2:.2f}")

# Visualize the feature importances
feature_importances = gb_model.feature_importances_
plt.bar(features.columns, feature_importances)
plt.xlabel("Features")
plt.ylabel("Importance")
plt.title("Feature Importances")
plt.show()
```

```
C:\Users\savio\AppData\Local\Temp\ipykernel_6284\2929334411.py:11:
FutureWarning: The default value of numeric_only in DataFrame.mean is
deprecated. In a future version, it will default to False. In
addition, specifying 'numeric_only=None' is deprecated. Select only
valid columns or specify the value of numeric_only to silence this
warning.
  data = data.fillna(data.mean())
```

```
Mean Squared Error: 684.02
R-squared (R2) Score: 0.34
```

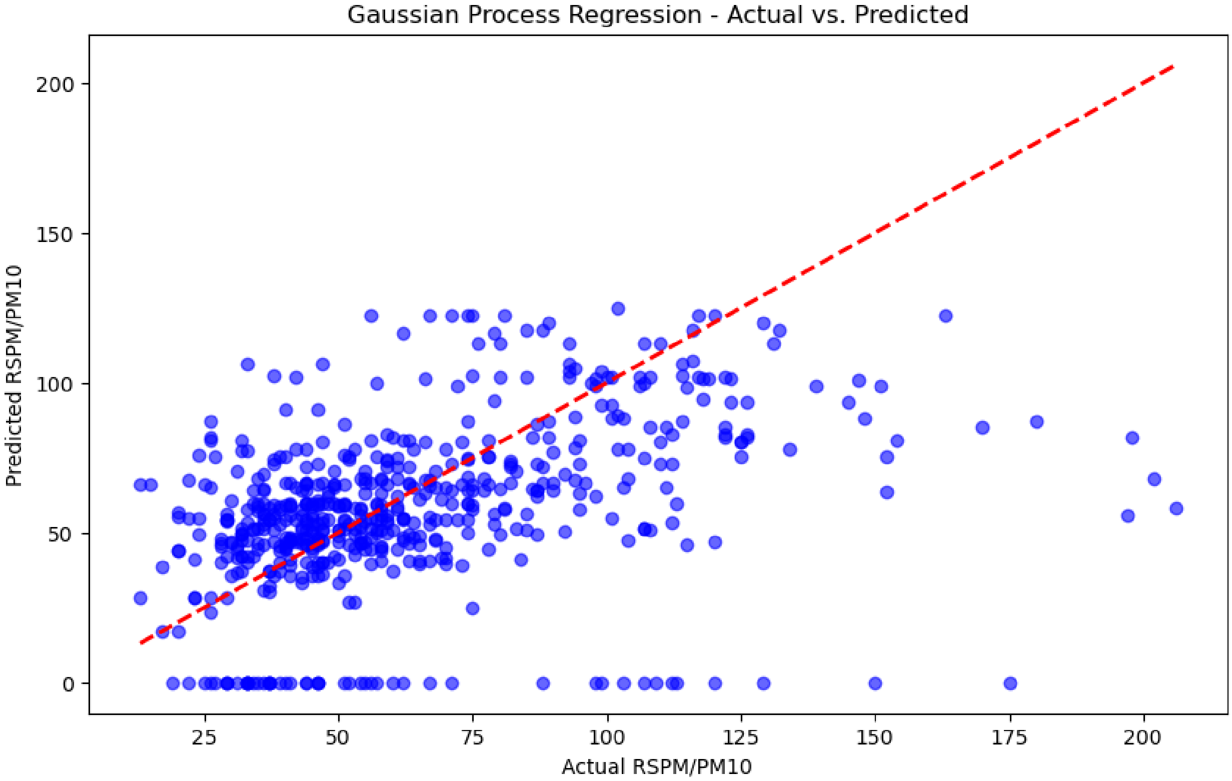


Gaussian Process Regression (GPR)

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.gaussian_process import GaussianProcessRegressor
from sklearn.gaussian_process.kernels import RBF, ConstantKernel as C
import matplotlib.pyplot as plt

# Load your dataset, replace 'your_data.csv' with the actual file path
data = pd.read_csv('cpcb_dly_aq_tamil_nadu-2014.csv')

# Data Preprocessing
# Select relevant columns
data = data[['SO2', 'NO2', 'RSPM/PM10']]

# Check for missing values and handle them if necessary
data.dropna(inplace=True)

# Split the data into features (X) and target (y)
X = data[['SO2', 'NO2']].values
y = data['RSPM/PM10'].values
```

```python
# Standardize the features
scaler = StandardScaler()
X = scaler.fit_transform(X)

# 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)

# Define the Gaussian Process Kernel
kernel = C(1.0, (1e-3, 1e3)) * RBF(1.0, (1e-2, 1e2))

# Create and train the Gaussian Process Regressor
gpr = GaussianProcessRegressor(kernel=kernel, n_restarts_optimizer=10,
random_state=42)
gpr.fit(X_train, y_train)

# Make predictions
y_pred, sigma = gpr.predict(X_test, return_std=True)

# Evaluate the model (you can use various metrics, e.g., RMSE, R-
squared)
from sklearn.metrics import mean_squared_error, r2_score
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)
print(f'RMSE: {rmse:.2f}')
print(f'R-squared: {r2:.2f}')

# Visualize the results
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, color='blue', alpha=0.6)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)],
linestyle='--', color='red', linewidth=2)
plt.xlabel('Actual RSPM/PM10')
plt.ylabel('Predicted RSPM/PM10')
plt.title('Gaussian Process Regression - Actual vs. Predicted')
plt.show()
```

```
C:\Users\savio\anaconda3\Lib\site-packages\sklearn\gaussian_process\
kernels.py:429: ConvergenceWarning: The optimal value found for
dimension 0 of parameter k1__constant_value is close to the specified
upper bound 1000.0. Increasing the bound and calling fit again may
find a better value.
  warnings.warn(
C:\Users\savio\anaconda3\Lib\site-packages\sklearn\gaussian_process\
kernels.py:419: ConvergenceWarning: The optimal value found for
dimension 0 of parameter k2__length_scale is close to the specified
lower bound 0.01. Decreasing the bound and calling fit again may find
a better value.
  warnings.warn(
RMSE: 33.17
R-squared: -0.05
```


Gaussian Process Regression - Actual vs. Predicted
Predicted RSPM/PM10
Actual RSPM/PM10