Lab Project: PolluMonitor

Name: Mahmudul Hasan Tamal

ID: 24341131

Sec: 07

Fall- 24

Course: CSE422

Table of Contents:	**Page No**.
● Introduction	3
● Dataset Description ○ Source ○ Description ○ Heatmap ○ Imbalanced Dataset	3-5
● Dataset preprocessing ○ Categorical Value ○ Null Value ○ Correlation ○ Feature Scaling	5-6
● Splitting	6
● Model Training & Testing ○ Decision Tree ○ Logistic Regression ○ Naive Bayes	7
● Comparison Analysis ○ Prediction Accuracy Bar-Chart ○ Precision & Recall Comparison ○ Confusion Matrices	8-9
● Conclusion	9

Introduction:

Air pollution is a major problem around the world, affecting people's health, the environment, and the climate. It's super important to find better ways to track and manage air quality. This project aims to predict air quality predictions accurately and it uses machine learning to monitor air pollution.

Dataset Description:

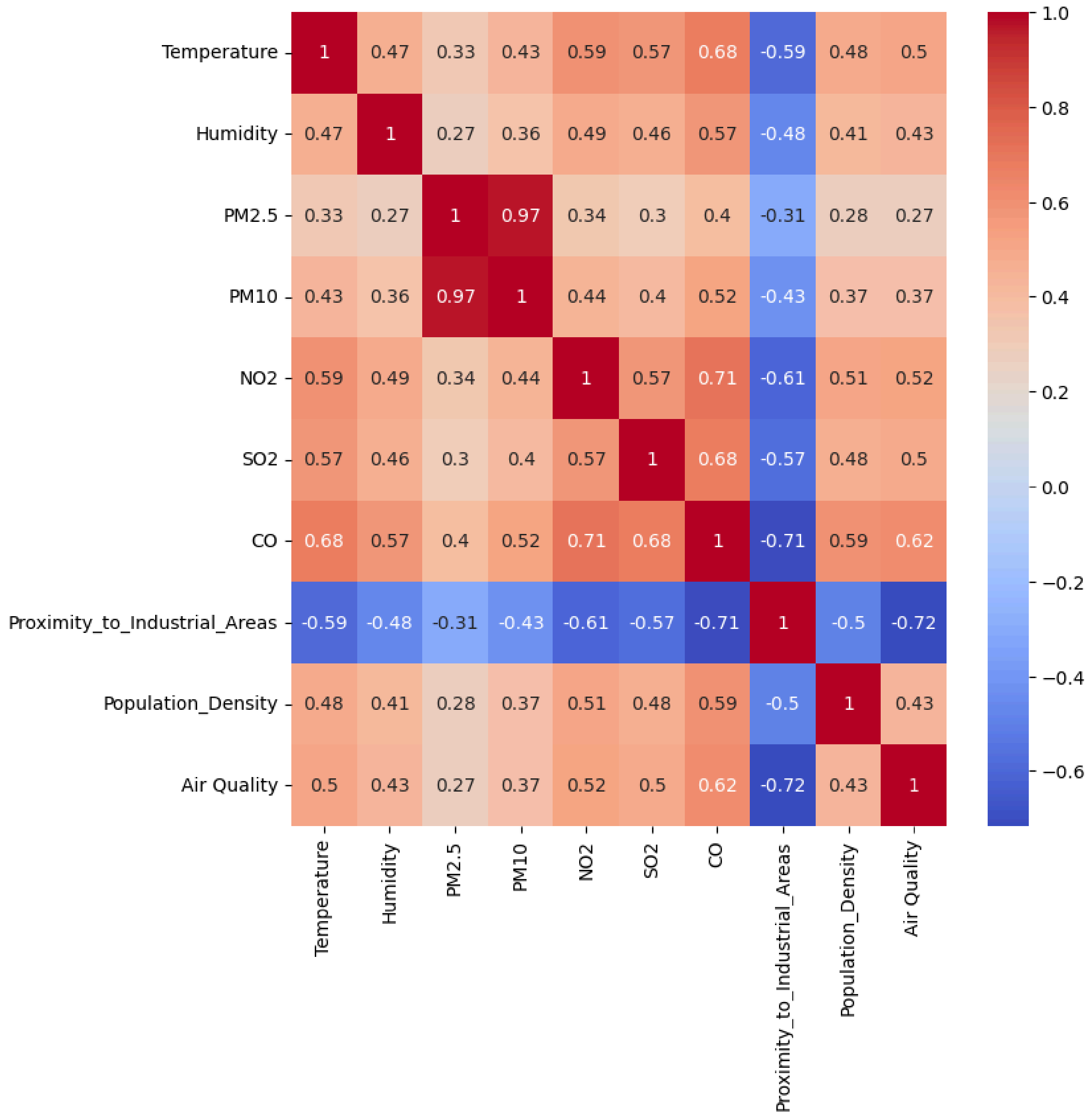
Source:

Mujtaba Mateen. (2024). Air Quality and Pollution Assessment [Data set]. Kaggle. https://doi.org/10.34740/KAGGLE/DS/6197184

Description:

- This dataset contains 9 input features and 1 output or target class.
- This is a classification problem, multi-class classification problem to be precise. Because, the air quality column is my target class. This column consists of categorical values. To be precise this is a multi-class classification problem as there are 4 unique classes Good, Moderate, Poor, and Hazardous.
- There are around 5000 data points.
- All the features are in quantitative and the target feature/class is in categorical.

Correlation using heatmap:

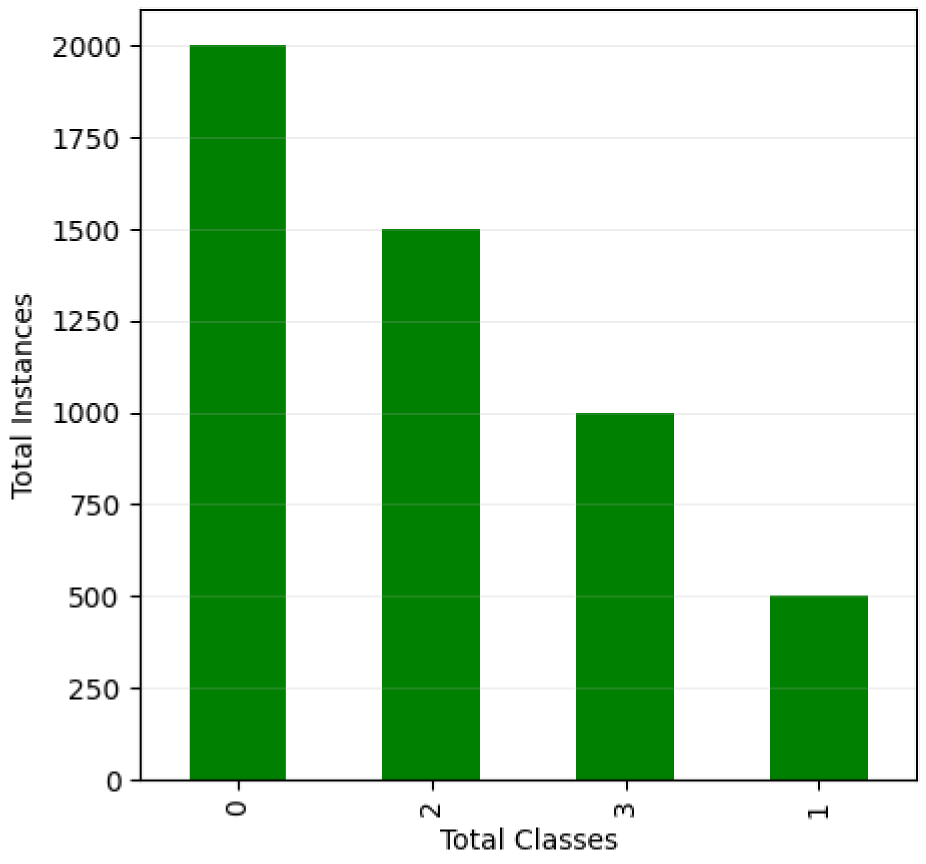


	Temperature	Humidity	PM2.5	PM10	NO2	SO2	CO	Proximity_to_Industrial_Areas	Population_Density	Air Quality
Temperature	1	0.47	0.33	0.43	0.59	0.57	0.68	-0.59	0.48	0.5
Humidity	0.47	1	0.27	0.36	0.49	0.46	0.57	-0.48	0.41	0.43
PM2.5	0.33	0.27	1	0.97	0.34	0.3	0.4	-0.31	0.28	0.27
PM10	0.43	0.36	0.97	1	0.44	0.4	0.52	-0.43	0.37	0.37
NO2	0.59	0.49	0.34	0.44	1	0.57	0.71	-0.61	0.51	0.52
SO2	0.57	0.46	0.3	0.4	0.57	1	0.68	-0.57	0.48	0.5
CO	0.68	0.57	0.4	0.52	0.71	0.68	1	-0.71	0.59	0.62
Proximity_to_Industrial_Areas	-0.59	-0.48	-0.31	-0.43	-0.61	-0.57	-0.71	1	-0.5	-0.72
Population_Density	0.48	0.41	0.28	0.37	0.51	0.48	0.59	-0.5	1	0.43
Air Quality	0.5	0.43	0.27	0.37	0.52	0.5	0.62	-0.72	0.43	1

Imbalanced Dataset:

From the bar chart below we can say that, in class: 1 there are very few instances compared to 0. That's why the dataset can be considered as an imbalanced dataset.

Target class bar-chart



Data pre-processing:

Categorical values: In my target class the values were in categorical order like "Good", "Poor", etc. ML algorithms can't work with categorical data. To solve it I imported LabelEncoder from Scikit-Learn library which helped me assign each unique categorical value to a unique numerical value. For example, all "good" is now 0 and "moderate" is 1.

Null values: Some rows had null values which could cause errors while training a ML model. To deal with this I used a method from Pandas library to drop the whole row that has missing values. My dataset was imbalanced from the beginning and the missing values were in random places. So dropping the rows didn't create any problems in my model.

Correlation: With correlation heatmap we can see PM2.5 and PM10 are highly correlated with the correlation value= 0.97. So this value is redundant. Dropping either one of these columns using Pandas library would be beneficial so, I dropped the PM10 column from the dataframe.

Feature Scaling (Normalization): In some ML algorithms, larger values can dominate smaller ones and it leads to biased results. To prevent it I used MinMaxScaler from the Scikit Learn library. MinMax scales a feature in the range of 0 and 1 by default.

<u>DataSet Splitting</u>:

My dataset is imbalanced so I have used "Stratified" splitting technique which helped me maintain fair representation of all the classes both in training and testing.

Then I defined test size 0.3 as instructed which splits the dataset and fixes 30% for testing and other for training. Also dropped the target class for 'x' so that only the feature columns stays for training and 'y' is target class column.

Model Training & Testing:

Decision Tree: Imported DecisionTreeClassifier from Scikit Learn Tree library. As my dataset is imbalanced, I passed an argument, class_weight=”balanced” so that the model adjusts the weight of each class and gives fair importance in each class not biased towards the majority class.

I chose this algorithm because it is suited for multi-class classification problems. It follows a tree approach which gives us clear visualization for decision making processes.

Logistic Regression: Imported LogisticRegression from Scikit Learn Library. Then same as the decision tree, I used clas_weight as the solution of class imbalance in my dataset.

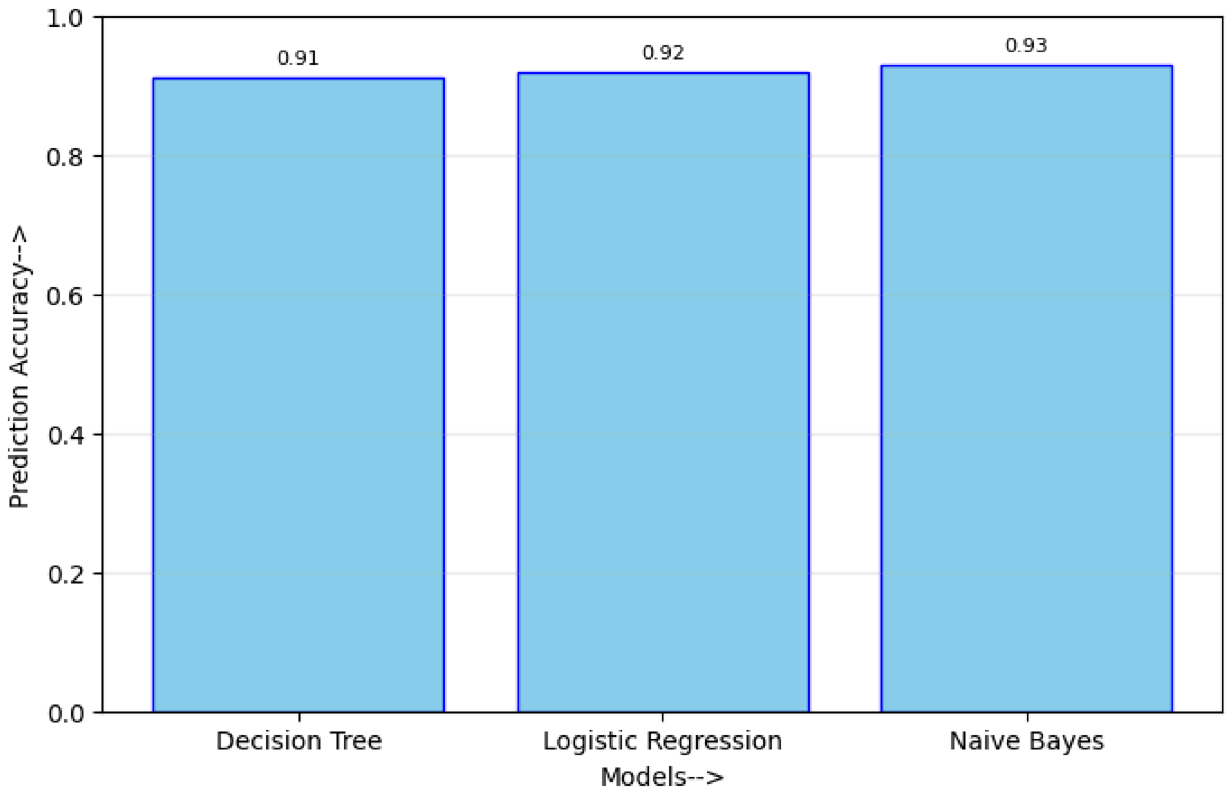
Reason behind choosing this is it can handle multi-class classification problems easily and stable for large datasets like with around 5000 instances. Moreover, it can deal with missing values, although I initially dealt with this problem.

Naive Bayes: In Naive Bayes I dealt with imbalance dataset differently. Giving a priority list as an argument helped this algorithm decide which class to give more priority than the other one.

Reason for choosing this algorithm is its efficiency. It can deal with large datasets very quickly.

<u>Comparison Analysis</u>:

Bar-chart on accuracy of all models:

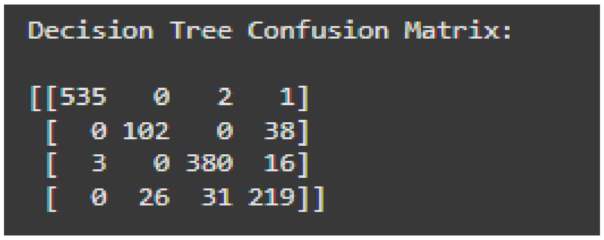


Precision & Recall comparison:

<u>Models</u>	***<u>Precision</u>***	***<u>Recall</u>***
Decision Tree	0.912281	0.913530
Logistic Regression	0.932210	0.923168
Naive Bayes	0.928440	0.926820

Confusion Matrices:

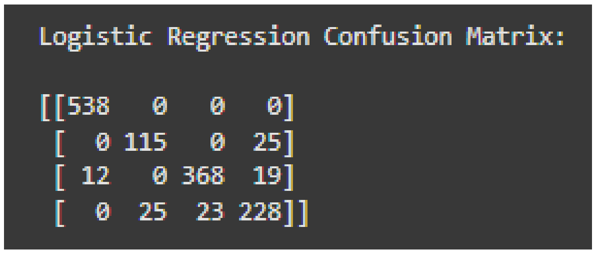
Decision Tree:



```
Decision Tree Confusion Matrix:

[[535   0   2   1]
 [  0 102   0  38]
 [  3   0 380  16]
 [  0  26  31 219]]
```

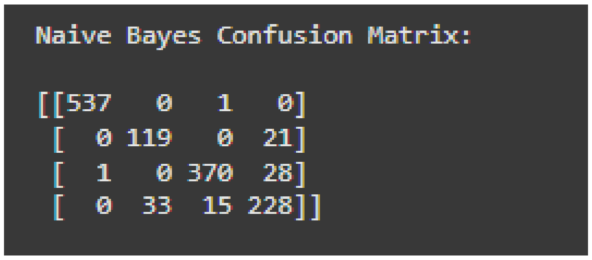
Logistic Regression:



```
Logistic Regression Confusion Matrix:

[[538   0   0   0]
 [  0 115   0  25]
 [ 12   0 368  19]
 [  0  25  23 228]]
```

Naive Bayes:



```
Naive Bayes Confusion Matrix:

[[537   0   1   0]
 [  0 119   0  21]
 [  1   0 370  28]
 [  0  33  15 228]]
```

<u>Conclusion</u>:

This project was designed to provide the solution to predict the Air Pollution. By adjusting and tweaking each model I tried to achieve an effective model as the solution for this problem. Through this effort, I have not only achieved our primary objective but also demonstrated the potential for innovative approaches to solve real-world challenges.