

LEBANESE
INTERNATIONAL
UNIVERSITY



الجامعة اللبنانية الدولية
الكلية اللبنانية للعلوم
الرياضية والهندسة

**School of Arts and Sciences
Computer Science Department**

CSCI634 – Air pollution prediction

Prepared by:
Ali Shreim 42130593

Presented to:
Dr. Ali Ballout

Spring 2024-2025

Contents

Introduction.....	3
Dataset	3
Methodology	4
Results.....	4
EDA.....	4
Some Visualization	5
Model evaluation	9
Random forest regression	9
Random Forest Classifier	9
Naïve Bayes.....	10
Neural network	10
Kmeans	11
Kmeans with PCA	11
Challenges	12
Flask and Interface.....	12

Introduction

The following report presents an in-depth exploration and predictive modelling using python. The main objective is to analyse a real-world dataset and to experiment with machine learning techniques, with the assessment of the results and their respective conclusions.

In this report we use many models like regression and classification in addition to neural network models in order to predict the results of our dataset target. Also we use clustering method to cluster the similar data together. Also these models are saved and deployed using flask to create an API's which can be used with a GUI to use the models in real world applications.

The main idea of this project is to identify the cities pollution and predict the class of pollution in the city. And to do this the Global Air Pollution have been used for training and evaluating the models.

Dataset

The dataset chosen "Global Air Pollution" provides geographical information on global air pollution levels from various countries and their respective cities, focusing on pollutants that strongly affects public health and environmental quality with specific measurements and air quality index (AQI) for each pollutant.

Air pollution refers to the contamination of indoor or outdoor environments by chemical, physical, or biological agents that alter the natural characteristics of the atmosphere. Major pollutants of concern include particulate matter (PM_{2.5}), carbon monoxide (CO), ozone (O₃), nitrogen dioxide (NO₂), and sulfur dioxide (SO₂). These pollutants can cause respiratory and cardiovascular diseases and contribute to global toxicity and death rates.

The Air Quality Index (AQI) is a numerical scale used to describe how clean or polluted the air is in a specific area. It helps people understand the potential health effects of breathing the air outside, especially when pollution levels are high.

The AQI is calculated based on the concentration levels of several key pollutants, including:

- **Particulate Matter (PM_{2.5} and PM₁₀)** – very small particles in the air that can enter the lungs and bloodstream.
- **Nitrogen Dioxide (NO₂)** – mostly from vehicles and industrial activity.
- **Ozone (O₃)** – formed when sunlight reacts with pollutants like nitrogen oxides.
- **Carbon Monoxide (CO)** – a gas released from burning fuels like gas, oil, or wood.

Each of these pollutants is assigned its own AQI value. The overall AQI for a location is usually determined by the highest value among these pollutants, known as the dominant pollutant.

The AQI scale is divided into categories, each representing a different level of health concern:

AQI Range	Category	Health Implication
0–50	Good	Air quality is satisfactory, no risk.
51–100	Moderate	Acceptable, but may affect sensitive people.
101–150	Unhealthy for Sensitive Groups	People with asthma or heart disease may feel effects.
151–200	Unhealthy	Everyone may begin to experience health issues.
201–300	Very Unhealthy	Health alert: serious effects for all.
301+	Hazardous	Emergency conditions; entire population at risk.

Dataset Structure

The dataset is organized with the following columns:

- **Country:** Name of the country.
- **City:** Name of the city.
- **AQI Value:** The overall Air Quality Index (AQI) value for the city.
- **AQI Category:** The overall air quality category.
- **CO AQI Value:** AQI value specifically for Carbon Monoxide.
- **CO AQI Category:** AQI category for Carbon Monoxide.
- **Ozone AQI Value:** AQI value specifically for Ozone.
- **Ozone AQI Category:** AQI category for Ozone.
- **NO₂ AQI Value:** AQI value specifically for Nitrogen Dioxide.
- **NO₂ AQI Category:** AQI category for Nitrogen Dioxide.
- **PM2.5 AQI Value:** AQI value for particulate matter with a diameter of 2.5 micrometers or less.
- **PM2.5 AQI Category:** AQI category for PM2.5 levels.

Methodology

First in EDA the unneeded data are removed from the model and a general information about data will be taken to know which model to use according to the column type. And the correlation matrix will be calculated to get the columns that are related to each other. Also some visualization will be applied to data to get general knowledge about data distribution, number of nulls, etc. To do these tasks pandas library is used to import data and make some calculations (like getting the number of nulls and correlations of data). And the matplotlib is used to visualize the data.

After selecting the needed columns for each model (categorical for classification and numerical for regression) these data are applied for models like Random Forest Classifier and Random Forest regression also for Naïve Bayes (Categorical NB). All the data had been scaled to get the best result as we use for each column type the best scaler that deal with it. In the training of model, a hyper parameter tuning had taken place to get the best model.

KMEANS is the method used in clustering the data into 6 clusters (according to health categories). In neural network we use a network build from 16 nodes as input layer then 32 and 8 nodes in 2 separated layers and finally a 1 layer to get the regression output.

The evaluation is take place by using confusion report for classification and r2 score and MAE for classification models. All these models are saved to create a interface to use them with flask for real product deployment.

Results

EDA

Nulls in data

```

Country          427
City              1
AQI Value        0
AQI Category     0
CO AQI Value     0
CO AQI Category  0
Ozone AQI Value  0
Ozone AQI Category 0
NO2 AQI Value    0
NO2 AQI Category 0
PM2.5 AQI Value  0
PM2.5 AQI Category 0
dtype: int64

```

we have 427 null data in the feature country and 1 in city here the null data will not affect the model results since these 2 columns will not be a features in the model.

Data types of each column

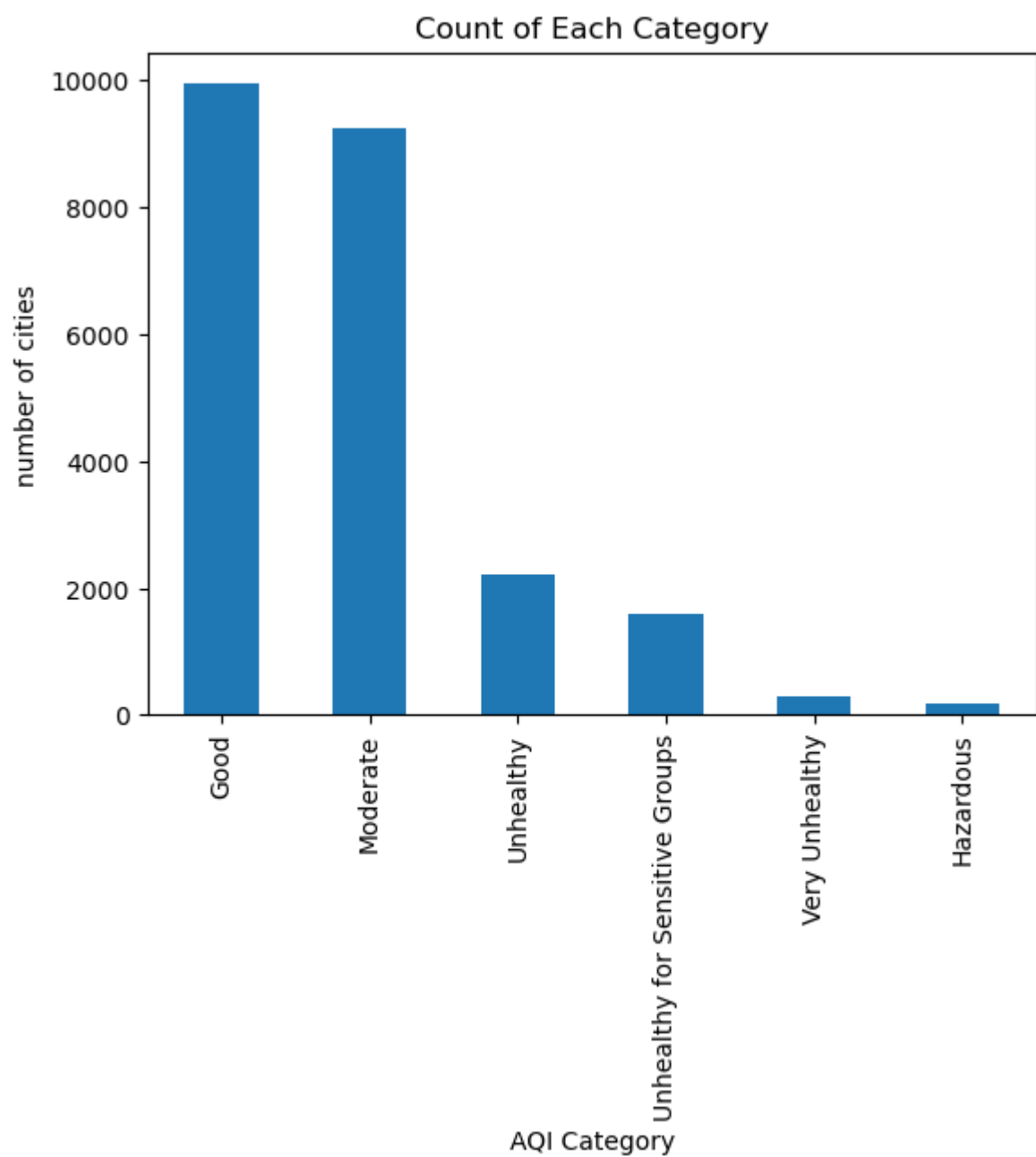
```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23463 entries, 0 to 23462
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Country                23036 non-null  object
1   City                   23462 non-null  object
2   AQI Value              23463 non-null  int64
3   AQI Category           23463 non-null  object
4   CO AQI Value           23463 non-null  int64
5   CO AQI Category        23463 non-null  object
6   Ozone AQI Value        23463 non-null  int64
7   Ozone AQI Category     23463 non-null  object
8   NO2 AQI Value          23463 non-null  int64
9   NO2 AQI Category       23463 non-null  object
10  PM2.5 AQI Value        23463 non-null  int64
11  PM2.5 AQI Category     23463 non-null  object
dtypes: int64(5), object(7)
memory usage: 2.1+ MB

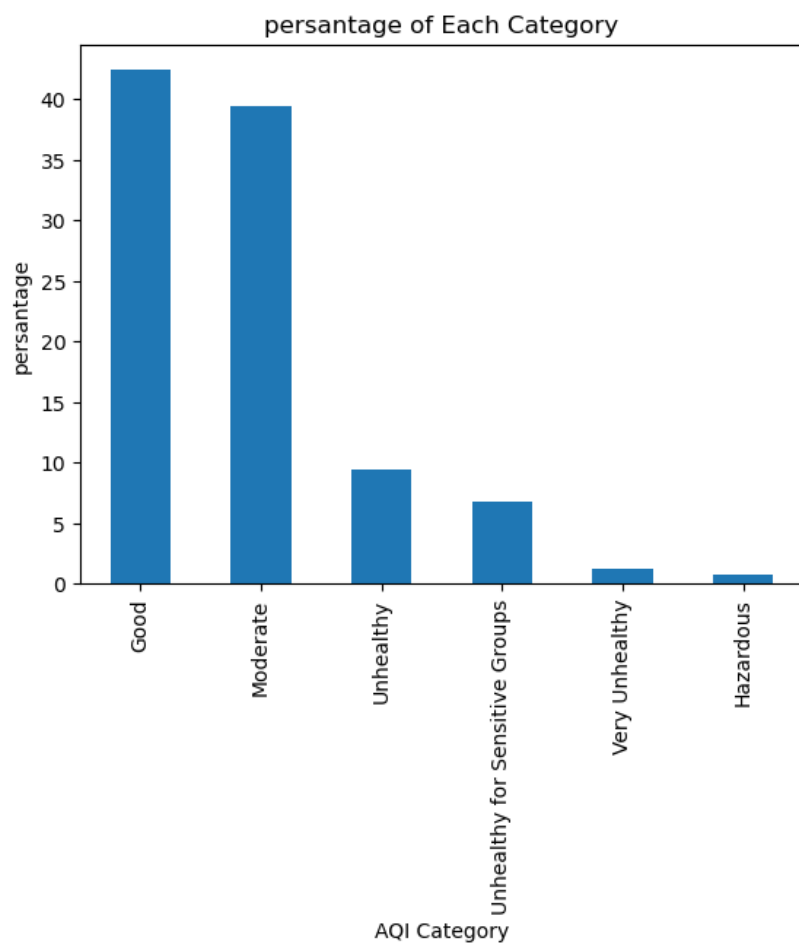
```

Some Visualization

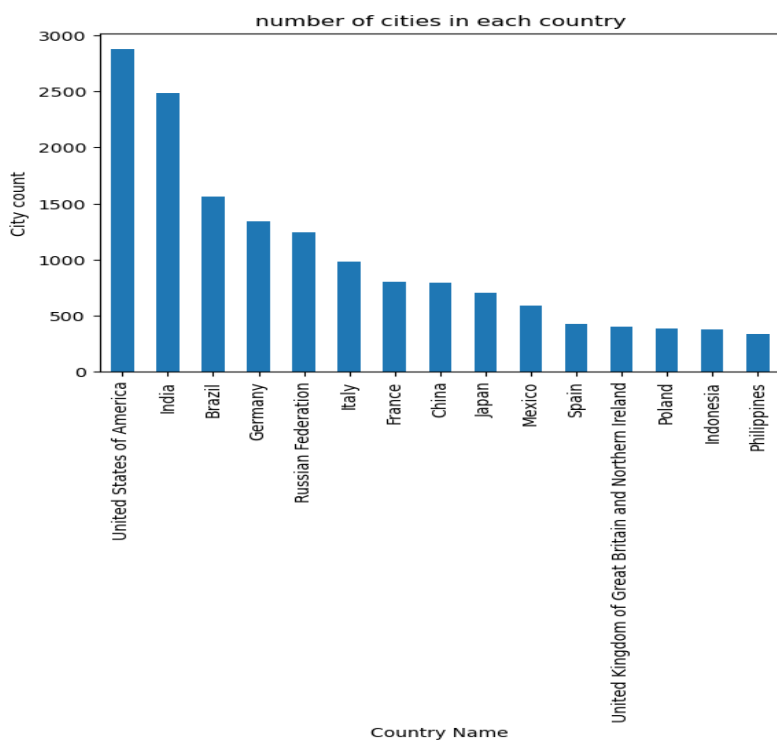
Get the number of cities in each AQI category



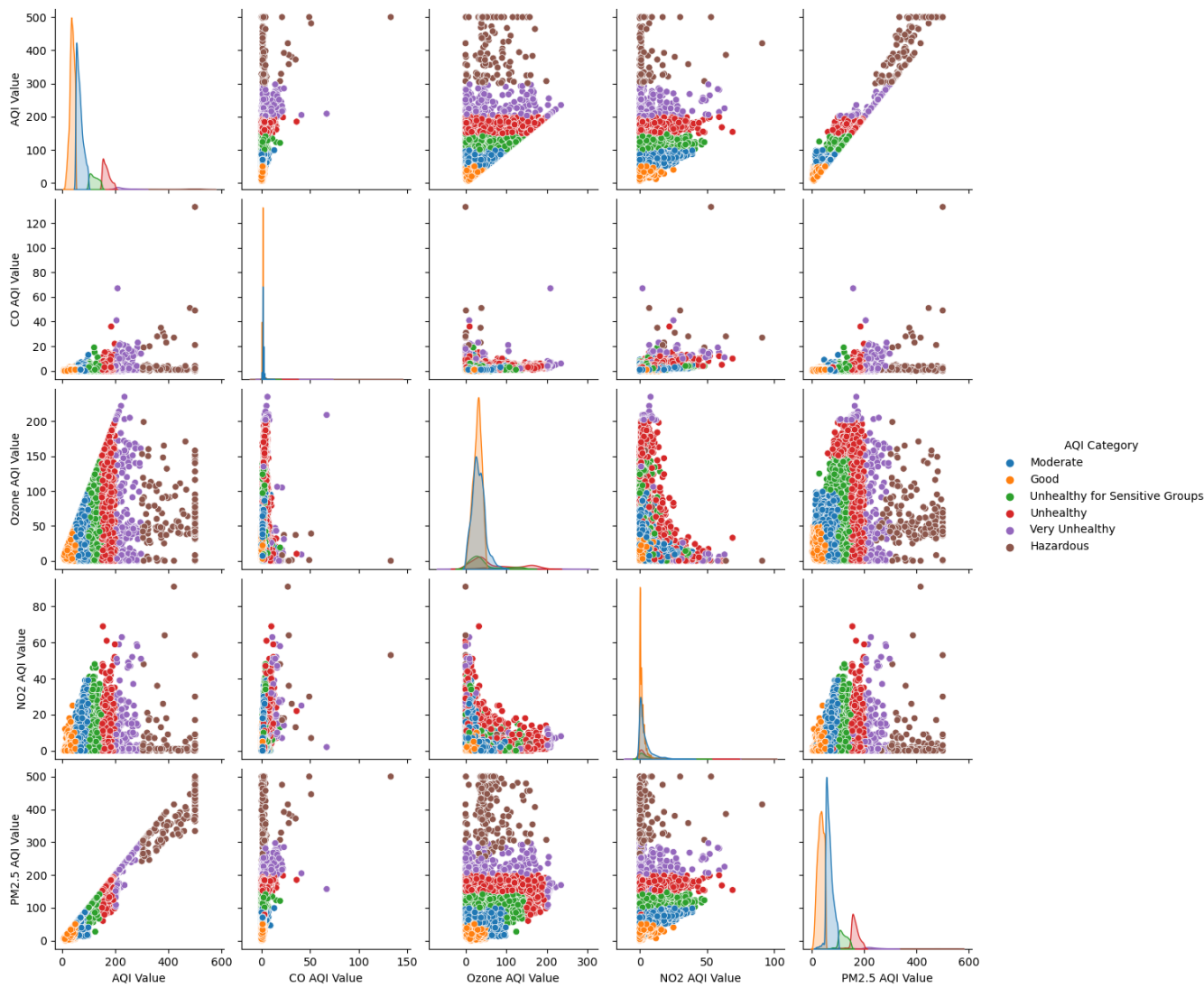
percentages of each category in the dataset



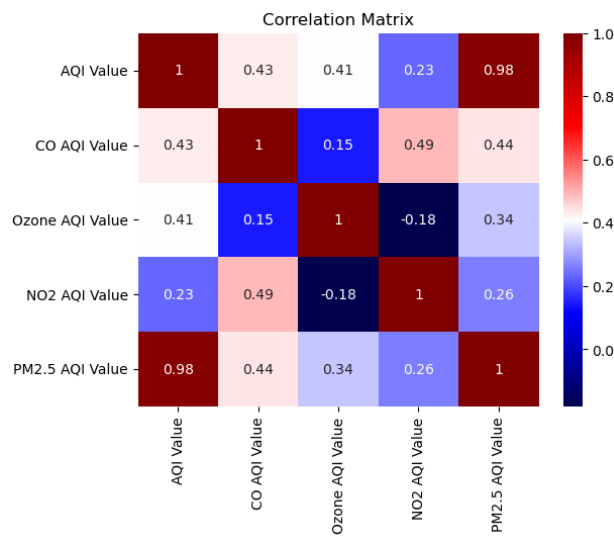
getting the higher 15 countries according to the number of cities



Distribution of Data



Correlation



Model evaluation

Random forest regression

First implementation for RFR is with number of estimator 3 and max depth 2. The r2 score is gives 0.88.

To get better result we use GridSearchCV with n_estimators 5,10,100 and max_depth None, 5, 7. And the best parameters are 100 for number of estimators and None for depth. This 2 parameters give r2 score 0.997.

Random Forest Classifier

First implementation for RFC is with number of estimator 3 and max depth 2. We get

[[2014 0 0 0 0 0] [0 36 0 0 0 6] [0 0 1837 0 0 0] [0 0 0 419 0 0] [0 0 1 0 320 0] [0 0 0 0 0 60]]									
						precision	recall	f1-score	support
Good						1.00	1.00	1.00	2014
Hazardous						1.00	0.86	0.92	42
Moderate						1.00	1.00	1.00	1837
Unhealthy						1.00	1.00	1.00	419
Unhealthy for Sensitive Groups						1.00	1.00	1.00	321
Very Unhealthy						0.91	1.00	0.95	60
accuracy								1.00	4693
macro avg						0.98	0.98	0.98	4693
weighted avg						1.00	1.00	1.00	4693

With hyper parameter tuning using GridSearchCv on 'n_estimators':[5,10,100], 'max_depth':[5,7] the best model is 'max_depth': 7, 'n_estimators': 5 we get

[[2014 0 0 0 0 0] [0 36 0 0 0 6] [0 0 1837 0 0 0] [0 0 0 419 0 0] [0 0 1 0 320 0] [0 0 0 0 0 60]]									
						precision	recall	f1-score	support
Good						1.00	1.00	1.00	2014
Hazardous						1.00	0.86	0.92	42
Moderate						1.00	1.00	1.00	1837
Unhealthy						1.00	1.00	1.00	419
Unhealthy for Sensitive Groups						1.00	1.00	1.00	321
Very Unhealthy						0.91	1.00	0.95	60
accuracy								1.00	4693
macro avg						0.98	0.98	0.98	4693
weighted avg						1.00	1.00	1.00	4693

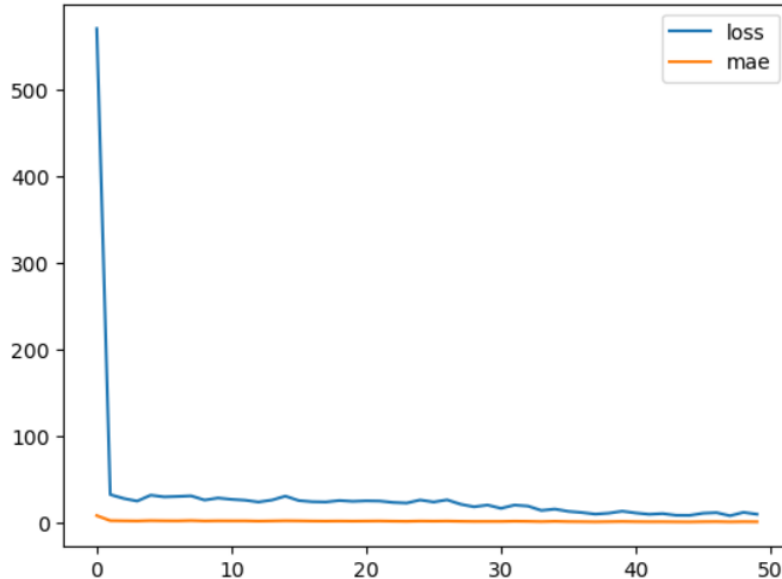
Naïve Bayes

In Naïve Bayes we use CategoricalNB and we get the following results

[[2014 0 0 0 0 0]									
[0 36 0 0 0 6]									
[0 0 1837 0 0 0]									
[0 0 6 413 0 0]									
[0 0 0 0 320 1]									
[0 0 0 0 3 57]]									
						precision	recall	f1-score	support
	Good					1.00	1.00	1.00	2014
	Hazardous					1.00	0.86	0.92	42
	Moderate					1.00	1.00	1.00	1837
	Unhealthy					1.00	0.99	0.99	419
Unhealthy for Sensitive Groups						0.99	1.00	0.99	321
Very Unhealthy						0.89	0.95	0.92	60
	accuracy							1.00	4693
	macro avg					0.98	0.96	0.97	4693
	weighted avg					1.00	1.00	1.00	4693

Neural network

The results of training are

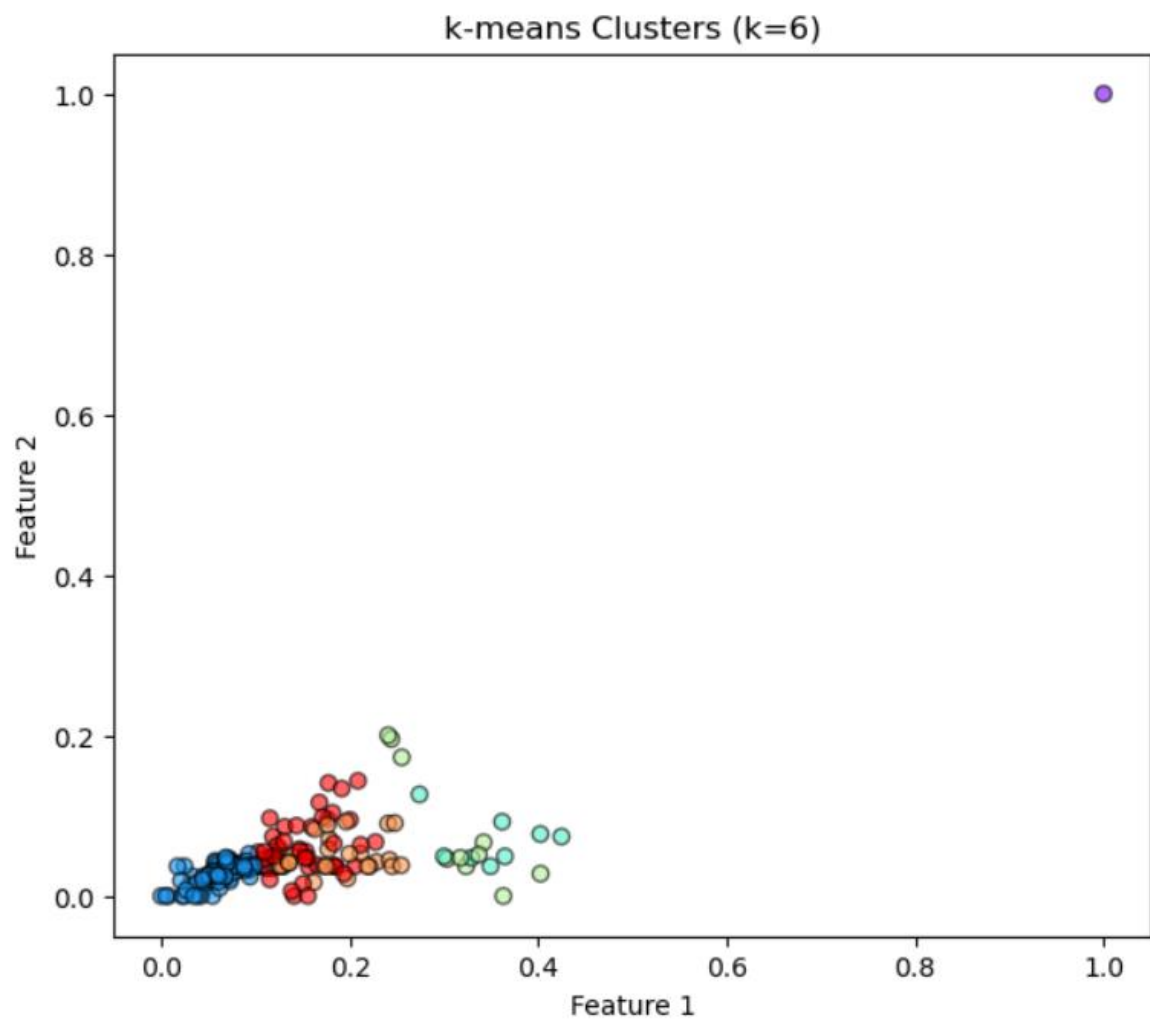


147/147 — 0s 906us/step - loss: 6.9293 - mae: 0.4506

WARNING:absl:You are saving your model as an HDF5 file via `model.save()` is considered legacy. We recommend using instead the native Keras format, `save_model(model, 'my_model.keras')`.

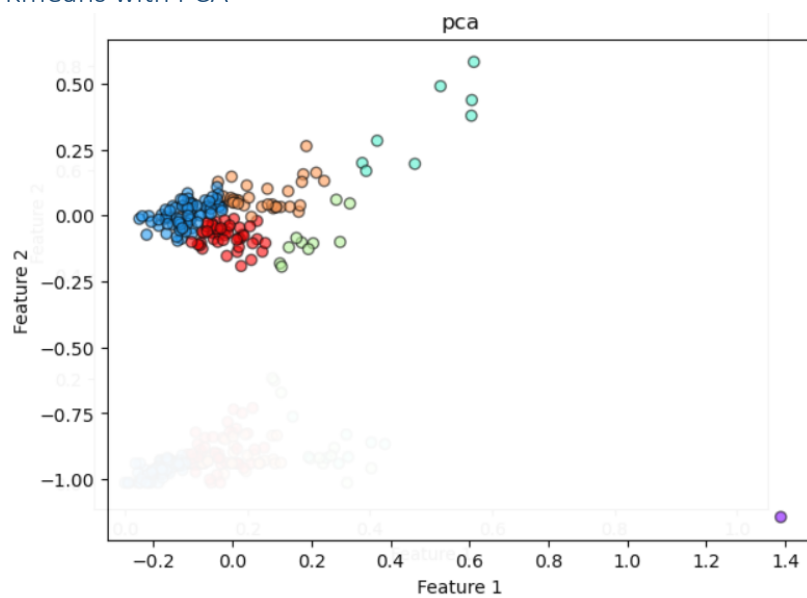
Test MAE: 0.43949902057647705

Kmeans



With silhouette score 0.366

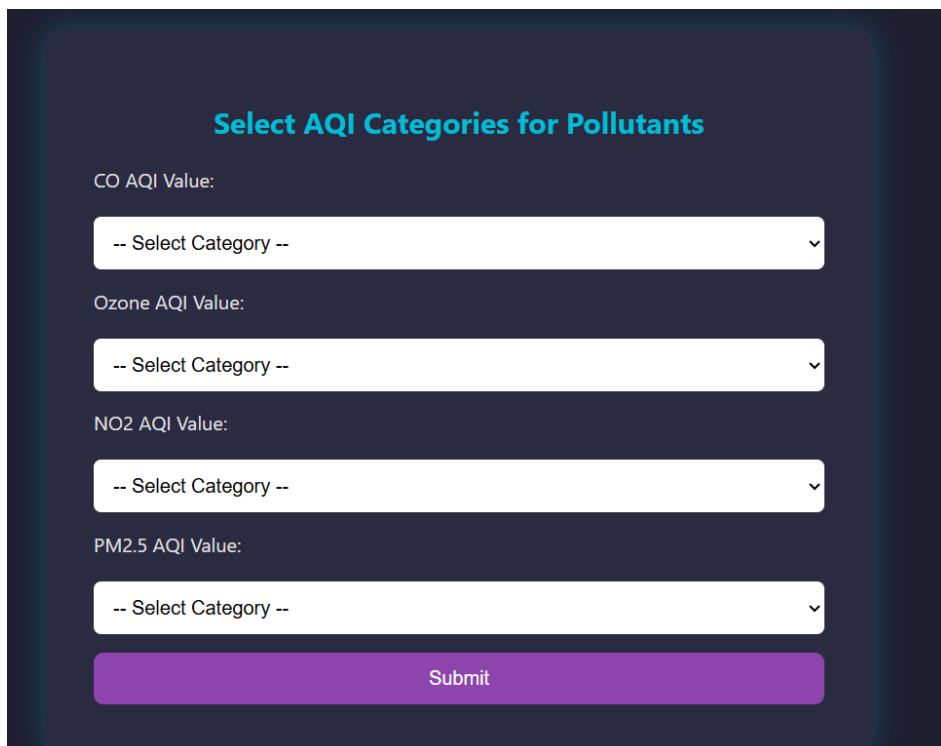
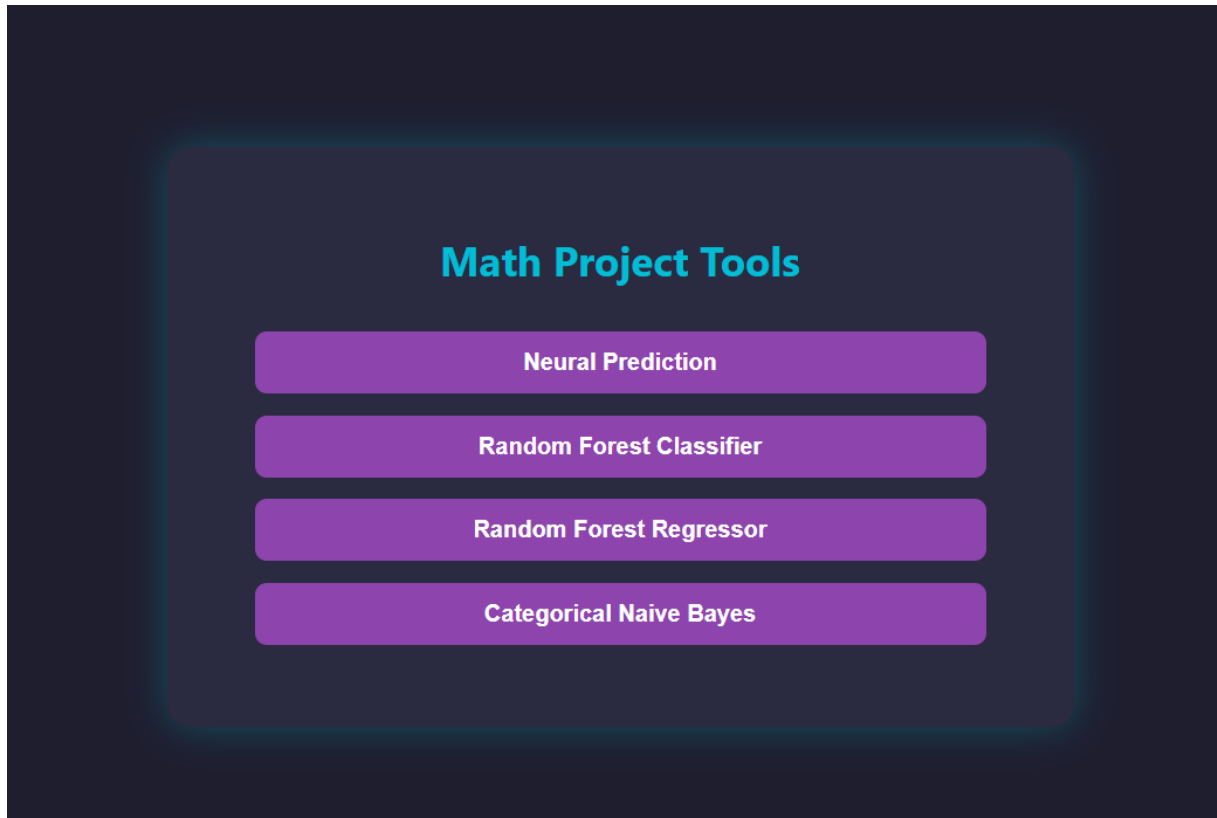
Kmeans with PCA



Challenges

The main challenge in this project is find a data as we see this data is not a good data for training do to the high correlation between target and PM2 value but in this project, we show all the steps of building the models and improve the accuracy.

Flask and Interface

A screenshot of a web application form titled "Select AQI Categories for Pollutants" in a teal font. The form contains four input sections, each with a label and a dropdown menu: "CO AQI Value:", "Ozone AQI Value:", "NO2 AQI Value:", and "PM2.5 AQI Value:". Each dropdown menu is white with the text "-- Select Category --" and a small downward arrow. At the bottom of the form is a purple "Submit" button. The form is set against a dark blue background with a subtle grid pattern.

Neural Prediction

CO AQI Value:

Ozone AQI Value:

NO2 AQI Value:

PM2.5 AQI Value:

Predict

Predicted Result: {"prediction":["Moderate"]}

The backend is built with flask

```
from flask import Flask, request, jsonify
from tensorflow.keras.models import load_model
import numpy as np
import joblib

neural_n = load_model("neural.h5", compile=False)
random_forest_reg = joblib.load('rfr_model_pipeline.pkl')
random_forest_class = joblib.load('rfc_model_pipeline.pkl')
categorical_NB_model = joblib.load('cnb_model_pipeline.pkl')

app = Flask(__name__)

@app.route('/neural', methods=['POST'])
def neural():
    data = request.get_json()
    try:
        features = np.array(data['features'], dtype=float).reshape(1, -1)
        scaler = joblib.load('nural_scaler.pkl')
        features = scaler.transform(features)
        print(features)
        prediction = neural_n.predict(features)
        return jsonify({'prediction': prediction.tolist()})
    except Exception as e:
        print("Error during prediction:", str(e))
        return jsonify({'error': str(e)}), 500
```

Here we load the models using joblib and keras and initialize the flask app. And every model is processed via the route and function name call. The above and below images show the implementation of these functions.

```
@app.route('/randomForest', methods=['POST'])
def randomForest():

    data = request.get_json()
    try:
        features = np.array(data['features'], dtype=float).reshape(1, -1)

        print(features)
        prediction = random_forest_reg.predict(features)
        return jsonify({'prediction': prediction.tolist()})
    except Exception as e:
        print("Error during prediction:", str(e))
        return jsonify({'error': str(e)}), 500
```

```
@app.route('/randomForestClassifier', methods=['POST'])
def randomForestClassifier():

    data = request.get_json()
    try:
        features = np.array(data['features']).reshape(1, -1)
        print(features)
        prediction = random_forest_class.predict(features)
        return jsonify({'prediction': prediction.tolist()})
    except Exception as e:
        print("Error during prediction:", str(e))
        return jsonify({'error': str(e)}), 500
```

```
@app.route('/categorical_NB', methods=['POST'])
def categorical_NB():
    data = request.get_json()
    try:
        features = np.array(data['features']).reshape(1, -1)
        prediction = categorical_NB_model.predict(features)
        return jsonify({'prediction': prediction.tolist()})
    except Exception as e:
        print("Error during prediction:", str(e))
        return jsonify({'error': str(e)}), 500
```

Conclusion

This project shows only some preprocessing techniques and how to apply hyper parameter tuning to the model and save then and finally deploy them using flask. But the results are not realistic and they are false because of the high correlation between PM2.5 and the AQI result.