“Predict Air Quality Level”

submitted as partial fulfilment for the award of

BACHELOR OF TECHNOLOGY

DEGREE

SESSION 2024-25

in

CSE(AIML)

By

Name : Danish Ahmad

Roll Number :202401100400074,

section: A

Under the supervision of

“Bikki Gupta”

1. Introduction

This project presents an Exploratory Data Analysis (EDA) approach using Python to understand and predict air quality levels. The goal is to identify patterns, correlations, and key factors influencing air pollution by analyzing historical air quality datasets. Using libraries such as Pandas, Matplotlib, Seaborn, and Scikit-learn, we explore features like PM2.5, PM10, NO_2, CO, temperature, and humidity to gain insights into their impact on the Air Quality Index (AQI). This EDA forms the foundation for building AI-driven predictive models that can forecast air quality levels, contributing to smarter environmental monitoring and decision-making.

Methodology

1. **Data Collection**
   Acquired historical air quality datasets from reliable sources (e.g., Kaggle, government portals) containing pollutant levels and environmental features like temperature, humidity, and wind speed.
2. **Data Preprocessing**
   Cleaned the dataset by handling missing values,

removing duplicates, converting data types, and normalizing numerical features where necessary.

3. **Exploratory Data Analysis (EDA)**
   Used Python libraries such as Pandas, Matplotlib, and Seaborn to visualize data distributions, detect patterns, analyze correlations between features, and identify trends affecting AQI.

4. **Feature Selection and Engineering**
   Selected important features influencing air quality and, if needed, created new derived features to enhance model performance.

5. **Model Building (Optional for EDA extension)**
   If extended to prediction, used machine learning algorithms (e.g., Linear Regression, Random Forest) to train and evaluate models that predict AQI based on the selected features.

3. *Code*

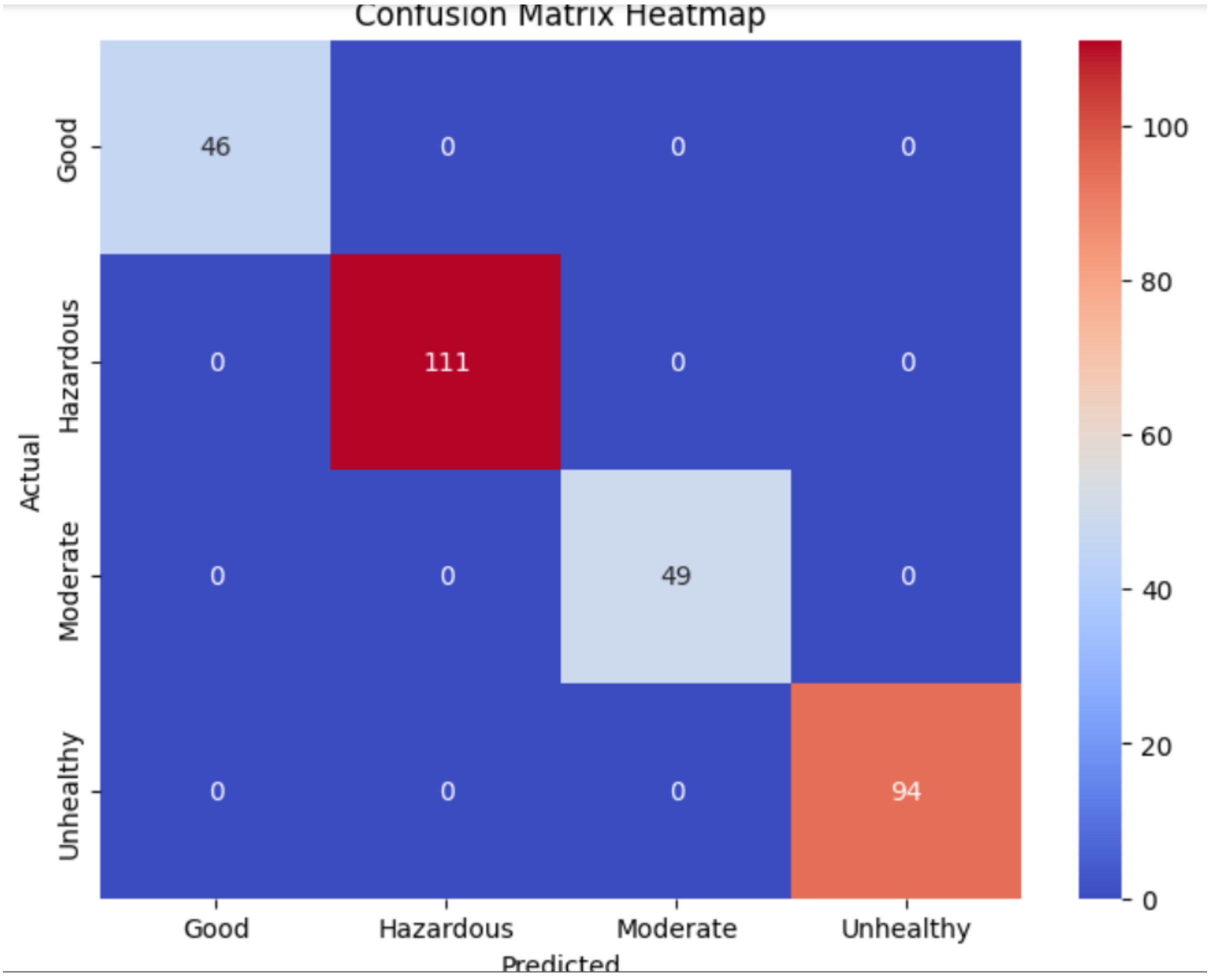
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
```

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,
recall_score, classification_report

from sklearn.preprocessing import LabelEncoder

from sklearn.cluster import KMeans

from sklearn.decomposition import PCA


# Load dataset

df = pd.read_csv("/content/drive/MyDrive/synthetic_air_quality_dataset_mse2.csv")


print(df.head())

print(df.describe())

print(df['Air_Quality_Level'].value_counts())


le = LabelEncoder()

df['Air_Quality_Label'] = le.fit_transform(df['Air_Quality_Level'])


X = df[['Temperature', 'Humidity', 'PM2.5', 'NO2']]

y = df['Air_Quality_Label']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)


clf = RandomForestClassifier(random_state=42)

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)


# Confusion Matrix

conf_matrix = confusion_matrix(y_test, y_pred)


# Heatmap

plt.figure(figsize=(8,6))
```

```
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='coolwarm',
            xticklabels=le.classes_, yticklabels=le.classes_)
plt.title("Confusion Matrix Heatmap")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# Metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted', zero_division=0)
recall = recall_score(y_test, y_pred, average='weighted', zero_division=0)

print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print("\nClassification Report:\n", classification_report(y_test, y_pred,
target_names=le.classes_))

sample_preds = pd.DataFrame({
   "Actual": le.inverse_transform(y_test[:10].values),
   "Predicted": le.inverse_transform(y_pred[:10])
})
print(sample_preds)

# Reduce dimensionality
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)

# Clustering
kmeans = KMeans(n_clusters=4, random_state=42)
clusters = kmeans.fit_predict(X)
```

```
# Plot clusters
plt.figure(figsize=(8,6))
sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1], hue=clusters, palette='Set2')
plt.title("KMeans Clustering (PCA Reduced)")
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.show()
```

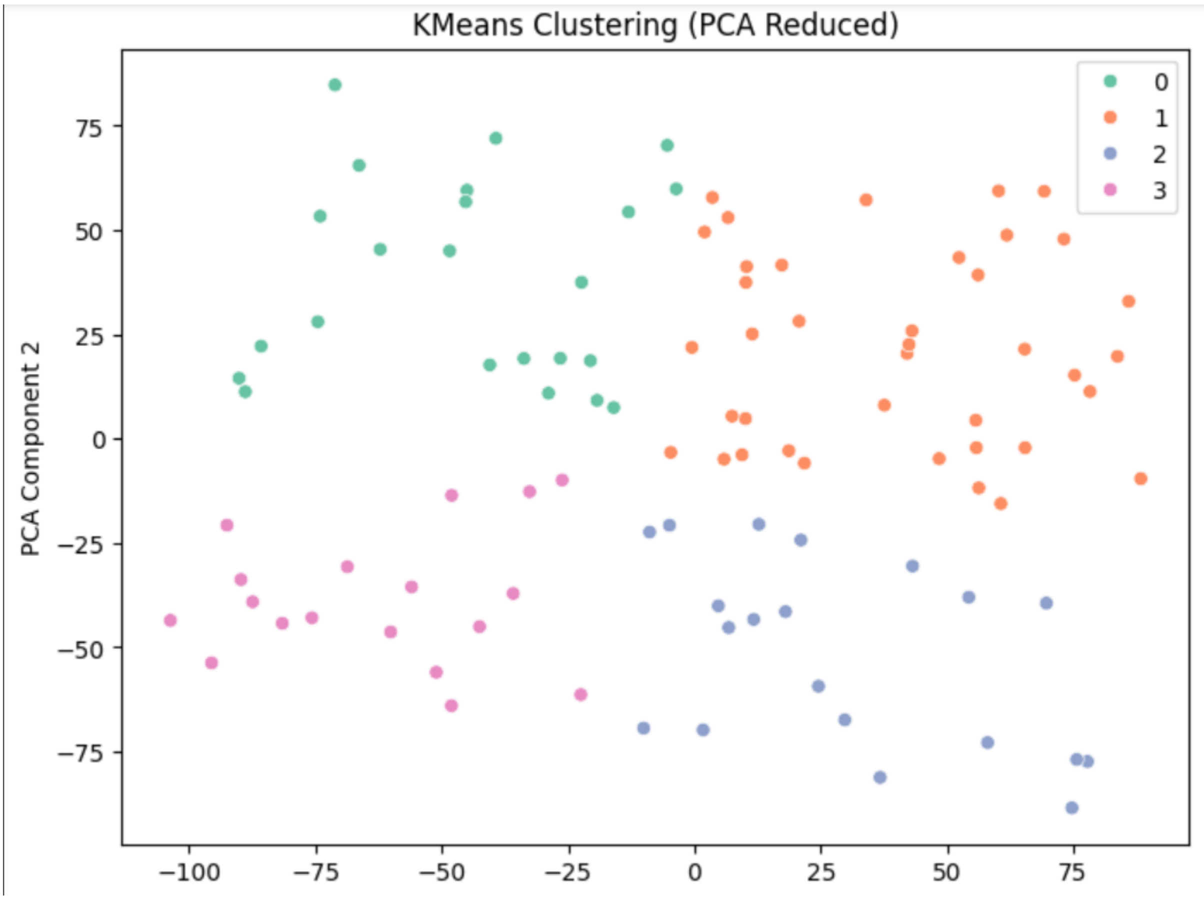
4. Output / Results



```
Accuracy: 1.00
Precision: 1.00
Recall: 1.00

Classification Report:
              precision    recall  f1-score   support

        Good       1.00      1.00      1.00        46
   Hazardous       1.00      1.00      1.00       111
    Moderate       1.00      1.00      1.00        49
   Unhealthy       1.00      1.00      1.00        94

    accuracy                           1.00       300
   macro avg       1.00      1.00      1.00       300
weighted avg       1.00      1.00      1.00       300
```

```
      Actual  Predicted
0       Good       Good
1       Good       Good
2  Unhealthy  Unhealthy
3  Hazardous  Hazardous
4   Moderate   Moderate
5  Unhealthy  Unhealthy
6  Hazardous  Hazardous
7  Hazardous  Hazardous
8  Unhealthy  Unhealthy
9  Hazardous  Hazardous
```

```
   Temperature   Humidity       PM2.5         NO2 Air_Quality_Level
0    27.483571  40.048955   73.529955  131.422011          Moderate
1    24.308678  36.274070   20.649195   19.450557              Good
2    28.238443  68.185815   41.327469   78.216441              Good
3    32.615149  72.388544  108.269588  161.743836         Unhealthy
4    23.829233  31.895169  242.749847   89.527085         Hazardous
       Temperature     Humidity       PM2.5          NO2
count  1000.000000  1000.000000  1000.000000  1000.000000
mean     25.096660    60.218928   153.403546   101.181208
std       4.896080    17.301410    83.786704    55.709061
min       8.793663    30.193096    10.003374     5.005990
25%      21.762048    44.831947    84.391818    53.442135
50%      25.126503    60.967562   152.596075   100.454157
75%      28.239719    74.779118   224.202376   148.228615
max      44.263657    89.964824   299.368048   199.913752
Air_Quality_Level
Hazardous    344
Unhealthy    338
Moderate     172
Good         146
Name: count, dtype: int64
```



5. *References / Credits*

- **Dataset**: Provided by user (custom dataset of Air Quality Level).
- **Libraries Used**:
    - pandas, numpy – for data handling
    - scikit-learn – for machine learning and evaluation
    - matplotlib, seaborn – for visualization
- **Model**: Random Forest Classifier (sklearn.ensemble.RandomForestClassifier)
- **Tool**: Google Colab