Assessment Report

on

“Predicting Air Quality Level"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE AI (B)

By Devansh Jadon

Roll_no.202401100300097

Under the supervision of

Mr. Shivansh Prasad

KIET Group of Institutions, Ghaziabad

Affiliated to

Dr. A.P.J. Abdul Kalam Technical University, Lucknow
(Formerly UPTU)

22 April, 2025

Introduction

Air quality is a critical environmental concern affecting public health and climate. By using historical data of pollutants like PM2.5 and NO2 along with environmental conditions like temperature, we can predict air quality levels and alert authorities or citizens in advance. This report demonstrates a machine learning approach to classify air quality levels using a Random Forest model.

Methodology

1. Data Upload: The user uploads a CSV dataset containing air quality indicators.

2. Label Encoding: The target variable 'quality_level' is encoded using LabelEncoder.

3. Feature Selection: We use 'pm25', 'no2', and 'temperature' as features.

4. Model Training: A Random Forest Classifier is used to train the model.

5. Evaluation: Metrics such as Accuracy, Precision, Recall, and a Confusion Matrix are generated.

6. User Input: The model also predicts air quality levels based on manual input of environmental parameters.

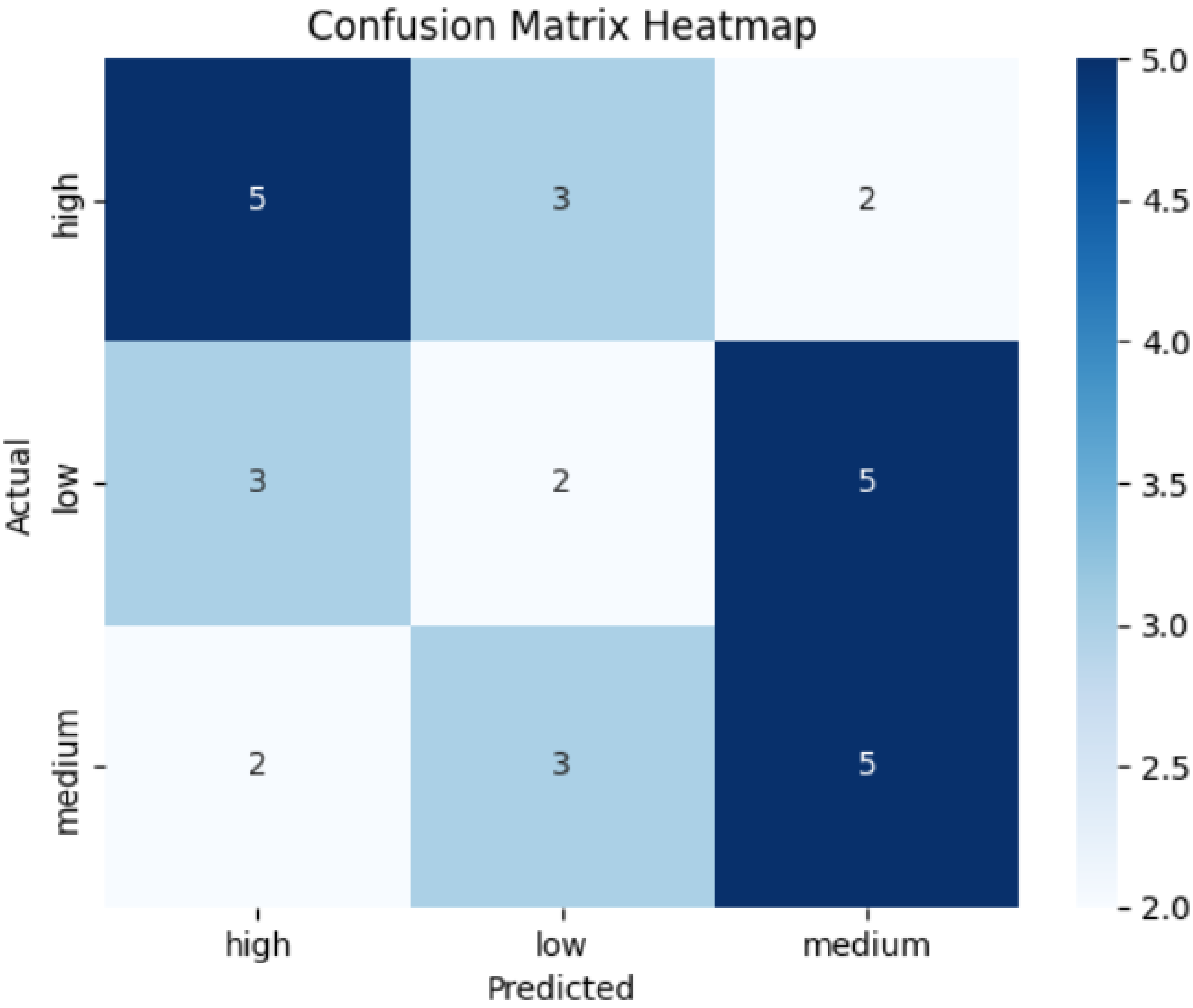
Code

```
# Install libraries if needed
# !pip install pandas scikit-learn seaborn matplotlib

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, precision_score, recall_score
import seaborn as sns
import matplotlib.pyplot as plt

# For Google Colab: Upload your CSV
from google.colab import files
uploaded = files.upload()

# Load the uploaded file
filename = list(uploaded.keys())[0]
df = pd.read_csv(filename)

# Step 1: Encode the target column (quality_level)
le = LabelEncoder()
df['quality_level'] = le.fit_transform(df['quality_level'])

# Show label mapping
label_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
print("\nLabel Mapping:", label_mapping)

# Step 2: Define features and labelA
X = df[['pm25', 'no2', 'temperature']]
y = df['quality_level']
```

```
# Step 3: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

# Step 4: Train the model
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Step 5: Predictions and Evaluation
y_pred = model.predict(X_test)

acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred, average='weighted')
rec = recall_score(y_test, y_pred, average='weighted')

print(f"\nAccuracy: {acc:.2f}")
print(f"Precision: {prec:.2f}")
print(f"Recall: {rec:.2f}\n")

print("Classification Report:\n")
print(classification_report(y_test, y_pred, target_names=le.classes_))

# Step 6: Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Heatmap')
plt.show()

# Step 7: Predict based on user input
```

```
print("\n--- Predict Air Quality Based on Your Input ---")
try:
    user_pm25 = float(input("Enter PM2.5 value: "))
    user_no2 = float(input("Enter NO2 value: "))
    user_temp = float(input("Enter Temperature value: "))

    user_input = [[user_pm25, user_no2, user_temp]]
    prediction = model.predict(user_input)
    predicted_label = le.inverse_transform(prediction)[0]

    print(f"\n🔍 Predicted Air Quality Level: **{predicted_label.upper()}**")
except Exception as e:
    print("Invalid input. Please enter numerical values only.")
```

Result



```
 air_quality.csv(text/csv) - 6086 bytes, last modified: 4/22/2025 - 100% done

Saving air_quality.csv to air_quality (6).csv


Label Mapping: {'high': np.int64(0), 'low': np.int64(1), 'medium': np.int64(2)}


Accuracy: 0.40
```

```
Precision: 0.39

Recall: 0.40


Classification Report:


                    precision    recall    f1-score   support


      high            0.50        0.50       0.50         10

      low            0.25         0.20        0.22         10

    medium           0.42         0.50       0.45         10

   accuracy                                  0.40         30

   macro avg         0.39         0.40       0.39         30

 weighted avg         0.39        0.40        0.39        30


--- Predict Air Quality Based on Your Input ---

Enter PM2.5 value: 120

Enter NO2 value: 5

Enter Temperature value: 69


🔍 Predicted Air Quality Level: **MEDIUM**
```