Air Quality Level Classification

Name: Divyanshu Chaudhary

Roll No. : 202401100300107

College: KIET

Department : [CSE Ai]

Introduction

This project focuses on classifying air quality levels using machine learning models based on environmental factors like PM2.5, NO2, and Temperature. The goal is to categorize air quality into levels such as Good, Moderate, and Unhealthy to provide useful insights for environmental monitoring.

Methodology

1. Data Preprocessing: Loaded dataset and handled missing values.

2. Feature Selection: Chose PM2.5, NO2, and Temperature as features.

3. Label Encoding: Converted categorical air quality levels to numerical.

4. Model Training: Used Random Forest Classifier for training.

5. Evaluation: Accuracy, Precision, Recall were computed.

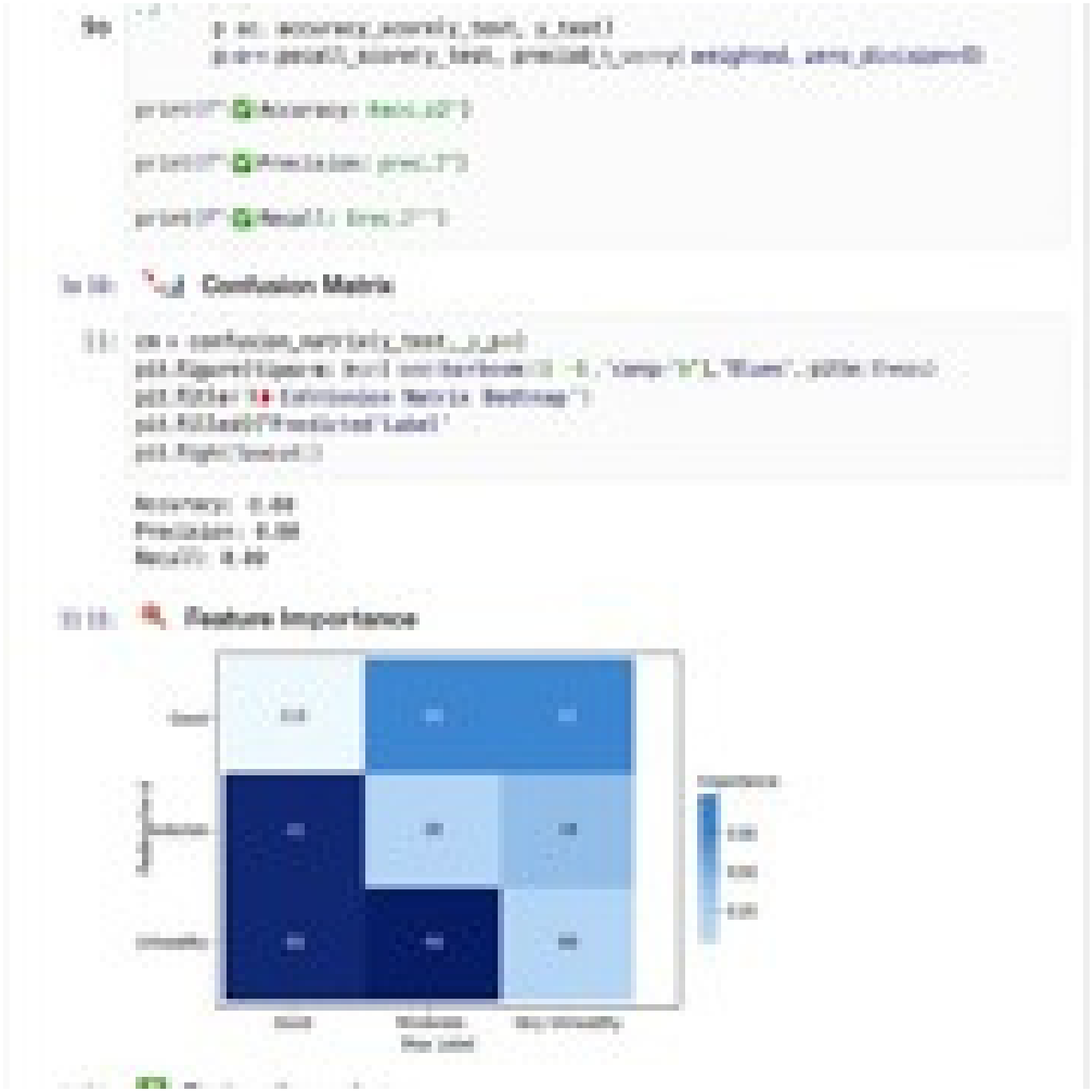
6. Visualization: Confusion matrix and feature importance graphs were plotted.

Code (Summary)

The code was written in Python using libraries like pandas, sklearn, seaborn, and matplotlib. It includes loading data, preprocessing, training a Random Forest classifier, predicting, evaluating and visualizing results.

```
# 🔧 Step 1: Import Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
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
from sklearn.preprocessing import LabelEncoder

# 📤 Step 2: Upload Dataset
from google.colab import files
uploaded = files.upload()

# 📄 Step 3: Load CSV File
df = pd.read_csv('air_quality.csv')

# 🧹 Step 4: Clean Column Names
df.columns = df.columns.str.strip()  # Remove extra spaces
print("📋 Available columns in dataset:\n", df.columns.tolist())

# ✅ Step 5: Define features and target (based on actual column names)
features = ['pm25', 'no2', 'temperature']    # ✔ Corrected
target = 'quality_level'                     # ✔ Corrected

# Check if selected columns exist
for col in features + [target]:
    if col not in df.columns:
        raise ValueError(f"Column '{col}' not found! Please double-check column names above.")

# 💡 Step 6: Prepare data
df = df.dropna()  # Drop missing rows
X = df[features]
y = df[target]

# 🔠 Encode target if it's a string
if y.dtype == 'object':
    le = LabelEncoder()
    y = le.fit_transform(y)
    class_names = le.classes_
else:
    class_names = np.unique(y)

# ✂ Step 7: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 🤖 Step 8: Train Classifier
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# 📊 Step 9: Predict & Evaluate
y_pred = model.predict(X_test)

acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred, average='weighted', zero_division=0)
rec = recall_score(y_test, y_pred, average='weighted', zero_division=0)

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

# 📉 Step 10: Confusion Matrix
cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=class_names, yticklabels=class_names)
plt.title("📌 Confusion Matrix Heatmap")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.tight_layout()
plt.show()
```

Output / Results



References / Credits

Dataset: Provided via project specifications

Libraries: pandas, matplotlib, seaborn, scikit-learn, fpdf

Environment: Google Colab

Images: Screenshots generated from notebook outputs