```
# -*- coding: utf-8 -*-
"""Untitled11.ipynb

Automatically generated by Colab.

Original file is located at

https://colab.research.google.com/drive/1A6gOz8GtfLYZB6AXweN7eIPkFgkrfa1D

data science
"""

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 classification_report, confusion_matrix

# Simulated data resembling real air quality dataset
data = {
    'PM2.5': np.random.uniform(5, 200, 1000),
    'NO2': np.random.uniform(10, 100, 1000),
    'CO': np.random.uniform(0.1, 10, 1000),
    'SO2': np.random.uniform(2, 50, 1000),
    'O3': np.random.uniform(10, 180, 1000),
}

df = pd.DataFrame(data)

# Define air quality level based on PM2.5 (simplified AQI categorization)
def categorize_air_quality(pm):
    if pm <= 50:
        return 'Good'
    elif pm <= 100:
        return 'Moderate'
    elif pm <= 150:
        return 'Unhealthy for Sensitive Groups'
    elif pm <= 200:
        return 'Unhealthy'
    else:
        return 'Very Unhealthy'

df['Air_Quality_Level'] = df['PM2.5'].apply(categorize_air_quality)

# Features and labels
X = df.drop('Air_Quality_Level', axis=1)
y = df['Air_Quality_Level']

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Model
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
```

```
# Predictions
y_pred = rf.predict(X_test)

# Evaluation
print("Classification Report:\n", classification_report(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))

# Feature Importance Plot
feature_importance = pd.Series(rf.feature_importances_,
index=X.columns).sort_values(ascending=False)
sns.barplot(x=feature_importance.values, y=feature_importance.index)
plt.title("Feature Importance for Air Quality Prediction")
plt.show()
```