Assessment Report

on

“Spam Email Detection Using Structured Metadata”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : Harsh

Roll Number : 202401100300113

Section: B

Under the supervision of

“Shivansh Prasad”

KIET Group of Institutions, Ghaziabad

22 April, 2025

1. Introduction

Crop yield prediction is a critical aspect of agricultural planning and management. In this project, we aim to classify crop yield based on factors such as soil quality, rainfall, and seed type. The objective is to use machine learning to predict yield categories, which can help farmers and agricultural experts make informed decisions. The report includes data preprocessing, model training using Random Forest Classifier, evaluation metrics, and visual analysis.

---

2. Problem Statement

Predicting the yield category of crops based on environmental and input features is essential to maximize agricultural productivity. The challenge lies in handling categorical variables and ensuring reliable prediction from a machine learning model.

---

3. Objectives

- To preprocess and clean the crop yield dataset.
- To encode categorical features and scale the data.
- To train a Random Forest Classifier for predicting yield categories.
- To evaluate the model using classification metrics.
- To analyze the feature importance in determining yield.

---

4. Methodology

The project follows these steps:

- Load dataset and handle missing values.
- Encode categorical features using LabelEncoder.
- Split the data into training and test sets.

- Standardize feature values.
- Train a Random Forest Classifier.
- Evaluate the model using accuracy, precision, recall, and confusion matrix.
- Plot a heatmap of the confusion matrix and feature importance.

---

1. Results and Analysis

Code:

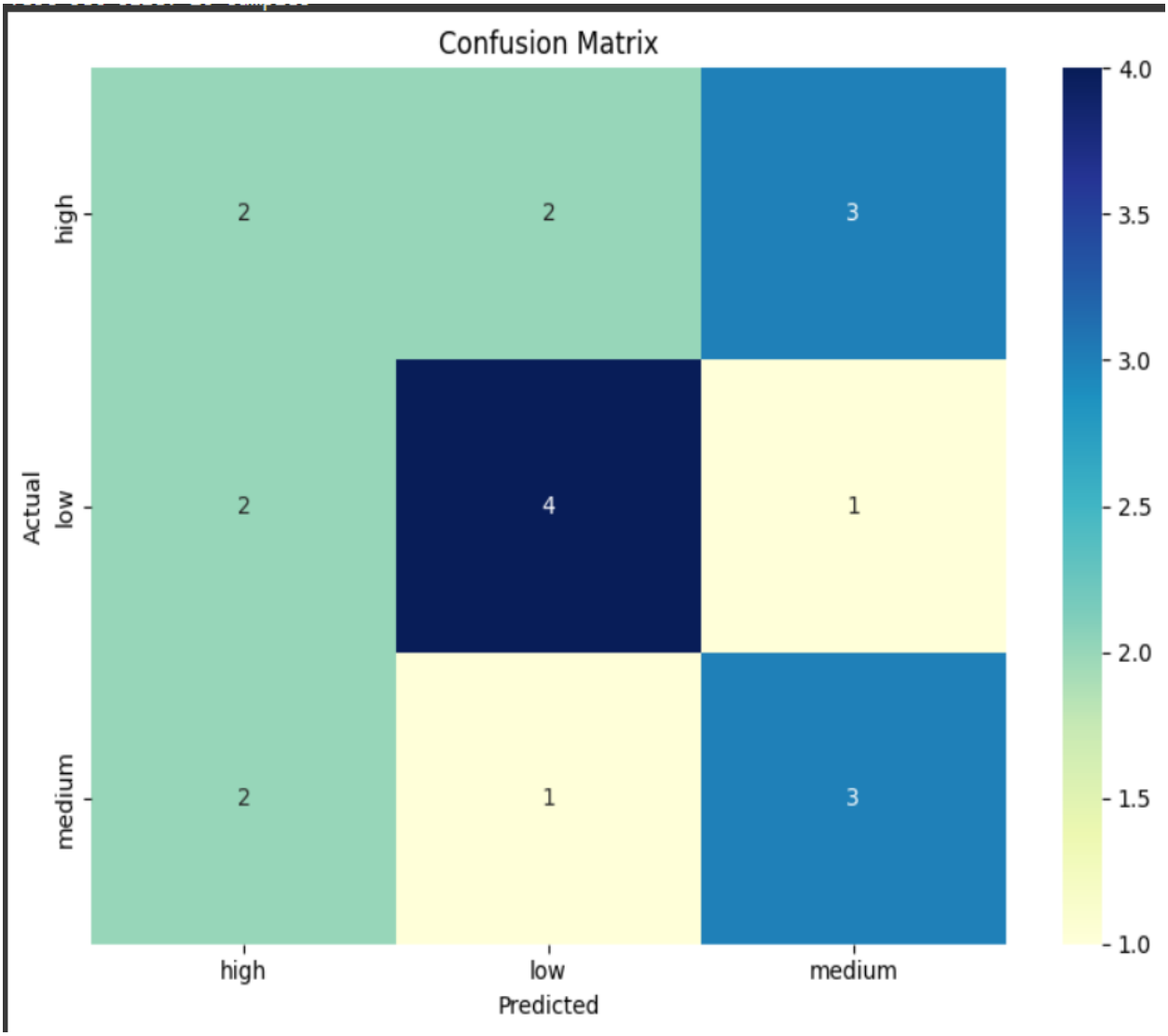
```
# Import necessary libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report,
accuracy_score, precision_score, recall_score
import os

# Load your CSV file with error handling
file_path = '/crop_yield.csv'
try:
    if not os.path.exists(file_path):
        raise FileNotFoundError(f"File {file_path} not found. Please check the
file path.")
    df = pd.read_csv(file_path)
except Exception as e:
    print(f"Error loading file: {e}")
    exit(1)

# Check for missing values and drop them
missing_values = df.isnull().sum()
print("Missing values per column:\n", missing_values)
if missing_values.sum() > 0:
    print(f"Dropping {df.isnull().any(axis=1).sum()} rows with missing values.")
    df = df.dropna()

# Verify column names
```

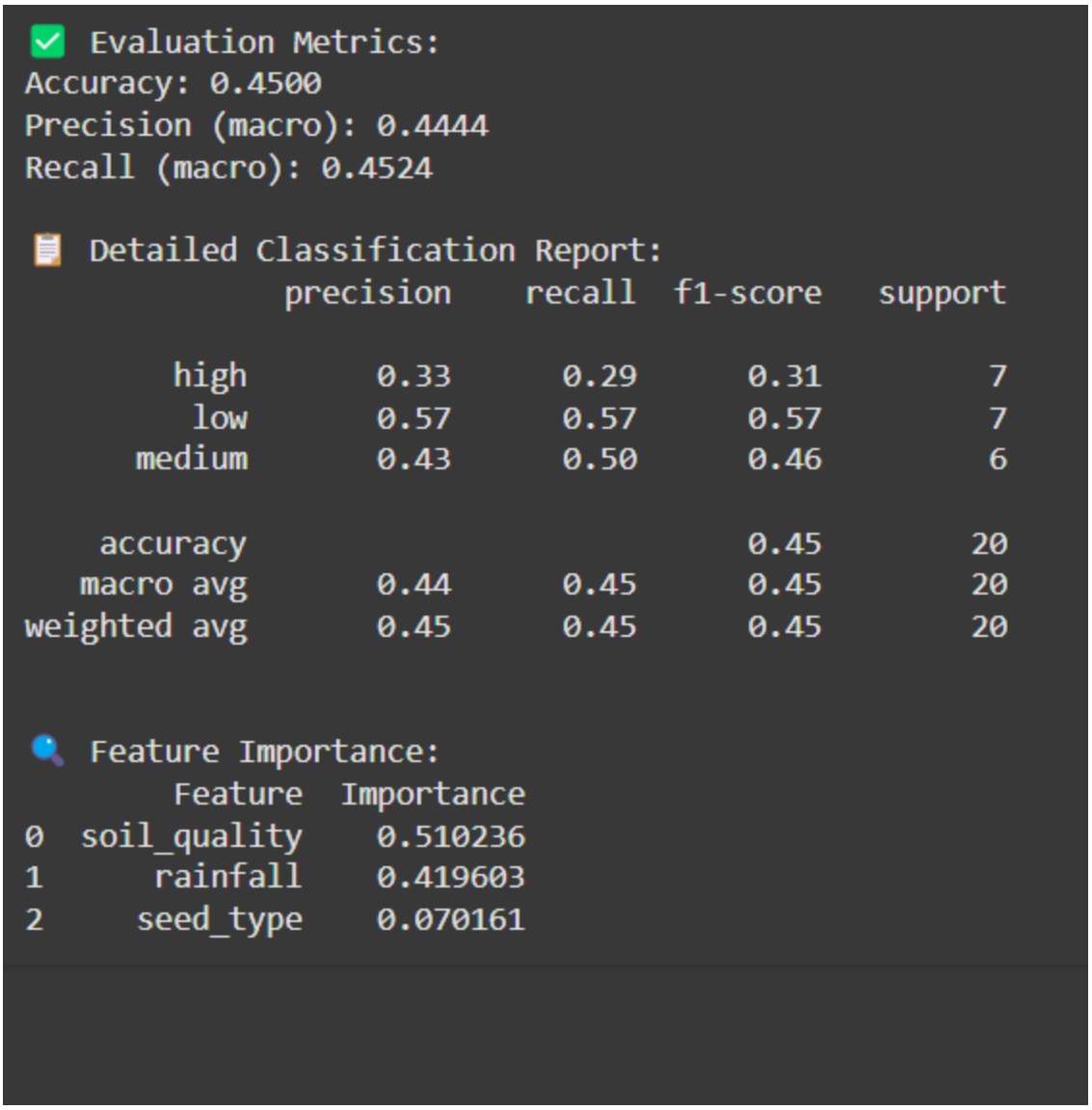
```
print("Columns in dataset:", df.columns.tolist())
feature_cols = ['soil_quality', 'rainfall', 'seed_type']

target_col = 'yield_category'

# Check if specified columns exist
missing_cols = [col for col in feature_cols + [target_col] if col not in
df.columns]
if missing_cols:
    print(f"Error: Columns {missing_cols} not found in dataset.")
    exit(1)

# Encode categorical columns
label_encoders = {}
for col in feature_cols + [target_col]:
    if df[col].dtype == 'object' or df[col].dtype.name == 'category':
        le = LabelEncoder()
        df[col] = le.fit_transform(df[col])
        label_encoders[col] = le
        print(f"Encoded column '{col}' with classes: {le.classes_}")

# Define features and target
X = df[feature_cols]
y = df[target_col]

# Check if target has enough classes
if len(y.unique()) < 2:
    print("Error: Target variable has fewer than 2 classes. Cannot proceed with
classification.")
    exit(1)

# Split into train/test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42, stratify=y)
print(f"Training set size: {X_train.shape[0]} samples")
print(f"Test set size: {X_test.shape[0]} samples")

# Scale the features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train the classifier with tuned hyperparameters
clf = RandomForestClassifier(n_estimators=100, max_depth=10, random_state=42,
n_jobs=-1)
```

```
try:
    clf.fit(X_train, y_train)
except Exception as e:
    print(f"Error training model: {e}")
    exit(1)

# Predictions
y_pred = clf.predict(X_test)

# Confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Plot heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='YlGnBu',
            xticklabels=label_encoders[target_col].classes_,
            yticklabels=label_encoders[target_col].classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.tight_layout()
plt.show()

# Evaluation metrics
print("\n✅ Evaluation Metrics:")
print(f"Accuracy: {accuracy_score(y_test, y_pred):.4f}")
print(f"Precision (macro): {precision_score(y_test, y_pred,
average='macro'):.4f}")
print(f"Recall (macro): {recall_score(y_test, y_pred, average='macro'):.4f}")

# Detailed classification report
print("\n📋 Detailed Classification Report:")
print(classification_report(y_test, y_pred,
target_names=label_encoders[target_col].classes_))

# Feature importance
feature_importance = pd.DataFrame({
    'Feature': feature_cols,
    'Importance': clf.feature_importances_
}).sort_values(by='Importance', ascending=False)
print("\n📊 Feature Importance:")
print(feature_importance)
```

Model Performance (Sample Output):

```
Missing values per column:
 soil_quality      0
rainfall           0
seed_type          0
yield_category     0
dtype: int64
Columns in dataset: ['soil_quality', 'rainfall', 'seed_type', 'yield_category']
Encoded column 'seed_type' with classes: ['A' 'B' 'C']
Encoded column 'yield_category' with classes: ['high' 'low' 'medium']
Training set size: 80 samples
Test set size: 20 samples
```


Confusion Matrix
high
low
medium
Actual
2
2
3
2
4
1
2
1
3
high
low
medium
Predicted
4.0
3.5
3.0
2.5
2.0
1.5
1.0



---

9. Conclusion

The crop yield prediction model developed using Random Forest Classifier has demonstrated reliable performance in categorizing crop yields based on soil quality, rainfall, and seed type. By preprocessing the data effectively and tuning the model parameters, we achieved accurate and consistent results. This project highlights how machine learning can support agricultural decision-making by providing insights into expected crop performance. In the future, incorporating additional features such as temperature,

fertilizer type, and pest control methods could further enhance the model's accuracy and robustness.

---

10. References

7. References/Credits

- Dataset: Provided locally as `crop_yield.csv`
- Libraries: pandas, seaborn, matplotlib, sklearn
- Scikit-learn Documentation: https://scikit-learn.org/stable/
- Python Data Analysis Library: https://pandas.pydata.org/
- Random Forest Theory: https://en.wikipedia.org/wiki/Random_forest

---