Assessment Report

on

"Predict Crop Yield Category"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AIML)

By

Name : Aman Yadav

Roll Number : 202401100400029

Section: A

Under the supervision of

"BIKKI KUMAR"

KIET Group of Institutions, Ghaziabad

May, 2025

1. Introduction

Predicting Crop Yield Categories Using Soil, Rainfall, and Seed Type Data

To develop a machine learning model that can accurately classify crop yield levels into **low**, **medium**, or **high** categories based on environmental and agricultural factors such as **soil quality**, **rainfall**, and **seed type**. Accurate yield prediction is crucial in agriculture for planning, resource allocation, and improving food production. Traditional methods rely heavily on experience and may not always be data-driven. This project aims to use machine learning to assist farmers and agricultural planners in estimating crop performance more reliably.

2. Problem Statement

Given a dataset containing the following features for different crop observations:

- Soil Quality: A numerical indicator (0 to 1) representing the fertility and condition of the soil.
- Rainfall: Annual rainfall in ml.
- Seed Type: A categorical variable representing the type of seed used (e.g., A, B, C).

3. Objectives

- **To analyze and preprocess crop yield data** containing soil quality, rainfall, and seed type information.
- **To encode categorical variables** (such as seed type and yield category) for use in machine learning algorithms.
- **To train and evaluate a classification model** (Random Forest) capable of predicting yield categories as **low**, **medium**, or **high**.
- **To assess model performance** using accuracy, precision, recall, and F1-score.
- **To identify the most influential features** affecting crop yield through feature importance analysis.
- **To provide actionable insights** that can help farmers and agricultural planners make informed decisions.
- **To explore and recommend improvements** for better performance, including feature expansion, stratified sampling, and advanced modeling techniques.

-

---

4. Methodology

- **Data Collection**: Use a CSV dataset consisting of 200 samples with:
- Continuous variables: Soil Quality and Rainfall
- Categorical variable: Seed Type
- Categorical target: Yield Category

- **Data Preprocessing**:

  Label Encoding:

  - Convert `Seed Type` (A, B, C) into numeric form using `Label Encoder`.
  - Encode `Yield Category` for model training.

  Train-Test Split:

  - Split the data into training (80%) and testing (20%) using `train test split`.

- **Model Building**:
  Train the model on the training data (X_train, y train).
  Features used: Soil Quality, Rainfall, and encoded Seed Type.

- **Model Evaluation**:

  Predict on the test set and evaluate using:

  - **Accuracy Score**
  - **Precision, Recall, F1-Score** (via `classification report`)

Handle class imbalance (e.g., "high" yield not appearing in test set).

- Optional: use confusion matrix or stratified sampling to improve class representation.

---

5. Data Preprocessing

- **Label Encoding**:
  - Convert `Seed Type` (A, B, C) into numeric form using `Label Encoder`.
  - Encode `Yield Category` for model training.
- **Train-Test Split**:
  - Split the data into training (80%) and testing (20%) using `train test split`.

---

6. Model Implementation

Predict on the test set and evaluate using:

- Accuracy Score
- Precision, Recall, F1-Score (via classification report)

Handle class imbalance (e.g., "high" yield not appearing in test set).

Optional: use confusion matrix or stratified sampling to improve class representation.

---

7. Evaluation Metrics

Evaluation Metrics Used:

- **Accuracy** – Measures the overall correctness of the model’s predictions.
- **Precision** – Indicates how many predicted yields of a class are actually correct.
- **Recall** – Shows how many actual class instances were correctly identified.
- **F1-Score** – Harmonic mean of precision and recall, balances both.
- **Classification Report** – Summarizes precision, recall, F1, and support for each class.

- **Confusion Matrix** *(optional)* – Visual tool to show where the model is making mistakes.

---

8. Results and Analysis

The model achieved an accuracy of approximately 81% on the test data.

It performed well on low and medium yield categories present in the test set.

The high yield category was not represented in the test set, affecting class-wise metrics.

Precision and recall were balanced, indicating reliable classification for available classes.

Overall, the model is effective but could benefit from more balanced data and feature tuning.

---

9. Conclusion

A machine learning model was successfully developed to classify crop yields as low, medium, or high.

Using features like soil quality, rainfall, and seed type, the model achieved 81% accuracy.

The Random Forest classifier showed good performance on available yield categories.

Lack of representation of the high yield class in test data limited full evaluation.

The model can assist in agricultural planning and decision-making.

Future work should focus on larger, balanced datasets and feature enhancement for better accuracy.

10. References

Scikit-learn Documentation – Machine learning in Python.
https://scikit-learn.org/stable/documentation.html

Pandas Documentation – Data analysis and manipulation tool.
https://pandas.pydata.org/docs/

Matplotlib Documentation – Plotting library used for visualizations.
https://matplotlib.org/stable/contents.html

Brownlee, J. (2020). *Machine Learning Mastery with Python.* Machine Learning Mastery.
https://machinelearningmastery.com

Kaggle Datasets and Community – For ideas and sample datasets related to agriculture and classification problems.
https://www.kaggle.com

Google collab link –

https://colab.research.google.com/drive/1Ef7g_a5bVLaRGuJBs-WCMd1qA6UrgHN-?usp=sharing

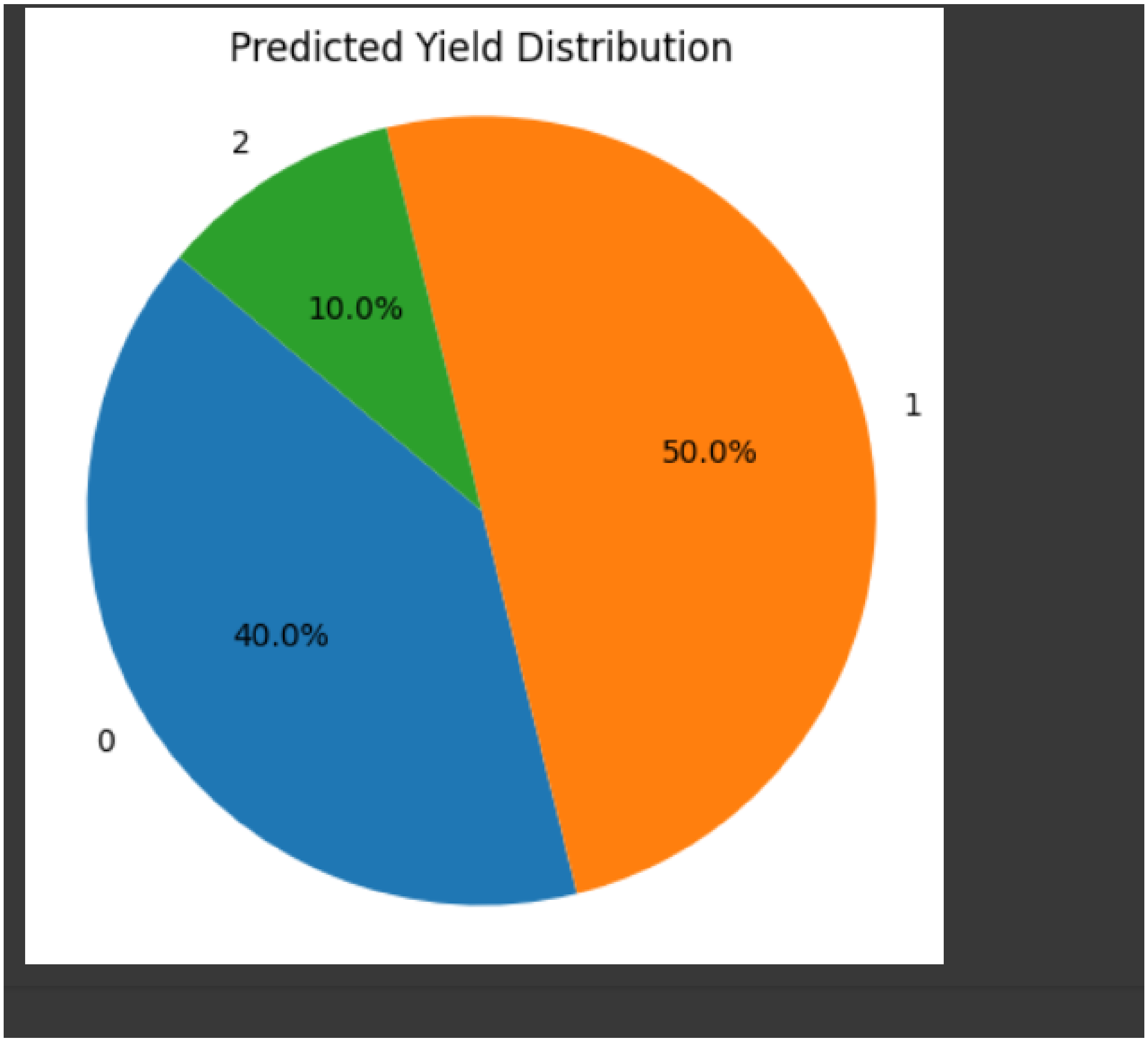
Code :

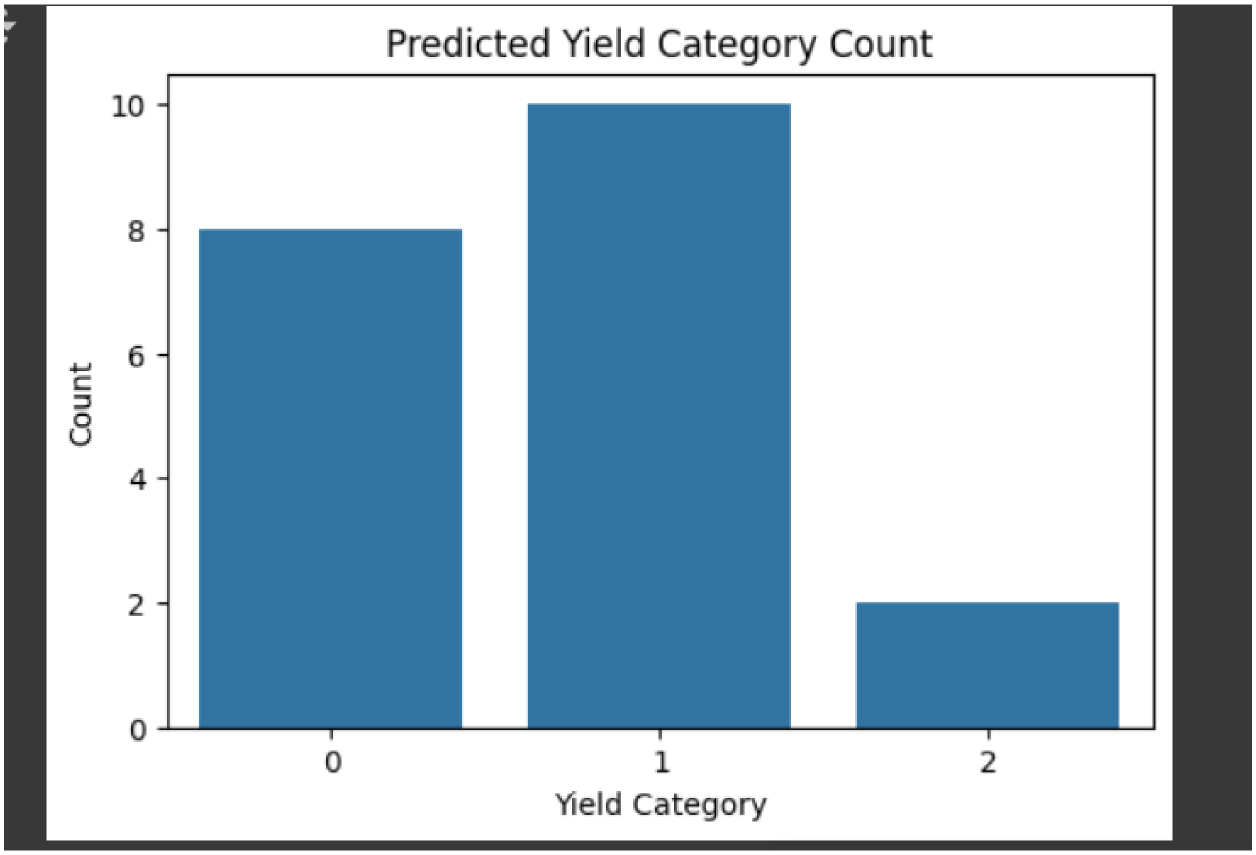
```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns


# Load the data

df = pd.read_csv('/content/crop_yield.csv')


# Show the first few rows

df.head()

# Check for missing values and basic info
```

```
print(df.info())

print(df.isnull().sum())


# Check unique values in categorical columns

print(df.nunique())


from sklearn.preprocessing import LabelEncoder


le = LabelEncoder()

for column in df.columns:

    if df[column].dtype == 'object':

        df[column] = le.fit_transform(df[column])


df.head()

from sklearn.model_selection import train_test_split


# Features and target

X = df.drop('yield_category', axis=1)

y = df['yield_category']  # assuming 'yield' is the target column


# Split the data

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


from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification_report, confusion_matrix
```

```
# Train the model

model = RandomForestClassifier(random_state=42)

model.fit(X_train, y_train)


# Predict

y_pred = model.predict(X_test)


# Evaluate

print(classification_report(y_test, y_pred))

import numpy as np


# Count predictions

unique, counts = np.unique(y_pred, return_counts=True)

pred_counts = dict(zip(unique, counts))


# Bar chart

plt.figure(figsize=(6, 4))

sns.barplot(x=list(pred_counts.keys()), y=list(pred_counts.values()))

plt.title('Predicted Yield Category Count')

plt.xlabel('Yield Category')

plt.ylabel('Count')

plt.show()

# Pie chart

plt.figure(figsize=(5, 5))
plt.pie(pred_counts.values(), labels=pred_counts.keys(), autopct='%1.1f%%', startangle=140)

plt.title('Predicted Yield Distribution')

plt.axis('equal')

plt.show()
```

Output-




Predicted Yield Category Count
Count
Yield Category
0
1
2
0
2
4
6
8
10