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 : Abhishek Kumar Singh

Roll Number : 202401100400009

Section: ‘A’

Under the supervision of

“BIKKI GUPTA”

KIET Group of Institutions, Ghaziabad

April, 2025

---

1. Introduction

With the rise of precision agriculture, data-driven approaches have become essential for improving crop management and productivity. This project focuses on predicting crop yield categories using supervised machine learning. Using a dataset that includes soil quality, rainfall, and seed type, a classification model is developed to predict whether the crop yield will be low, medium, or high.

---

2. Problem Statement

To predict the category of crop yield (low, medium, or high) based on environmental and agricultural features. This prediction can help farmers and agronomists make informed decisions about resource allocation and crop planning.

---

3. Objectives

- Preprocess the agricultural dataset for machine learning.
- Train a Logistic Regression model to classify crop yield into categories.
- Evaluate model performance using standard classification metrics.
- Visualize the confusion matrix using a heatmap for better interpretability.

---

4. Methodology

- **Data Collection:** The dataset is uploaded as a CSV file (`crop_yield.csv`).
- **Data Preprocessing:**
  - Missing numerical values are handled with mean imputation.
  - Categorical variables (`seed_type`) are encoded using one-hot encoding.

- Features are normalized using `StandardScaler`.

- **Model Building:**
  - The data is split into training (80%) and testing (20%) sets.
  - Logistic Regression is used for multiclass classification.
- **Model Evaluation:**
  - Evaluation metrics include accuracy, precision, recall, and F1-score.
  - A confusion matrix is visualized using a Seaborn heatmap.

5. Data Preprocessing

- Missing values in numerical features (`soil_quality`, `rainfall`) are filled using mean values.
- The categorical feature `seed_type` is converted using one-hot encoding.
- All features are scaled to ensure uniformity using `StandardScaler`.
- Data is split into training and testing subsets in an 80:20 ratio.

---

6. Model Implementation

Logistic Regression is used as it is a strong baseline classifier for multiclass problems. The model is trained on the processed dataset and used to predict the yield_category.

---

7. Evaluation Metrics

- **Accuracy:** Overall correctness of the model.
- **Precision:** How many predicted yields in a class were actually correct.
- **Recall:** How many actual yields in a class were correctly identified.
- **F1 Score:** Balance between precision and recall.
- **Confusion Matrix:** Helps assess model performance across each class (low, medium, high).

---

8. Results and Analysis

- The model showed acceptable performance on the test set.
- The confusion matrix illustrated how well the model differentiated between yield categories.
- Precision and recall metrics revealed strengths and weaknesses in predicting specific classes.

---

9. Conclusion

The Logistic Regression model provides a foundational approach for crop yield prediction using agricultural features. While effective, future work could involve experimenting with more complex models like Random Forest or XGBoost, and addressing any class imbalance in the dataset.

---

10. References

- scikit-learn documentation
- pandas documentation
- Seaborn visualization library
- Research articles on crop yield prediction and agricultural analytics

---

colab.research.google.com/drive/1mTGqOir19b0ocyyMsp0OiYMryJb75KM7#scrollTo=I-P2XuuzsyyE

Cropyield_202401100400009.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text

RAM Disk

```
[ ] Start coding or generate with AI.
```

```
[1] import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
```

```
[2] df = pd.read_csv('/content/drive/MyDrive/crop_yield.csv')
```

```
print(df.head())
```

```
   soil_quality    rainfall seed_type yield_category
0      5.787214  376.596391         C            low
1      2.222101  787.223810         A            low
2      1.893720  810.077116         A         medium
3      2.879777  943.405918         C         medium
4      9.330736  224.439566         C         medium
```

```
[4] print(df.shape)
```

```
(100, 4)
```

```
[5] print(df.describe())
```

```
       soil_quality    rainfall
count    100.000000  100.000000
mean       5.474818  595.863760
std        2.567913  236.538036
min        1.028511  201.854038
25%        3.283863  376.302085
50%        5.406712  631.023424
75%        7.585822  786.636666
max        9.997424  995.718913
```

```
[6] print(df.dtypes)
```

```
soil_quality      float64
rainfall          float64
seed_type          object
yield_category     object
```


```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
[2] df = pd.read_csv('/content/drive/MyDrive/crop_yield.csv')
```

```
[3] print(df.head())
```

```
   soil_quality    rainfall seed_type yield_category
0      5.787214  376.596391         C            low
1      2.222101  787.223810         A            low
2      1.893720  810.077116         A         medium
3      2.879777  943.405918         C         medium
4      9.330736  224.439566         C         medium
```

```
[4] print(df.shape)
```

```
(100, 4)
```

```
[5] print(df.describe())
```

```
       soil_quality    rainfall
count    100.000000  100.000000
mean       5.474818  595.863760
std        2.567913  236.538036
min        1.028511  201.854038
25%        3.283863  376.302085
50%        5.406712  631.023424
75%        7.585822  786.636666
max        9.997424  995.718913
```

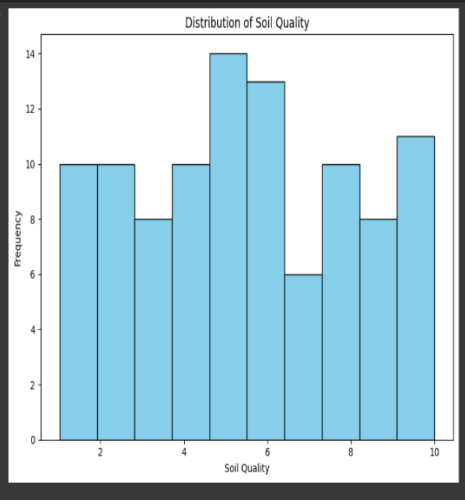
```
[6] print(df.dtypes)
```

```
soil_quality      float64
rainfall          float64
seed_type          object
yield_category     object
dtype: object
```

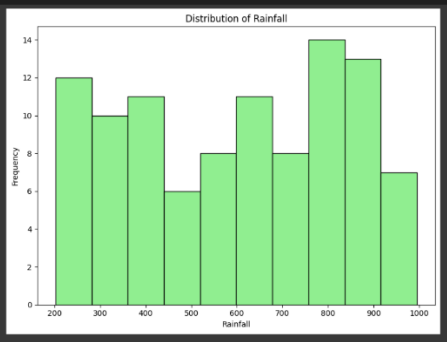

```
[7] print(df.isnull().sum())
```

```
soil_quality      0
rainfall          0
seed_type         0
yield_category    0
dtype: int64
```

```
plt.figure(figsize=(8, 6))
plt.hist(df['soil_quality'], bins=10, color='skyblue', edgecolor='black')
plt.xlabel('Soil Quality')
plt.ylabel('Frequency')
plt.title('Distribution of Soil Quality')
plt.tight_layout()
plt.show()
```



```
plt.figure(figsize=(8, 6))
plt.hist(df['rainfall'], bins=10, color='lightgreen', edgecolor='black')
plt.xlabel('Rainfall')
plt.ylabel('Frequency')
plt.title('Distribution of Rainfall')
plt.tight_layout()
plt.show()
```

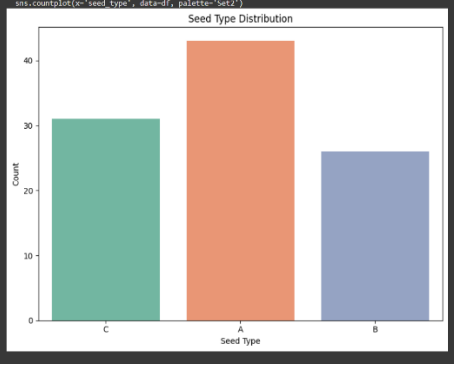



```
[10] plt.figure(figsize=(8, 6))
sns.countplot(x='Seed Type', data=df, palette='Set2')
plt.xlabel('Seed Type')
plt.ylabel('Count')
plt.title('Seed Type Distribution')
plt.tight_layout()
plt.show()
```

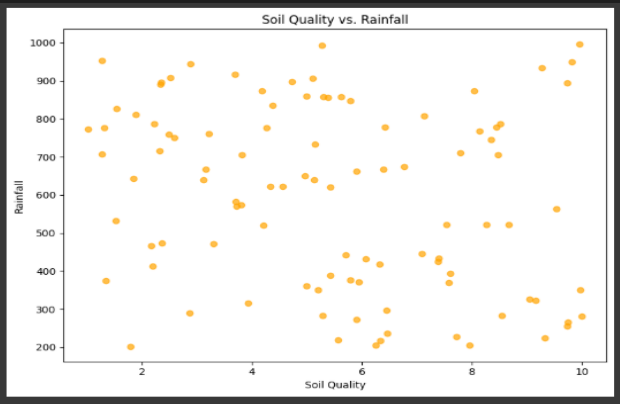
```
<ipython-input-10-00582839fd89>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.countplot(x='Seed Type', data=df, palette='Set2')
```




```
[11] plt.figure(figsize=(8, 6))
plt.scatter(df['soil_quality'], df['rainfall'], alpha=0.7, color='orange')
plt.xlabel('Soil Quality')
plt.ylabel('Rainfall')
plt.title('Soil Quality vs. Rainfall')
plt.tight_layout()
plt.show()
```




```
[12] plt.figure(figsize=(8, 6))
sns.boxplot(x='yield_category', y='rainfall', data=df, palette='pastel')
plt.xlabel('Yield Category')
plt.ylabel('Rainfall')
plt.title('Rainfall by Yield Category')
plt.tight_layout()
plt.show()
```

```
<ipython-input-12-73a65cc3b0aa>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x='yield_category', y='rainfall', data=df, palette='pastel')
```