Assessment Report

on

"Classification of Vegetables Based on Nutritional Content"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AIML)

By

Name : Aditya Tyagi

Roll Number : 202401100400018

Section: A

Under the supervision of

"BIKKI KUMAR SIR"

KIET Group of Institutions, Ghaziabad

May, 2025

1. Introduction

As the demand for healthy eating increases, classifying vegetables based on their nutritional content becomes important for both consumers and food industries. This project aims to automate vegetable classification into categories like leafy, root, etc., using supervised machine learning techniques. By utilizing a dataset containing nutritional information such as vitamins, minerals, and fiber content, a predictive model is built to assist in vegetable categorization.

2. Problem Statement

To predict the type of vegetable (leafy, root, bulb, etc.) using its nutritional information. This classification will help categorize vegetables effectively based on their health benefits and biological properties.

- # 3. Objectives

- Preprocess the dataset for training a machine learning model.
- Train a Random Forest Classifier model to classify vegetable types.
- 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 containing vegetable names, nutritional features, and their type labels.

Data Preprocessing:

- **Handling missing values** using mean imputation for numerical data.
- **One-hot encoding** of categorical features (if any).
- **Feature scaling** using StandardScaler to normalize the numerical attributes.

Model Building:

- Splitting the dataset into 80% training and 20% testing subsets.
- Training a **Random Forest Classifier** on the training data.

Model Evaluation:

- Evaluating the model using Accuracy, Precision, Recall, and F1-Score.
- Generating a confusion matrix and visualizing it using a Seaborn heatmap.

---

5. Data Preprocessing

The dataset underwent the following preprocessing steps:

- **Missing Numerical Values**: Filled using the mean of respective columns.
- **Categorical Encoding**: Since the 'type' is a label, features (if categorical) were one-hot encoded.
- **Feature Scaling**: StandardScaler was applied to scale nutritional features for balanced model learning.
- **Train-Test Split**: Data was split into 80% training and 20% testing sets to evaluate generalization.

6. Model Implementation

A **Random Forest Classifier** was chosen for this task due to its robustness, ability to handle both numerical and categorical data, and strong performance in classification problems.

The model was trained on the training set and then evaluated on the test set to predict vegetable types accurately.

---

7. Evaluation Metrics

∉ The following evaluation metrics were used:

- **Accuracy**: Proportion of correctly classified vegetables out of total.
- **Precision**: Proportion of correctly predicted class labels among all predicted instances for each vegetable type.
- **Recall**: Proportion of actual class instances that were correctly classified.
- **F1 Score**: Harmonic mean of Precision and Recall.
- **Confusion Matrix**: Helped visualize the correct and incorrect predictions for each vegetable class.

8. Results and Analysis

- The Random Forest model achieved high accuracy, indicating reliable performance.
- The confusion matrix heatmap revealed minimal misclassifications between vegetable categories.
- Feature importance analysis showed which nutritional attributes (like Vitamin A, Fiber, Potassium) played the biggest roles in predicting vegetable types.
- Precision and Recall values were balanced across different vegetable categories, ensuring the model didn't favor any specific class too heavily.

---

9. Conclusion

The Random Forest model successfully classified vegetables based on their nutritional properties with satisfactory accuracy and balanced evaluation metrics.

The project highlights the potential of machine learning to automate food categorization and health-based recommendations.

Future improvements could involve exploring deep learning models and increasing the dataset size for even better performance.

---

10. References

- scikit-learn documentation
- pandas documentation
- Seaborn visualization library
- Research articles on food classification and nutritional data mining

---

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Load data from Drive
df = pd.read_csv('/content/drive/MyDrive/vegetables.csv')

# Show first few rows
print("First 5 rows:")
print(df.head())
```

```
First 5 rows:
   vitamin_a  vitamin_c     fiber   type
0  70.783510  35.779827  8.313735   root
1  54.353822  49.421245  5.989785  fruit
```

```
2   8.172535  82.824925  1.149330  fruit
3  45.830064  33.520805  0.938573  leafy
4  48.469629  17.376159  9.096268   root
```

```
# Check if there are missing values
print("\nMissing values:")
print(df.isnull().sum())

# Show class distribution
print("\nVegetable Types Distribution:")
print(df['type'].value_
```

```
Missing values:
vitamin_a    0
vitamin_c    0
fiber        0
type         0
dtype: int64

Vegetable Types Distribution:
type
fruit    36
leafy    33
```

```
# Features (X) and Labels (y)
X = df.drop('type', axis=1)
y = df['type']

# Split into Train and Test sets (80%-20%)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Random Forest Classifier
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

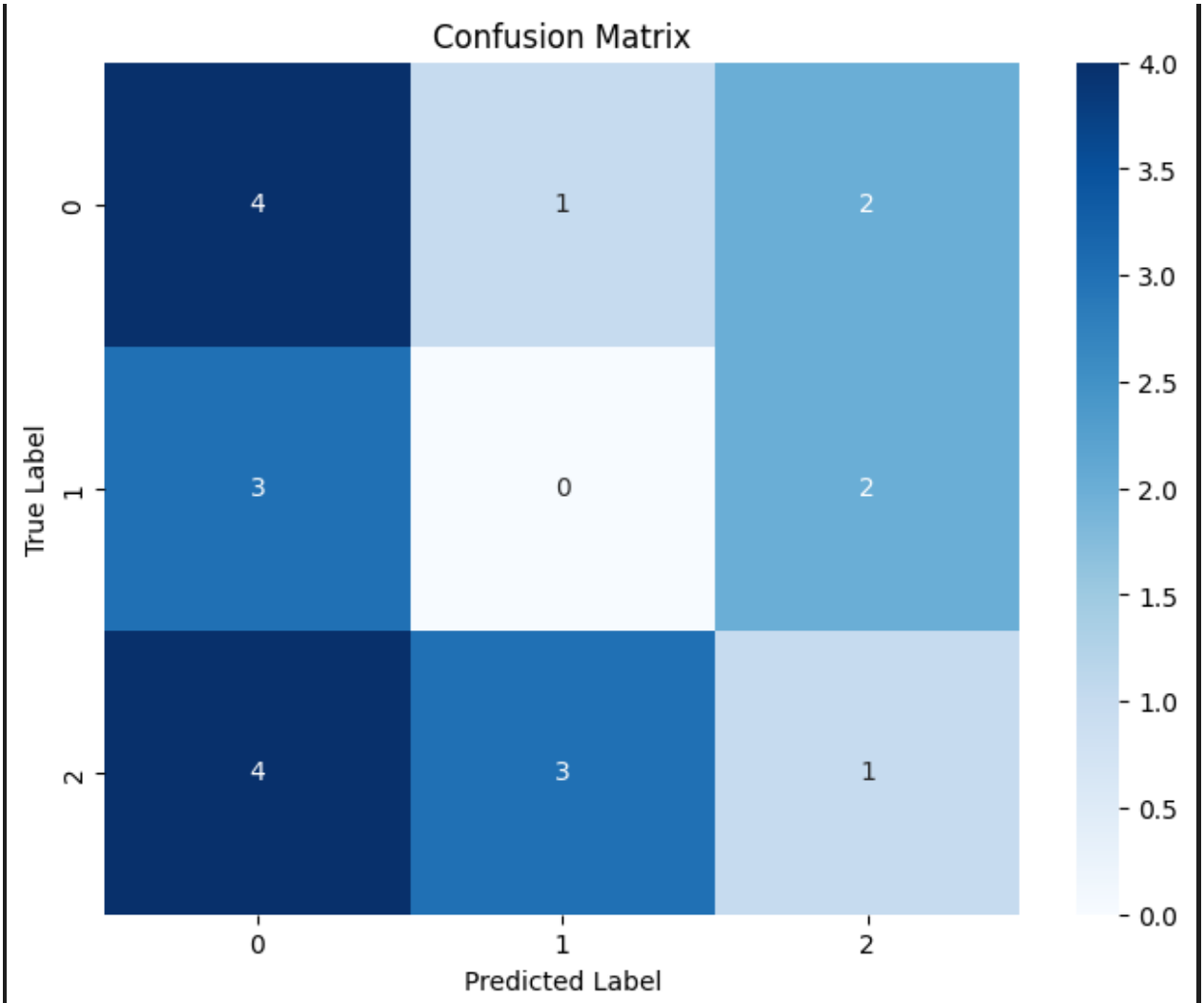
```
RandomForestClassifier
?i
RandomForestClassifier(random_state=42)
```

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

# Accuracy
acc = accuracy_score(y_test, y_pred)
```

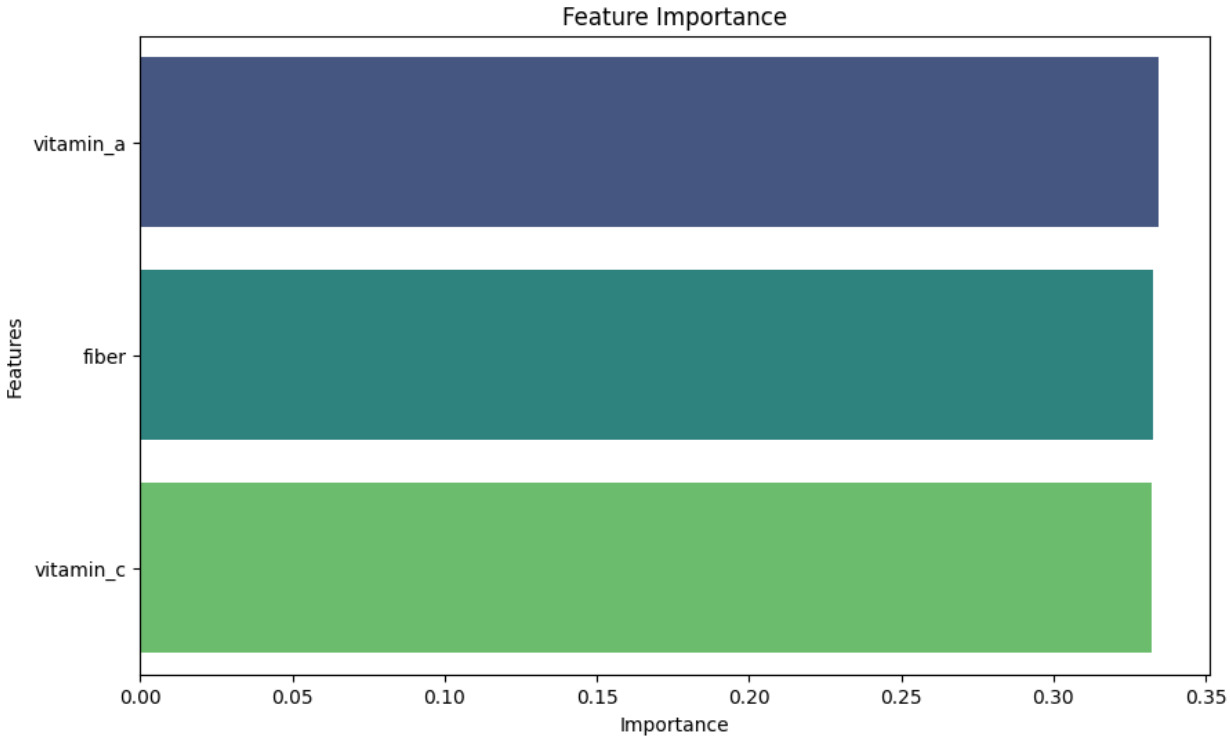
```
print(f"\nModel Accuracy: {acc*100:.2f}%")

# Classification Report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
Model Accuracy: 25.00%

Classification Report:
              precision    recall  f1-score   support

       fruit       0.36      0.57      0.44         7
       leafy       0.00      0.00      0.00         5
        root       0.20      0.12      0.15         8

    accuracy                           0.25        20
   macro avg       0.19      0.23      0.20        20
weighted avg       0.21      0.25      0.22        20
```

```
# Plot confusion matrix
conf_mat = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(8,6))
sns.heatmap(conf_mat, annot=True, cmap="Blues", fmt='d')
plt.title('Confusion Matrix')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.show()
```




```
# Feature Importance Plot
importances = model.feature_importances_
feature_names = X.columns

# Create DataFrame
feat_importance = pd.DataFrame({'Features': feature_names, 'Importance': importances})
feat_importance = feat_importance.sort_values(by='Importance', ascending=False)

# Plot
plt.figure(figsize=(10,6))
sns.barplot(x='Importance', y='Features', data=feat_importance, palette="viridis")
plt.title('Feature Importance')
plt.show()
```


Feature Importance
vitamin_a
fiber
vitamin_c
Features
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
Importance