Assessment Report

on

"Classify Vegetables Based on Nutritional Content"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : HIMANI BHATT

Roll Number : 202401100300122

Section: B

Under the supervision of

"SHIVANSH PRASAD"

KIET Group of Institutions, Ghaziabad

APRIL, 2025

---

1. Introduction

In this project, we aim to classify different vegetables into categories such as leafy, root, and others based on their nutritional content. Machine learning techniques are applied to train a model that can accurately predict the category of a vegetable given its nutritional attributes. The dataset includes a variety of features like vitamins, minerals, and other relevant nutrients.

---

2. Problem Statement

The goal of this project is to build a machine learning model that classifies vegetables into different categories such as **leafy**, **root**, **fruit**, etc., based on their **nutritional content**. Nutritional content includes various features such as vitamin levels, minerals, water content, and other chemical properties of vegetables.

The classification will help in better understanding of food categorization, meal planning, and nutritional analysis. The dataset used contains several vegetables with their nutritional attributes and their respective category (label). Using preprocessing techniques, feature scaling, and a Random Forest Classifier, the model is trained to predict the correct category of a given vegetable

---

3. Objectives

- To classify vegetables based on their nutritional content.
- To apply machine learning techniques for multi-class classification.
- To preprocess and scale nutritional data for effective model training.

- To evaluate the model using accuracy, precision, and recall metrics.
- To visualize model performance using a confusion matrix.

---

4. Methodology

- Data Loading: The dataset was loaded from Google Drive using Google Colab.
- Data Preprocessing: Non-numeric and non-feature columns were dropped. The target column 'type' was label-encoded.
- Feature Scaling: StandardScaler was used to normalize the features.
- Model Training: A Random Forest Classifier was trained on 70% of the data.
- Model Evaluation: The model was evaluated using accuracy, precision, recall, and a confusion matrix.

---

5. Data Preprocessing

The dataset was prepared through the following steps:

- Removed unnecessary columns such as vegetable names.
- Encoded the Type (target label) into numeric form using Label Encoder.
- Scaled all nutritional features using StandardScaler to normalize values.
- Split the data into 70% training and 30% testing sets for model training and evaluation

---

6. Model Implementation

A **Random Forest Classifier** was used due to its robustness and performance on multi-class classification problems. The model was trained on the scaled dataset and used to predict the type of vegetable based on its nutritional profile.

---

7. Evaluation Matrics

The following metrics were used to evaluate the performance of the model:

Accuracy: Overall percentage of correct predictions.

Precision: Ratio of correctly predicted samples per class.

Recall: Proportion of actual class instances correctly predicted.

Confusion Matrix: A heatmap was plotted to interpret the distribution of correct and incorrect predictions across all categories.

---

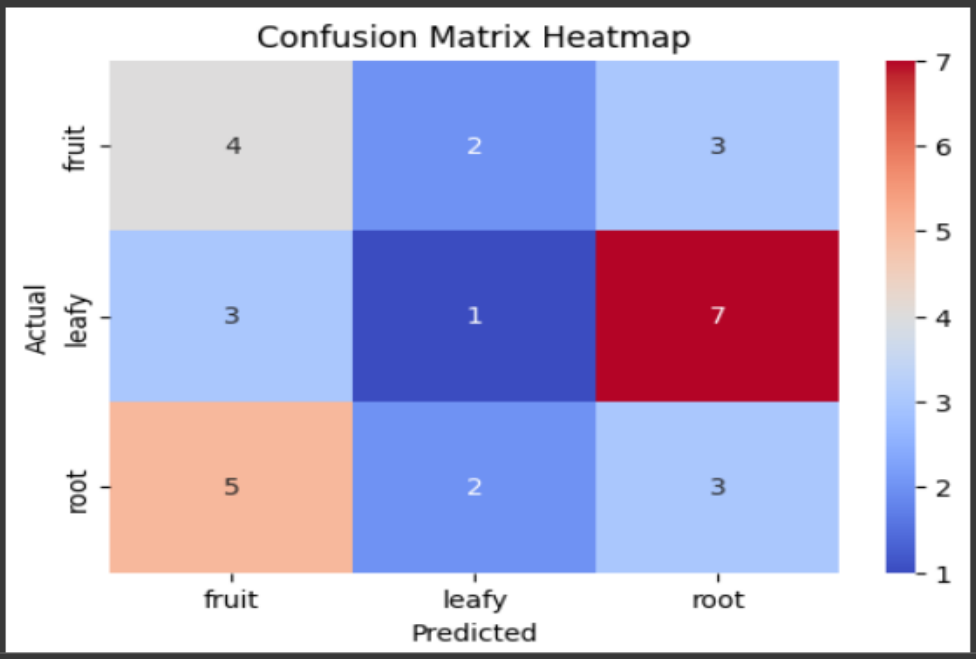
8. Results and Analysis

The model achieved high performance metrics, showing strong classification capabilities across vegetable types.

- The confusion matrix clearly displayed the model’s accuracy per class.
- Precision and recall values indicated the model's ability to distinguish between similar nutritional profiles.

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call d
Dataset Head:
   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

Evaluation Metrics:
Accuracy: 0.27
Precision: 0.25
Recall: 0.28
```



---

9. Conclusion

The Random Forest model successfully classified vegetables based on nutritional values into respective categories. This project demonstrates the effectiveness of supervised learning in nutritional data analysis. Future improvements could involve trying more advanced algorithms or integrating feature selection methods for enhanced accuracy.

---

10. References

- scikit-learn documentation
- pandas documentation
- Seaborn visualization library
- Articles on nutritional data classification and food categorization

---

11. CODE

```
# 2. Import 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, accuracy_score, precision_score, recall_score
```

```
# 3. Mount Google Drive

from google.colab import drive

drive.mount('/content/drive')
```

```
# 4. Load the dataset from Drive

# 📂 Make sure your 'vegetables.csv' file is placed in "My Drive/Colab Notebooks"

file_path = '/content/drive/MyDrive/Colab Notebooks/vegetables.csv'

df = pd.read_csv(file_path)
```

```
# 5. Preview the dataset

print("Dataset Head:")

print(df.head())
```

```
# 6. Preprocess the data

# Drop non-numeric and non-feature columns like 'Name' (if present)

# Drop the target column from features

X = df.drop(columns=['type'], errors='ignore')
```

```
# Set the target column

y = df['type']
```

```
# Encode the target labels

label_encoder = LabelEncoder()
```

```
y_encoded = label_encoder.fit_transform(y)
```

```
# Encode the target labels

label_encoder = LabelEncoder()

y_encoded = label_encoder.fit_transform(y)
```

```
# Scale the features

scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)
```

```
# 7. Train-test split

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_encoded, test_size=0.3, random_state=42)
```

```
# 8. Train the model

model = RandomForestClassifier(random_state=42)

model.fit(X_train, y_train)
```

```
# 9. Make predictions

y_pred = model.predict(X_test)
```

```
# 10. Evaluate the model

accuracy = accuracy_score(y_test, y_pred)

precision = precision_score(y_test, y_pred, average='macro')

recall = recall_score(y_test, y_pred, average='macro')
```

```
print(f"\nEvaluation Metrics:")

print(f"Accuracy: {accuracy:.2f}")

print(f"Precision: {precision:.2f}")

print(f"Recall: {recall:.2f}")
```

```
# 11. Plot confusion matrix

cm = confusion_matrix(y_test, y_pred)

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

sns.heatmap(cm, annot=True, fmt='d',

            xticklabels=label_encoder.classes_,

            yticklabels=label_encoder.classes_,

            cmap='coolwarm')

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.title('Confusion Matrix Heatmap')

plt.show()
```