SHYAM MOHAN KALYAN PALAPARTHY

811294943

ADVANCE MECHINE LEARNING

ASSIGNMENT-1

Summary Report

The synthetic dataset was taken using **make_blobs** with the samples 200, which was divided into 3 different clusters centres and with 2 different features each. Now we have used KNN-K-Nearest Neighbours classification was used to classify the data. This data was now split into the training dataset (80%) and another is testing dataset (20%). The classifier was trained using 3 nearest neighbours(**n_neighbors=3**).

Thereafter when predications were made on the test set, The model accuracy was 0.925 or 92.5%. This shows that the KNN classifier which was created was able to do the classification of the test samples with excellent accuracy.

The two scatter plots were created:

1.Traning set-output: This scatter plot shows the training data.

2.Test set-predictions output-This scatter plot shows the test data.

Code Explanation:

Data Creation: we used the **make_blobs** to create a synthetic dataset with 200 sample which was divided into 3 different clusters centres and with 2 different features each.

Data Splitting: This data was now split into the training dataset (80%) and another is testing dataset (20%).

KNN Classifier: The K-Nearest Neighbours classifies with **n_neighbors=3**.The training data using classifier with **.fit()** method.

Prediction: This function **.predict()** is used to check the predictions on the test set.

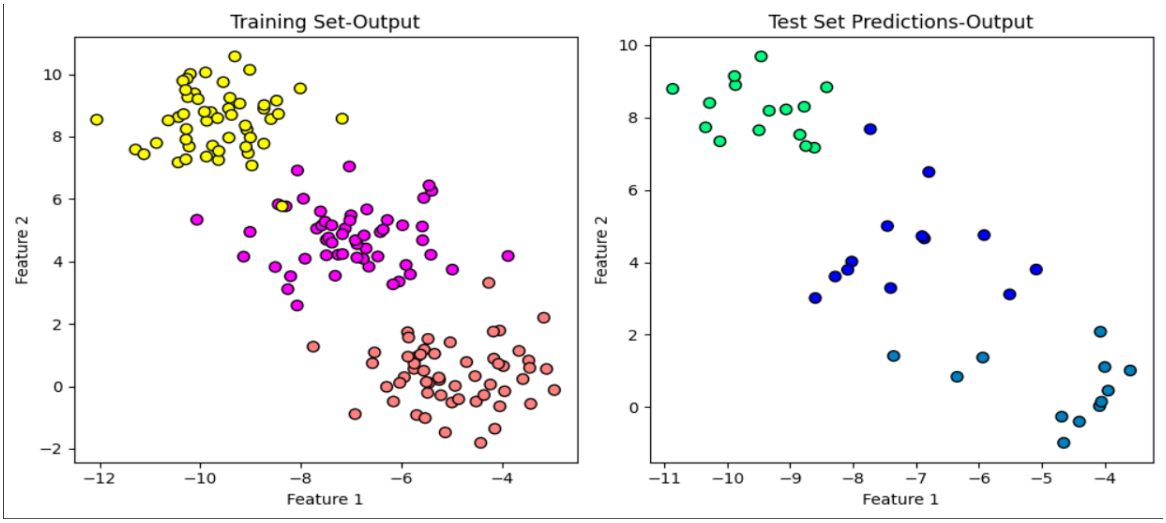
Evaluations: The predicted values and Actual Target Values are printed, and these calculate the accuracy using the command **accuracy_score.**

Plotting: Two scatter plots are created to display the training data and test data predictions. Now we have adjusted the layout using the plt.tight_layout() to avoid overlapping.

Accuracy: The model which we have created achieved the accuracy of 0.925(92.5%) on the test data, this shows that KNN classifier done perfectly on the synthetic dataset.

Output:

```
Predictions from the classifier:
[1 1 2 0 1 0 2 2 0 2 2 1 0 0 1 1 1 1 1 0 0 1 2 0 1 2 2 2 0 2 2 0 2 2 0 0 2
 2 1 0]
Target values:
[1 1 2 0 1 0 2 2 0 2 2 1 0 0 1 1 1 1 1 0 0 1 2 0 1 2 2 2 0 2 2 0 2 2 0 0 2
 2 1 0]
Accuracy: 1.000
```



Appendix:

```
# Importing dataset from Scikit-Learn Library
from sklearn import datasets
# Importing this to calculate the accuracy score
from sklearn.metrics import accuracy_score
#Importing this to split the dataset into traning and testing dataset.
from sklearn.model_selection import train_test_split
# For implementing the KNN classifier
from sklearn.neighbors import KNeighborsClassifier
# For generating synthetic datasets (blobs) for classification
from sklearn.datasets import make_blobs
# For plotting the data using Matplotlib
import matplotlib.pyplot as plt
# For numerical operations using NumPy
import numpy as np
```

```
#Generate synthetic data using make_blobs
b_features, b_labels = make_blobs(n_samples=200, centers=3, n_features=2,
random_state=12)

#Split the synthetic data into training and testing sets
b_train_f, b_test_f, b_train_l, b_test_l = train_test_split(b_features, b_labels, train_size=0.8,
test_size=0.2, random_state=12)

#Initialize the KNN classifier
knn_clf = KNeighborsClassifier(n_neighbors=3)

#Train the KNN classifier using the training data
knn_clf.fit(b_train_f, b_train_l)

#Make predictions on the test set
b_test_pred = knn_clf.predict(b_test_f)

# Print the predictions, actual target values, and accuracy of the model
print(f"Predictions from the classifier:\n{b_test_pred}")
print(f"Target values:\n{b_test_l}")
print(f"Accuracy: {accuracy_score(b_test_pred, b_test_l):.3f}")

# Plotting the training and test data
plt.figure(figsize=(10, 5))  # Set the figure size

# Plot 1: Training data
plt.subplot(1, 2, 1)  # Create a 1x2 grid of plots, this is the first plot
plt.scatter(b_train_f[:, 0], b_train_f[:, 1], c=b_train_l, cmap='spring', edgecolor='k', s=50)
plt.title('Training Set-Output')  # Set the title of the plot
plt.xlabel('Feature 1')  # Label for the x-axis
plt.ylabel('Feature 2')  # Label for the y-axis
```

```
# Plot 2: Test data with predictions
plt.subplot(1, 2, 2)  # Create the second plot in the 1x2 grid
plt.scatter(b_test_f[:, 0], b_test_f[:, 1], c=b_test_pred, cmap='winter', edgecolor='k', s=50)
plt.title('Test Set Predictions-Output')  # Set the title of the plot
plt.xlabel('Feature 1')  # Label for the x-axis
plt.ylabel('Feature 2')  # Label for the y-axis

# Adjust layout to prevent overlap and display the plots
plt.tight_layout()
plt.show()
```