Name: - Adarsh Upadhyay

Reg. No.: - 23MCA0237

#Nave Bayes on Iris Dataset

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score

#Loading the datasets
X, y = load_iris(return_X_y=True)

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.4, random_state=42)

# Initializing NB model
model = GaussianNB()

# Training the model
model.fit(X_train, y_train)
```

```
GaussianNB()
```

```
# Making predictions on the testing set
y_pred_train = model.predict(X_train)
y_pred_test = model.predict(X_test)

# Calculating accuracy
train_accuracy_NB = accuracy_score(y_train, y_pred_train)
test_accuracy_NB = accuracy_score(y_test, y_pred_test)
print("Accuracy train:", train_accuracy_NB)
print("Accuracy test:", test_accuracy_NB)
```

```
Accuracy train: 0.9444444444444444
Accuracy test: 0.9666666666666667
```

Multi-layer Perceptron Classifier on Iris Dataset

```
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score, classification_report

# Load the dataset from CSV
X, y = load_iris(return_X_y=True)

# Splitting the dataset into training and testing sets
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.4, random_state=42)

# Initialize the Multi-layer Perceptron classifier
mlp_classifier = MLPClassifier(hidden_layer_sizes=(100, 50),
activation='relu', solver='adam', max_iter=1000, random_state=42)

# Train the model
mlp_classifier.fit(X_train, y_train)

# Predictions
y_pred_train = mlp_classifier.predict(X_train)
y_pred_test = mlp_classifier.predict(X_test)

# Evaluate the model
train_accuracy_MLPC = accuracy_score(y_train, y_pred_train)
test_accuracy_MLPC = accuracy_score(y_test, y_pred_test)

print(f"Accuracy on Training Set: {train_accuracy_MLPC:.2f}")
print(f"Accuracy on Test Set: {test_accuracy_MLPC:.2f}")

# Print classification report
print("\nClassification Report on Test Set:")
print(classification_report(y_test, y_pred_test))
```

```
Accuracy on Training Set: 0.99
Accuracy on Test Set: 0.98

Classification Report on Test Set:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        23
           1       0.95      1.00      0.97        19
           2       1.00      0.94      0.97        18

    accuracy                           0.98        60
   macro avg       0.98      0.98      0.98        60
weighted avg       0.98      0.98      0.98        60
```

Comparision of train and test acurracy

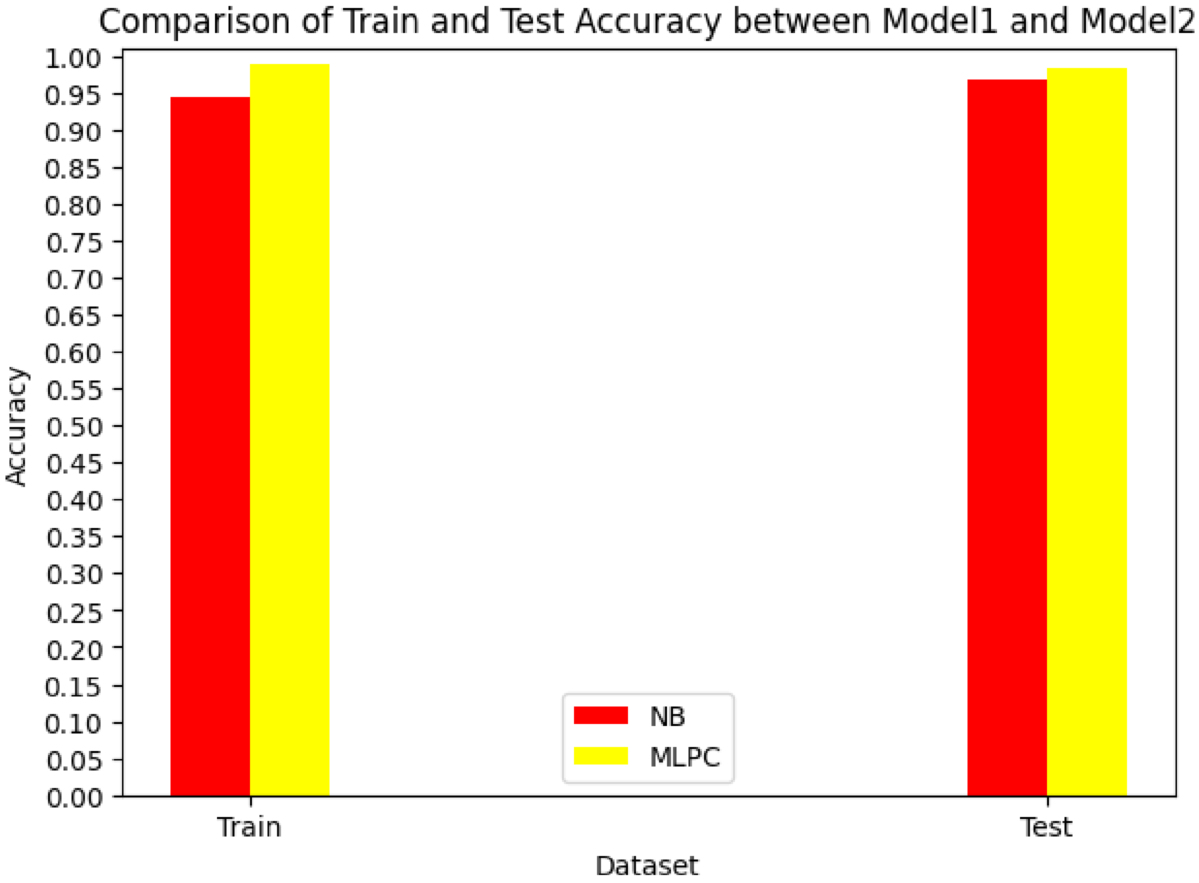
```
import numpy as np
import matplotlib.pyplot as plt


# Labels
labels = ['Train', 'Test']
models = ['NB', 'MLPC']
```

```
# Width of the bars
bar_width = 0.10
index = np.arange(len(labels))
y_ticks = [i/100 for i in range(0,101,5)]


# Plotting the grouped bar graph
plt.figure(figsize=(7, 5))
plt.bar(index - bar_width/2, [train_accuracy_NB, test_accuracy_NB],
bar_width, label=models[0], color='red')
plt.bar(index + bar_width/2, [train_accuracy_MLPC,
test_accuracy_MLPC], bar_width, label=models[1], color='yellow')

# Adding labels and title
plt.xlabel('Dataset')
plt.ylabel('Accuracy')
plt.title('Comparison of Train and Test Accuracy between Model1 and
Model2')
plt.xticks(index, labels)
plt.ylim(0, 1.01)  # Setting y-axis limit to be between 0 and 1
plt.yticks(y_ticks)
plt.legend()
plt.show()
```


Comparison of Train and Test Accuracy between Model1 and Model2
Accuracy
1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Train
Test
Dataset
NB
MLPC