EX.NO :

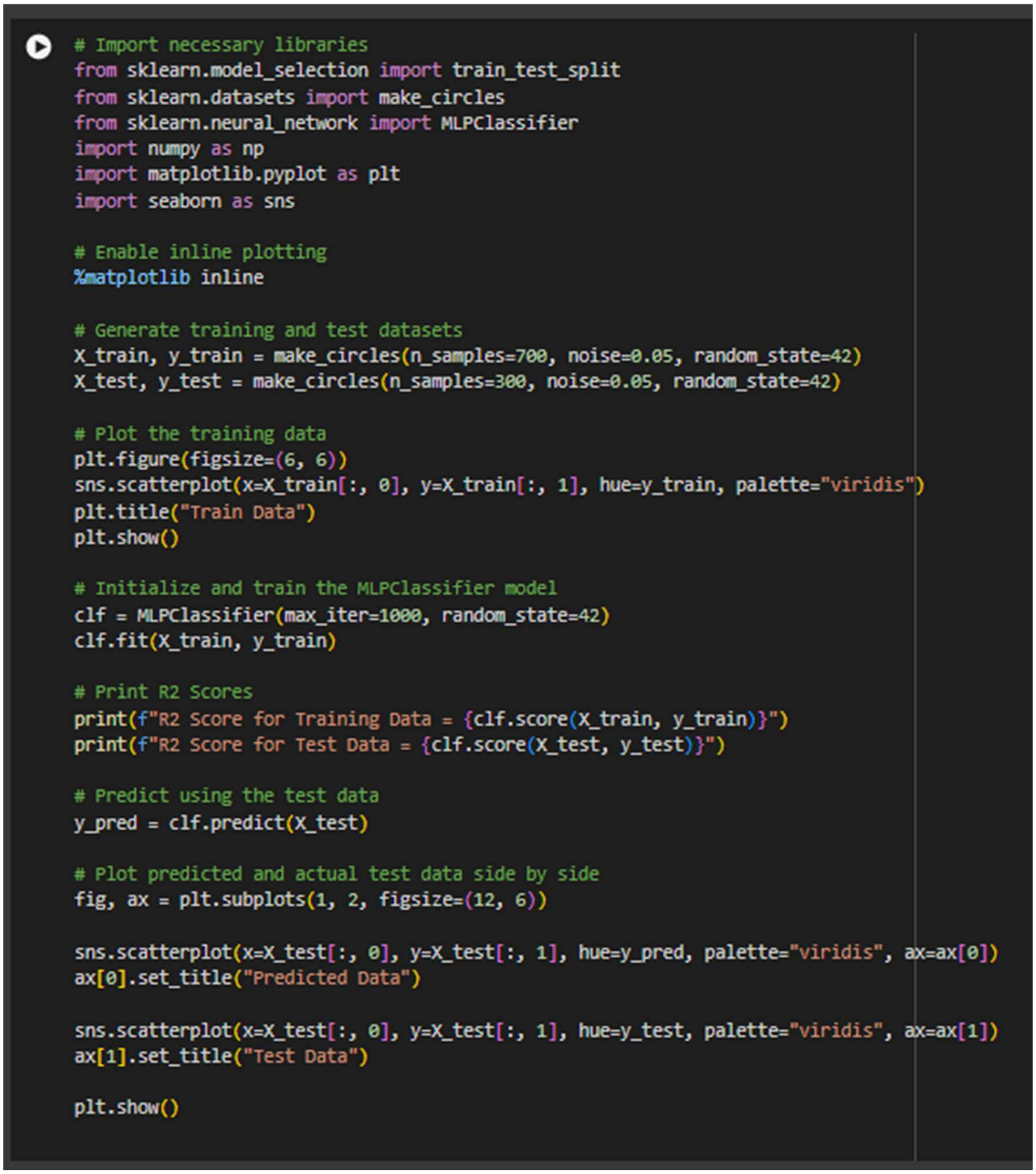
DATE :

IMPLEMENTING ARTIFICIAL NEURAL NETWORKS FOR AN APPLICATION USING PYTHON - CLASSIFICATION

AIM :

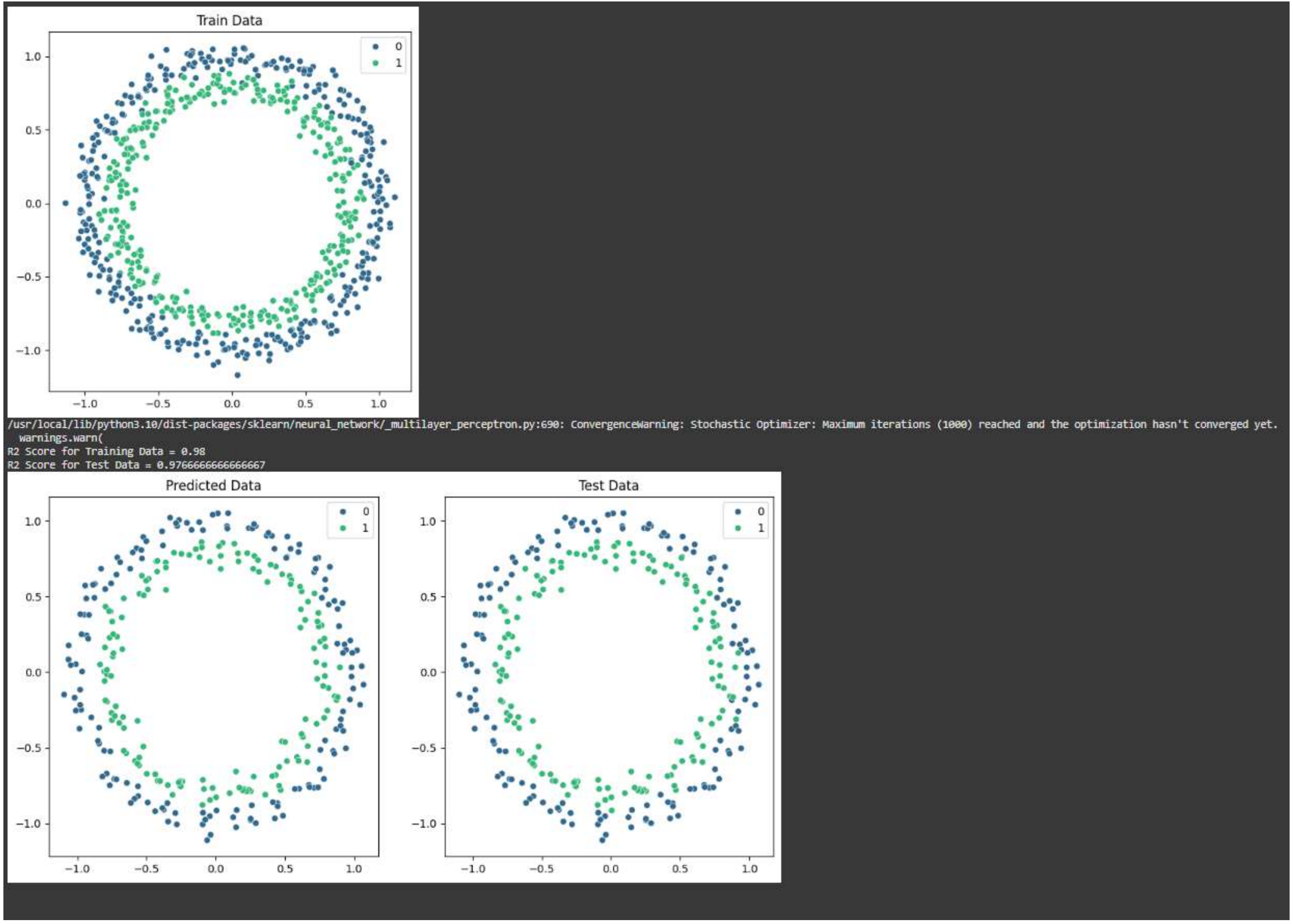
To implementing artificial neural networks for an application in classification using python.

Source Code :



```
# Import necessary libraries
from sklearn.model_selection import train_test_split
from sklearn.datasets import make_circles
from sklearn.neural_network import MLPClassifier
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Enable inline plotting
%matplotlib inline

# Generate training and test datasets
X_train, y_train = make_circles(n_samples=700, noise=0.05, random_state=42)
X_test, y_test = make_circles(n_samples=300, noise=0.05, random_state=42)

# Plot the training data
plt.figure(figsize=(6, 6))
sns.scatterplot(x=X_train[:, 0], y=X_train[:, 1], hue=y_train, palette="viridis")
plt.title("Train Data")
plt.show()

# Initialize and train the MLPClassifier model
clf = MLPClassifier(max_iter=1000, random_state=42)
clf.fit(X_train, y_train)

# Print R2 Scores
print(f"R2 Score for Training Data = {clf.score(X_train, y_train)}")
print(f"R2 Score for Test Data = {clf.score(X_test, y_test)}")

# Predict using the test data
y_pred = clf.predict(X_test)

# Plot predicted and actual test data side by side
fig, ax = plt.subplots(1, 2, figsize=(12, 6))

sns.scatterplot(x=X_test[:, 0], y=X_test[:, 1], hue=y_pred, palette="viridis", ax=ax[0])
ax[0].set_title("Predicted Data")

sns.scatterplot(x=X_test[:, 0], y=X_test[:, 1], hue=y_test, palette="viridis", ax=ax[1])
ax[1].set_title("Test Data")

plt.show()
```

OUTPUT :



```
/usr/local/lib/python3.10/dist-packages/sklearn/neural_network/_multilayer_perceptron.py:690: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (1000) reached and the optimization hasn't converged yet.
  warnings.warn(
R2 Score for Training Data = 0.98
R2 Score for Test Data = 0.9766666666666667
```

RESULT :

Thus Program is Executed Successfully And Output is Verified.