

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
```

```
data = load_iris()
X = data.data
y = data.target
```

```
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.3, random_state=42)
```

```
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
pca = PCA(n_components=2)
X_train_pca = pca.fit_transform(X_train)
X_test_pca = pca.transform(X_test)
```

```
lda = LDA(n_components=2)
X_train_lda = lda.fit_transform(X_train, y_train)
X_test_lda = lda.transform(X_test)
```

```
clf_pca = LogisticRegression()
clf_pca.fit(X_train_pca, y_train)

clf_lda = LogisticRegression()
clf_lda.fit(X_train_lda, y_train)
```

▼ **LogisticRegression** ⓘ ?

```
LogisticRegression()
```

```
y_pred_pca = clf_pca.predict(X_test_pca)
y_pred_lda = clf_lda.predict(X_test_lda)
```

```
print("Accuracy with PCA:", accuracy_score(y_test, y_pred_pca))
print("Accuracy with LDA:", accuracy_score(y_test, y_pred_lda))

print("\nClassification Report (PCA):")
print(classification_report(y_test, y_pred_pca))

print("\nClassification Report (LDA):")
print(classification_report(y_test, y_pred_lda))
```

```
Accuracy with PCA: 0.9111111111111111
Accuracy with LDA: 1.0
```

```
Classification Report (PCA):
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	0.91	0.77	0.83	13
2	0.80	0.92	0.86	13
accuracy			0.91	45
macro avg	0.90	0.90	0.90	45
weighted avg	0.92	0.91	0.91	45

```
Classification Report (LDA):
```

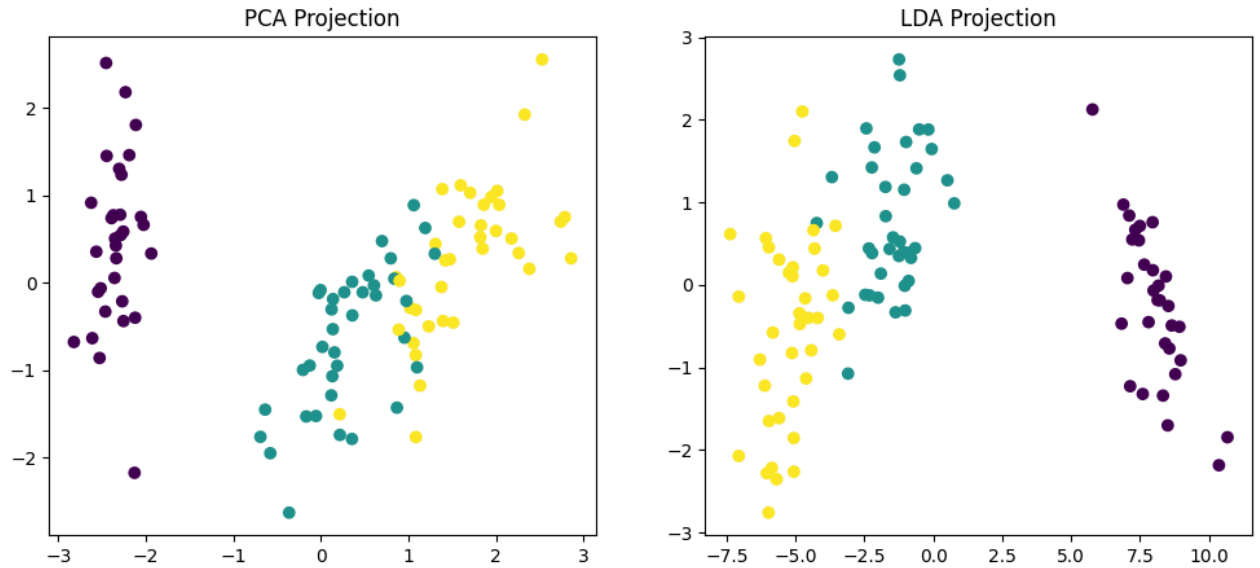
	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	1.00	1.00	1.00	13
2	1.00	1.00	1.00	13
accuracy			1.00	45
macro avg	1.00	1.00	1.00	45
weighted avg	1.00	1.00	1.00	45

```
plt.figure(figsize=(12,5))

plt.subplot(1,2,1)
plt.scatter(X_train_pca[:,0], X_train_pca[:,1], c=y_train)
plt.title("PCA Projection")

plt.subplot(1,2,2)
plt.scatter(X_train_lda[:,0], X_train_lda[:,1], c=y_train)
plt.title("LDA Projection")

plt.show()
```



Start coding or [generate](#) with AI.