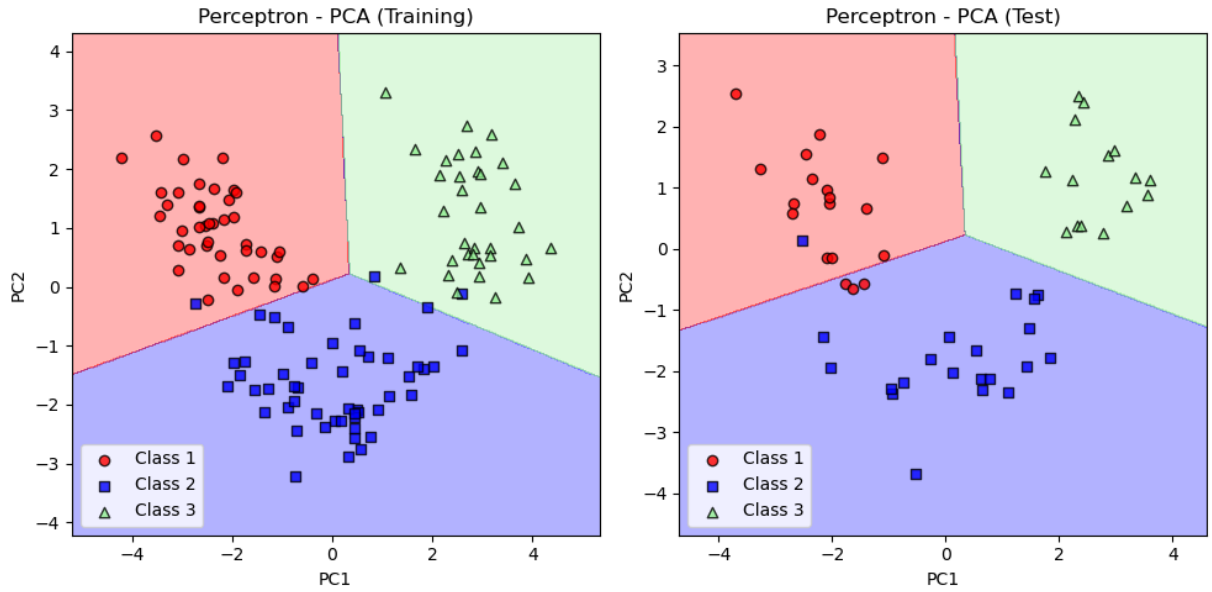
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.discriminant_analysis import
LinearDiscriminantAnalysis as LDA
from sklearn.linear_model import Perceptron,
LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap

# 載入資料
df_wine =
pd.read_csv('https://archive.ics.uci.edu/ml/machine-
learning-databases/wine/wine.data', header=None)
df_wine.columns = ['Class label', 'Alcohol', 'Malic acid',
'Ash', 'Alcalinity of ash', 'Magnesium',
                   'Total phenols', 'Flavanoids',
'Nonflavanoid phenols', 'Proanthocyanins',
                   'Color intensity', 'Hue', 'OD280/OD315 of
diluted wines', 'Proline']

# 分割特徵與標籤
X = df_wine.iloc[:, 1:].values
y = df_wine.iloc[:, 0].values

# 分割訓練集與測試集
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, stratify=y, random_state=0)

# 標準化
sc = StandardScaler()
X_train_std = sc.fit_transform(X_train)
X_test_std = sc.transform(X_test)

# PCA降維
```

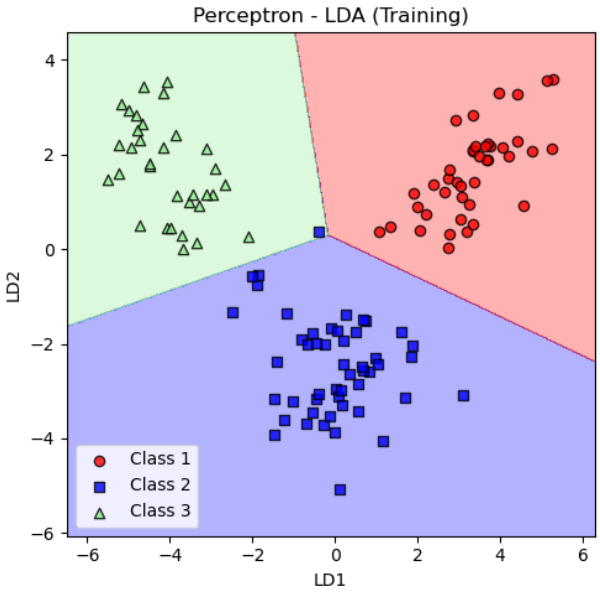
```
pca = PCA(n_components=2)
X_train_pca = pca.fit_transform(X_train_std)
X_test_pca = pca.transform(X_test_std)

# LDA降維
lda = LDA(n_components=2)
X_train_lda = lda.fit_transform(X_train_std, y_train)
X_test_lda = lda.transform(X_test_std)

def plot_decision_regions(X, y, classifier, test_idx=None,
resolution=0.02):
    markers = ('o', 's', '^', 'v', '<')
    colors = ('red', 'blue', 'lightgreen', 'gray', 'cyan')
    cmap = ListedColormap(colors[:len(np.unique(y))])

    x1_min, x1_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    x2_min, x2_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx1, xx2 = np.meshgrid(np.arange(x1_min, x1_max,
resolution),
                           np.arange(x2_min, x2_max,
resolution))
    lab = classifier.predict(np.array([xx1.ravel(),
xx2.ravel()]).T)
    lab = lab.reshape(xx1.shape)
    plt.contourf(xx1, xx2, lab, alpha=0.3, cmap=cmap)
    plt.xlim(xx1.min(), xx1.max())
    plt.ylim(xx2.min(), xx2.max())

    for idx, cl in enumerate(np.unique(y)):
        plt.scatter(x=X[y == cl, 0], y=X[y == cl, 1],
                    alpha=0.8, c=colors[idx],
                    marker=markers[idx], label=f'Class
{cl}',
                    edgecolors='black')  # 修正這裡：edcolor -
> edgecolors

# 定義分類器清單
classifiers = {
    'Perceptron': Perceptron(eta0=0.1, random_state=0),
    'Logistic Regression':
LogisticRegression(random_state=0),
    'Support Vector Machine': SVC(kernel='rbf',
random_state=0),
```

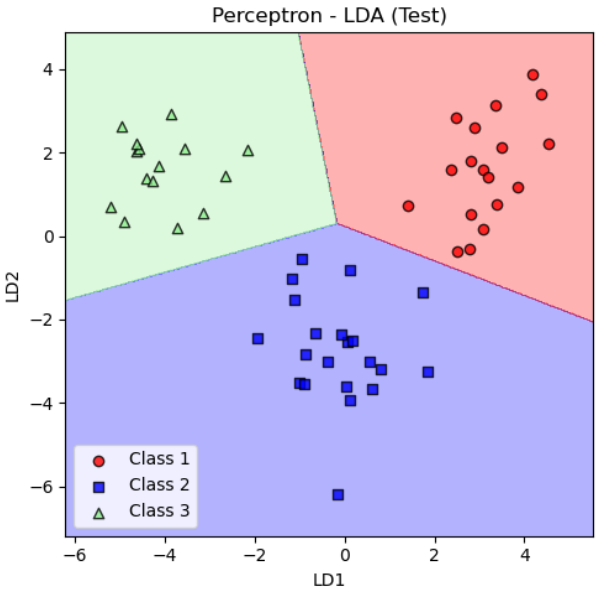
```
    'Decision Tree':
DecisionTreeClassifier(random_state=0),
    'Random Forest':
RandomForestClassifier(n_estimators=100, random_state=0),
    'K-Nearest Neighbors':
KNeighborsClassifier(n_neighbors=5)
}

# 儲存結果的字典
results = {'PCA': {}, 'LDA': {}}

# 訓練和評估每個分類器
for name, clf in classifiers.items():
    print(f"\nProcessing {name}...")

    # PCA
    clf.fit(X_train_pca, y_train)
    train_pred_pca = clf.predict(X_train_pca)
    test_pred_pca = clf.predict(X_test_pca)
    results['PCA'][name] = {
        'train_acc': accuracy_score(y_train,
train_pred_pca),
        'test_acc': accuracy_score(y_test, test_pred_pca)
    }

    # 繪製決策區域 (PCA)
    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plot_decision_regions(X_train_pca, y_train, clf)
    plt.title(f'{name} - PCA (Training)')
    plt.xlabel('PC1')
    plt.ylabel('PC2')
    plt.legend(loc='lower left')

    plt.subplot(1, 2, 2)
    plot_decision_regions(X_test_pca, y_test, clf)
    plt.title(f'{name} - PCA (Test)')
    plt.xlabel('PC1')
    plt.ylabel('PC2')
    plt.legend(loc='lower left')
    plt.tight_layout()
    plt.show()
```

```
        # LDA
        clf.fit(X_train_lda, y_train)
        train_pred_lda = clf.predict(X_train_lda)
        test_pred_lda = clf.predict(X_test_lda)
        results['LDA'][name] = {
            'train_acc': accuracy_score(y_train,
    train_pred_lda),
            'test_acc': accuracy_score(y_test, test_pred_lda)
        }

        # 繪製決策區域 (LDA)
        plt.figure(figsize=(10, 5))
        plt.subplot(1, 2, 1)
        plot_decision_regions(X_train_lda, y_train, clf)
        plt.title(f'{name} - LDA (Training)')
        plt.xlabel('LD1')
        plt.ylabel('LD2')
        plt.legend(loc='lower left')

        plt.subplot(1, 2, 2)
        plot_decision_regions(X_test_lda, y_test, clf)
        plt.title(f'{name} - LDA (Test)')
        plt.xlabel('LD1')
        plt.ylabel('LD2')
        plt.legend(loc='lower left')
        plt.tight_layout()
```

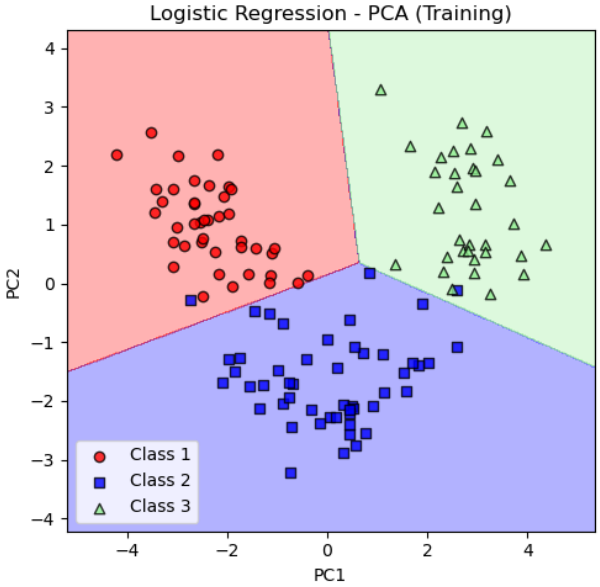
```
Processing Perceptron...
```

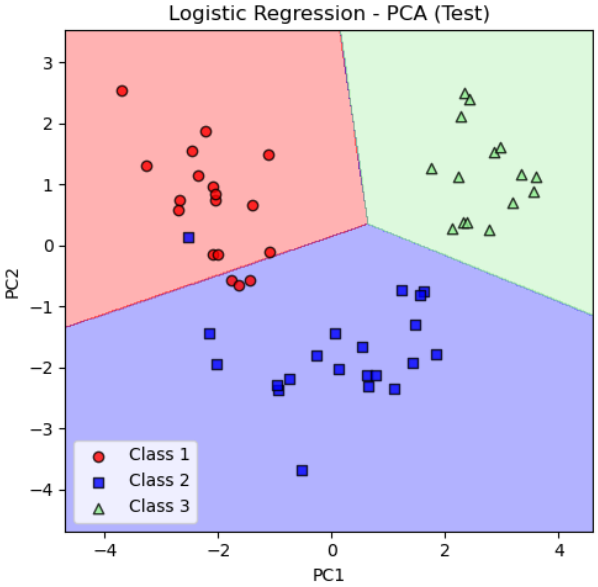


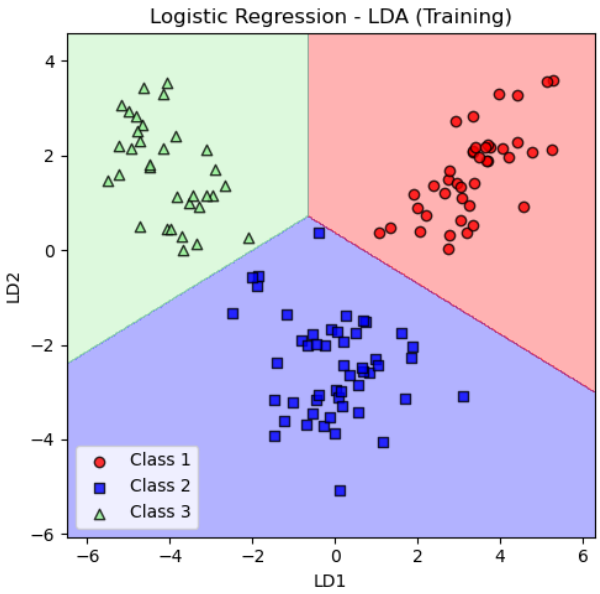


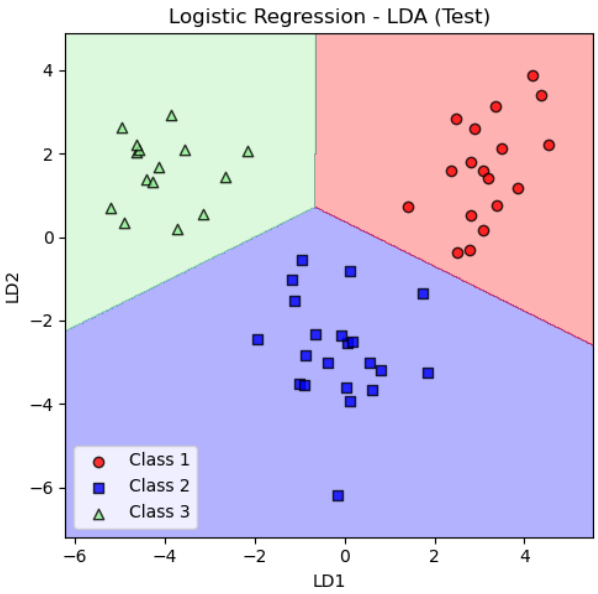



```
Processing Logistic Regression...
```

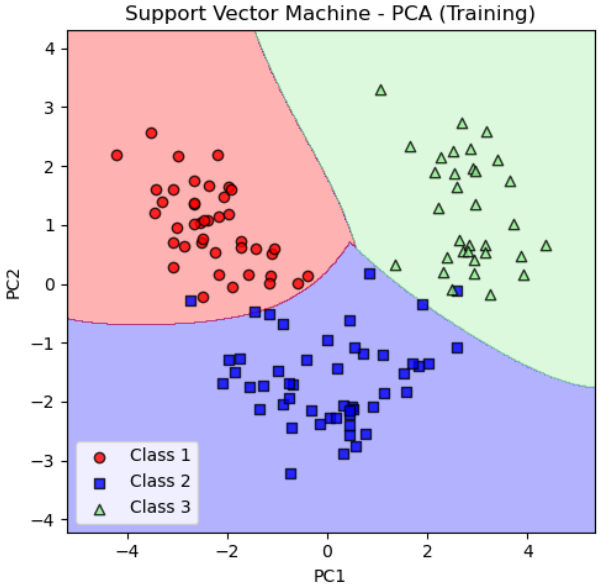


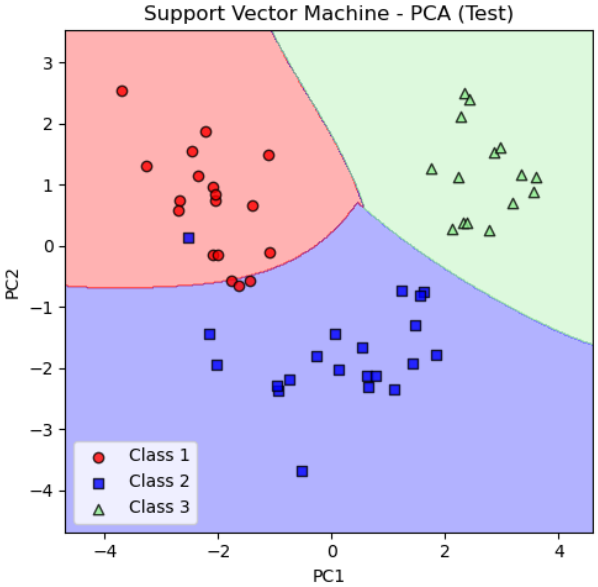


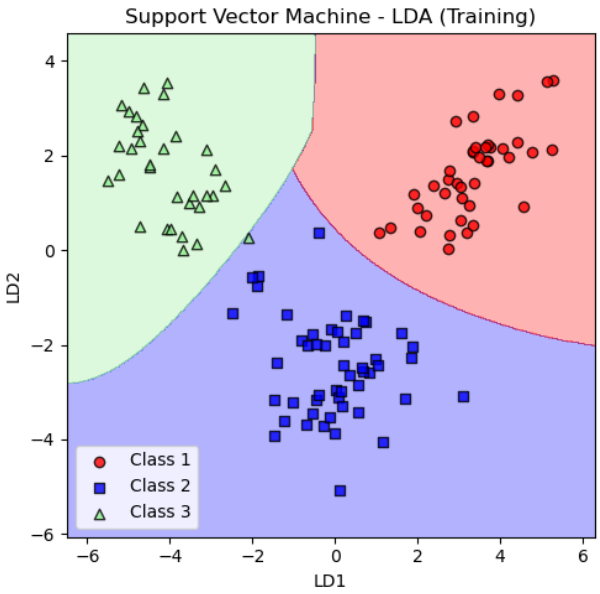


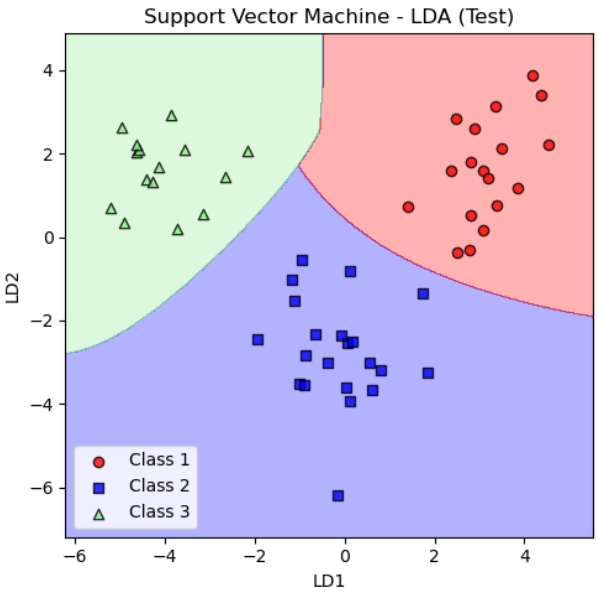



```
Processing Support Vector Machine...
```

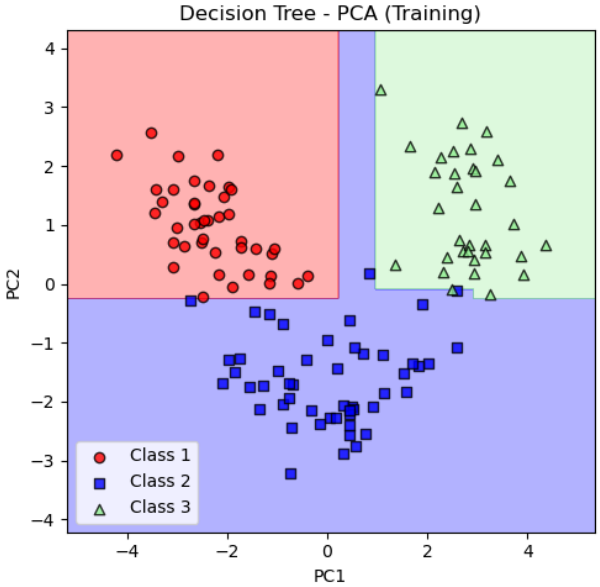


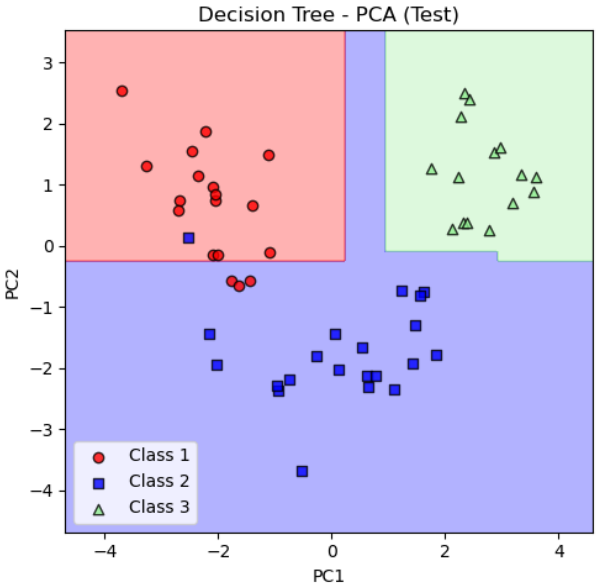


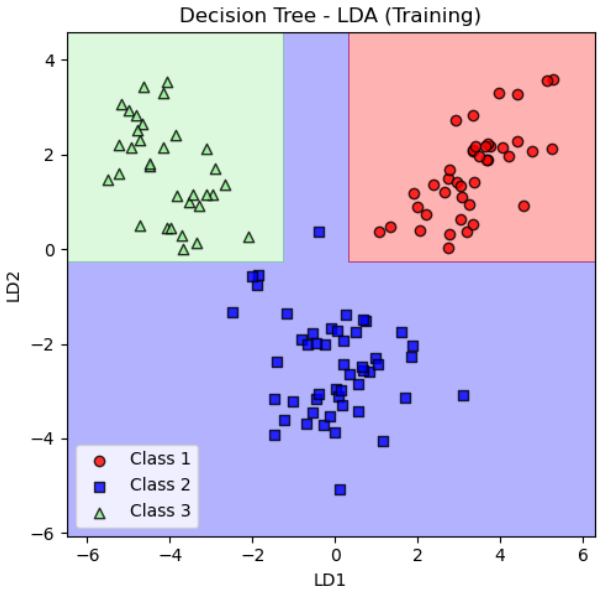


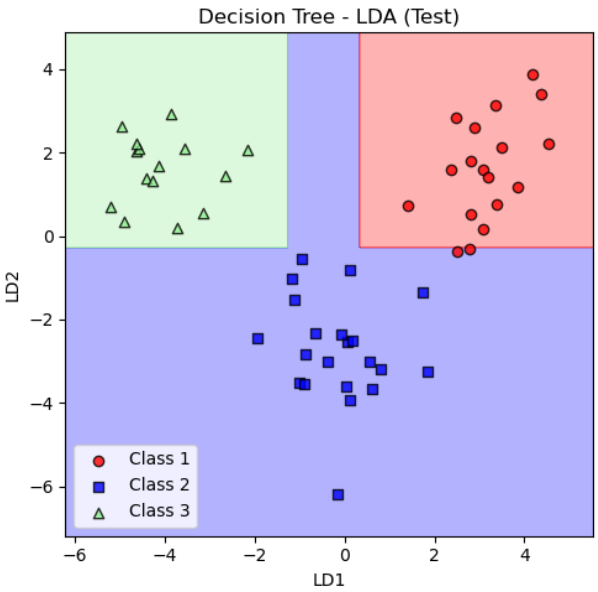


Processing Decision Tree...

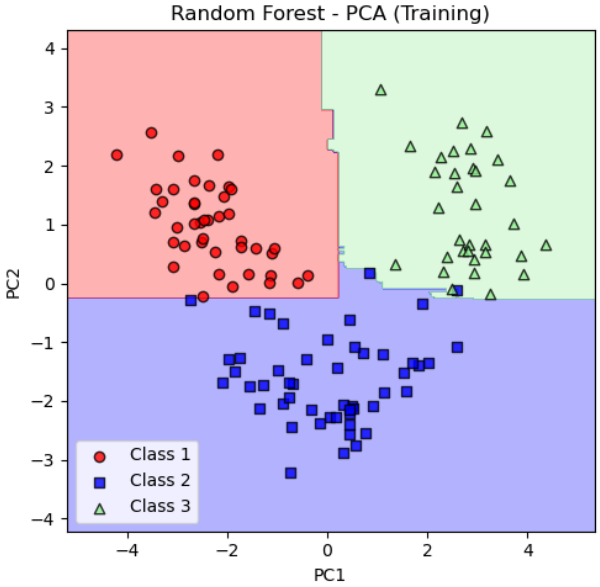


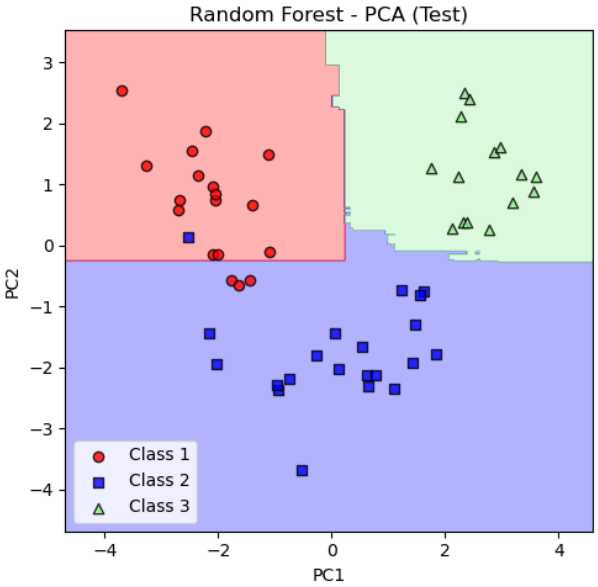


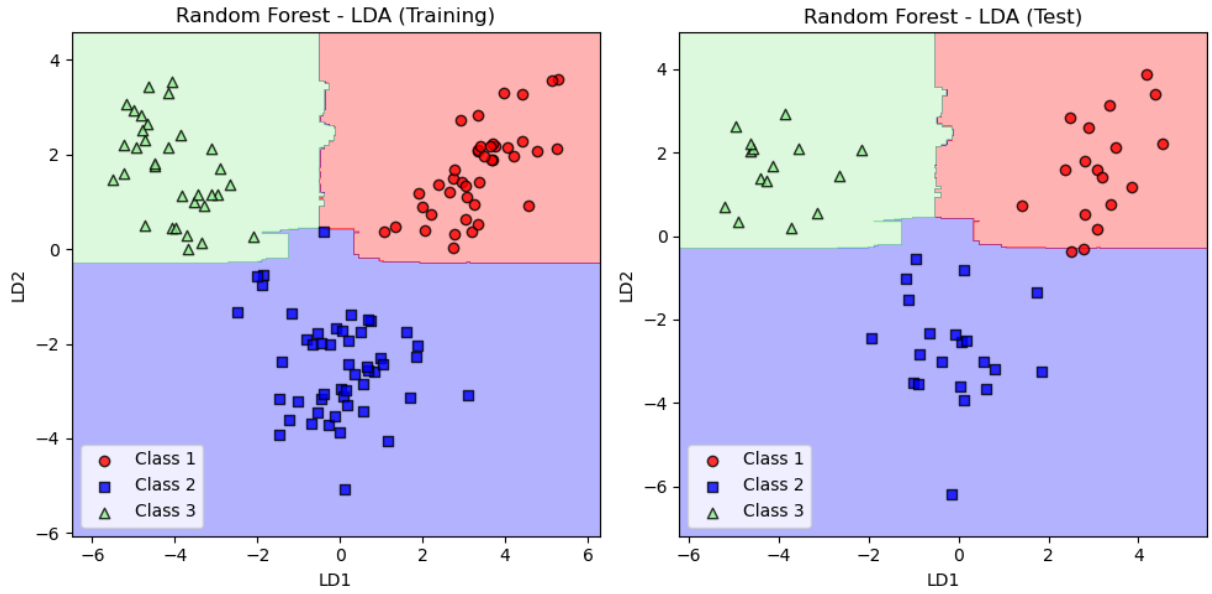





```
Processing Random Forest...
```

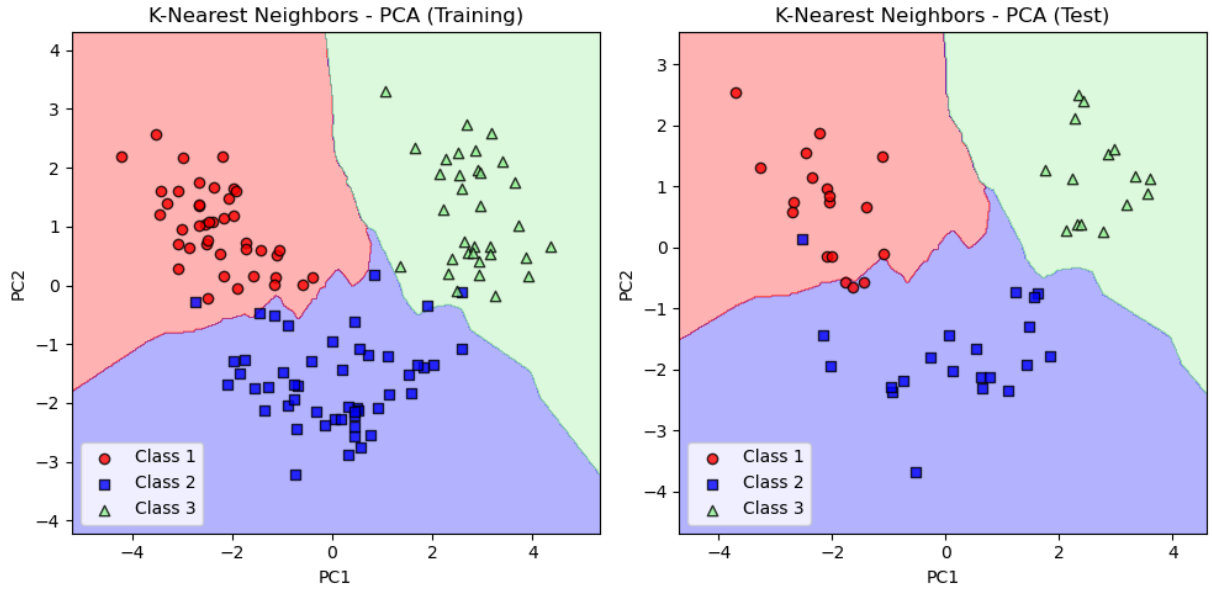






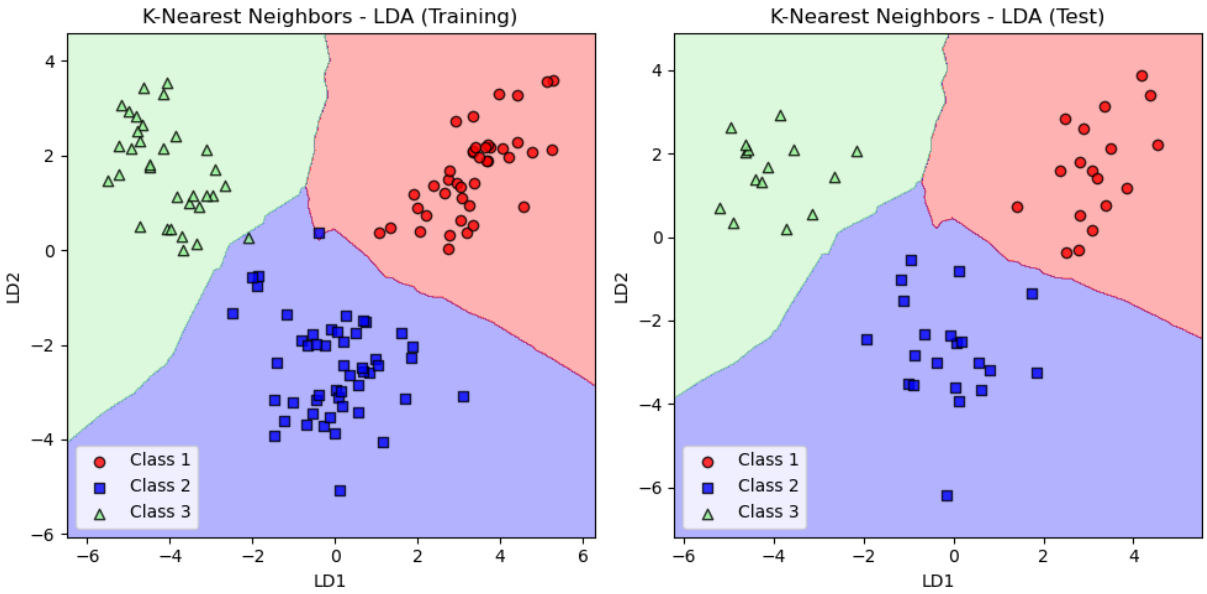

```
Processing K-Nearest Neighbors...
C:\Anaconda3\envs\pyml-book\lib\site-packages\sklearn\neighbors
\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `m
ode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` w
ill become False, the `axis` over which the statistic is taken
will be eliminated, and the value None will no longer be accept
ed. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Anaconda3\envs\pyml-book\lib\site-packages\sklearn\neighbors
\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `m
ode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` w
ill become False, the `axis` over which the statistic is taken
will be eliminated, and the value None will no longer be accept
ed. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Anaconda3\envs\pyml-book\lib\site-packages\sklearn\neighbors
\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `m
ode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` w
ill become False, the `axis` over which the statistic is taken
will be eliminated, and the value None will no longer be accept
ed. Set `keepdims` to True or False to avoid this warning.
```

```
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Anaconda3\envs\pyml-book\lib\site-packages\sklearn\neighbors
\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `m
ode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` w
ill become False, the `axis` over which the statistic is taken
will be eliminated, and the value None will no longer be accept
ed. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```




```
C:\Anaconda3\envs\pyml-book\lib\site-packages\sklearn\neighbors
\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `m
ode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` w
ill become False, the `axis` over which the statistic is taken
will be eliminated, and the value None will no longer be accept
ed. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Anaconda3\envs\pyml-book\lib\site-packages\sklearn\neighbors
\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `m
ode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` w
ill become False, the `axis` over which the statistic is taken
will be eliminated, and the value None will no longer be accept
ed. Set `keepdims` to True or False to avoid this warning.
```

```
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Anaconda3\envs\pyml-book\lib\site-packages\sklearn\neighbors
\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `m
ode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` w
ill become False, the `axis` over which the statistic is taken
will be eliminated, and the value None will no longer be accept
ed. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Anaconda3\envs\pyml-book\lib\site-packages\sklearn\neighbors
\_classification.py:228: FutureWarning: Unlike other reduction
functions (e.g. `skew`, `kurtosis`), the default behavior of `m
ode` typically preserves the axis it acts along. In SciPy 1.11.
0, this behavior will change: the default value of `keepdims` w
ill become False, the `axis` over which the statistic is taken
will be eliminated, and the value None will no longer be accept
ed. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```




```
...  # 打印結果表格
    print("\nResults Table:")
    print("="*80)
    print("{:<20} {:<20} {:<20} {:<20}".format("Algorithm", "
    PCA (Train/Test)", "LDA (Train/Test)", "Difference
    (Test)"))
    print("="*80)
    for name in classifiers.keys():
```

```
        pca_train = results['PCA'][name]['train_acc']
        pca_test = results['PCA'][name]['test_acc']
        lda_train = results['LDA'][name]['train_acc']
        lda_test = results['LDA'][name]['test_acc']
        diff = lda_test - pca_test

        print("{:<25}{:.3f}/{:.3f} {:<8} {:.3f}/{:.3f} {:<8}
    {:.3f}".format(
            name,
            pca_train, pca_test, "",
            lda_train, lda_test, "",
            diff
        ))
    print("="*80)
Results Table:
================================================================
=================
Algorithm              PCA (Train/Test)   LDA (Train/Test)
Difference (Test)
================================================================
=================
Perceptron               0.976/0.926         0.992/1.000
0.074
Logistic Regression      0.984/0.926         1.000/1.000
0.074
Support Vector Machine   0.976/0.926         0.992/1.000
0.074
Decision Tree            1.000/0.926         1.000/0.963
0.037
Random Forest            1.000/0.926         1.000/0.963
0.037
K-Nearest Neighbors      0.976/0.926         0.984/1.000
0.074
================================================================
=================
```

In [ ]: