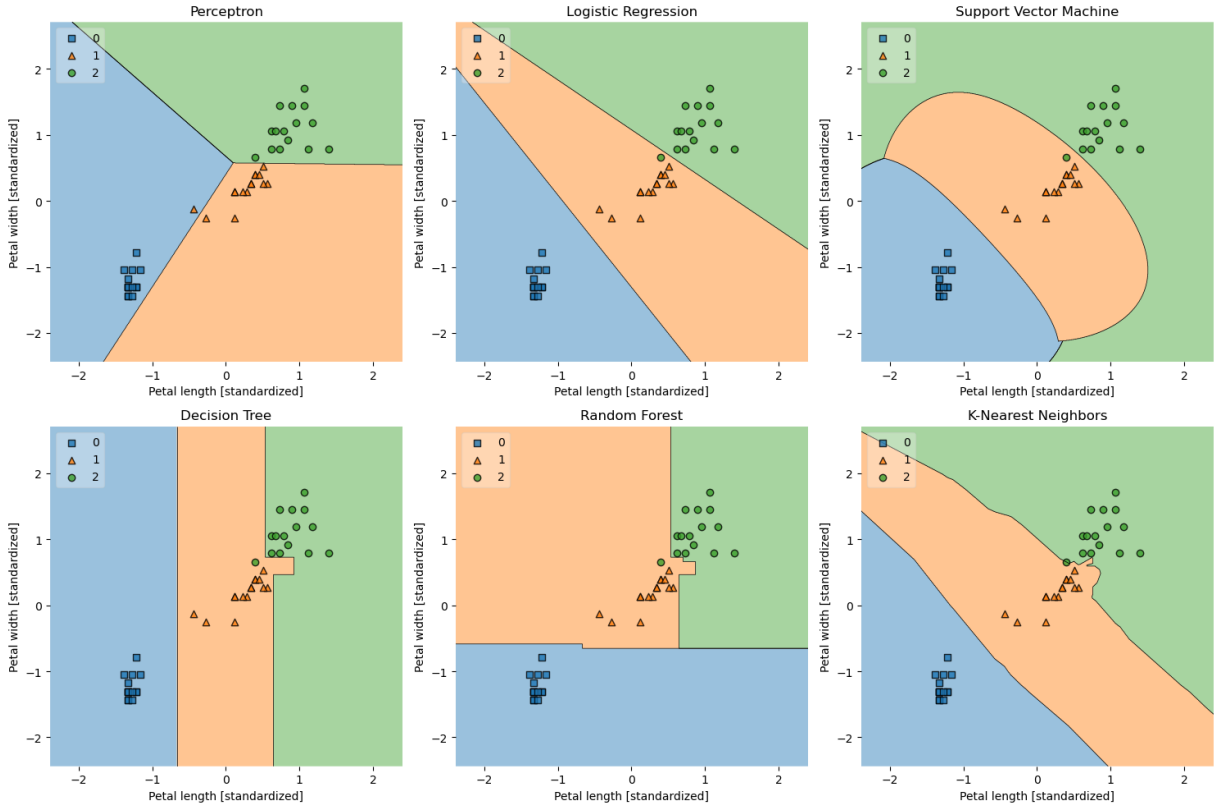
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Perceptron,
LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from mlxtend.plotting import plot_decision_regions

# 載入資料集
iris = load_iris()
X = iris.data[:, [2, 3]]  # 選擇花瓣長度與花瓣寬度
y = iris.target

# 分割資料集
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=1, stratify=y)

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

# 定義演算法
classifiers = {
    "Perceptron": Perceptron(max_iter=100, eta0=0.05,
random_state=1),
    "Logistic Regression": LogisticRegression(C=0.5,
random_state=1, solver='lbfgs', max_iter=300),
    "Support Vector Machine": SVC(),
    "Decision Tree": DecisionTreeClassifier(),
    "Random Forest": RandomForestClassifier(),
    "K-Nearest Neighbors": KNeighborsClassifier()
}

# 訓練模型並評估
results = []
```

```
    plt.figure(figsize=(15, 10))  # 設置總體圖形大小

    for idx, (name, clf) in enumerate(classifiers.items()):
        clf.fit(X_train_std, y_train)
        y_train_pred = clf.predict(X_train_std)
        y_test_pred = clf.predict(X_test_std)
        train_accuracy = accuracy_score(y_train, y_train_pred)
        test_accuracy = accuracy_score(y_test, y_test_pred)
        results.append((name, train_accuracy, test_accuracy))

        # 繪製決策邊界
        plt.subplot(2, 3, idx + 1)  # 在一個畫布上畫6個子圖
        plot_decision_regions(X_test_std, y_test, clf=clf,
    legend=2)
        plt.title(name)
        plt.xlabel('Petal length [standardized]')
        plt.ylabel('Petal width [standardized]')

    plt.tight_layout()
    plt.show()


     # 打印表格頭部
    print("{:<25} {:<15} {:<15}".format("Algorithm", "訓練資料正
    確率", "測試資料正確率"))
    print("-" * 61)

    # 打印結果
    for name, train_acc, test_acc in results:
        print("{:<25}     {:<15.2f}       {:
    <15.2f}".format(name, train_acc, test_acc))
```

```
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)
```



Algorithm	訓練資料正確率	測試資料正確率
Perceptron	0.94	0.98
Logistic Regression	0.94	0.98
Support Vector Machine	0.96	0.98
Decision Tree	0.99	0.98

Random Forest	0.99	0.98
K-Nearest Neighbors	0.97	1.00

In [ ]:

In [ ]: