自主練習

- 載入「葡萄酒數據集」，並將其分為70%訓練，30%測試(random_state =0)

```
from sklearn.datasets import load_wine
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

wine = load_wine()

X, y = wine.data, wine.target
print('Class labels:', np.unique(y))

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
                                                    random_state=0)
```

```
Class labels: [0 1 2]
```

自主練習

- 將數據標準化

```
sc = StandardScaler()
sc.fit(X_train)
X_train_std = sc.transform(X_train)
X_test_std = sc.transform(X_test)
```

自主練習

- PCA 降維 (k = 2)
- 利用邏輯回歸(logistic regression)做分類並對訓練集畫出決策區域圖

```
# PCA
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression

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

lr = LogisticRegression()
lr.fit(X_train_pca, y_train)



plot_decision_regions(X_train_pca, y_train, classifier=lr)
#plt.savefig('images.png', dpi=300)
lr.score(X_train_pca, y_train)
```

自主練習

- LDA 降維 (k = 2)
- 利用邏輯回歸(logistic regression)做分類並對訓練集畫出決策區域圖

```
# LDA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA

lda = LDA(n_components=2)
X_train_lda = lda.fit_transform(X_train_std, y_train)

lr = LogisticRegression()
lr = lr.fit(X_train_lda, y_train)

plot_decision_regions(X_train_lda, y_train, classifier=lr)
plt.xlabel('LD 1')
plt.ylabel('LD 2')
plt.legend(loc='lower left')

lr.score(X_train_lda, y_train)
```