資料科學 Report of Test 2

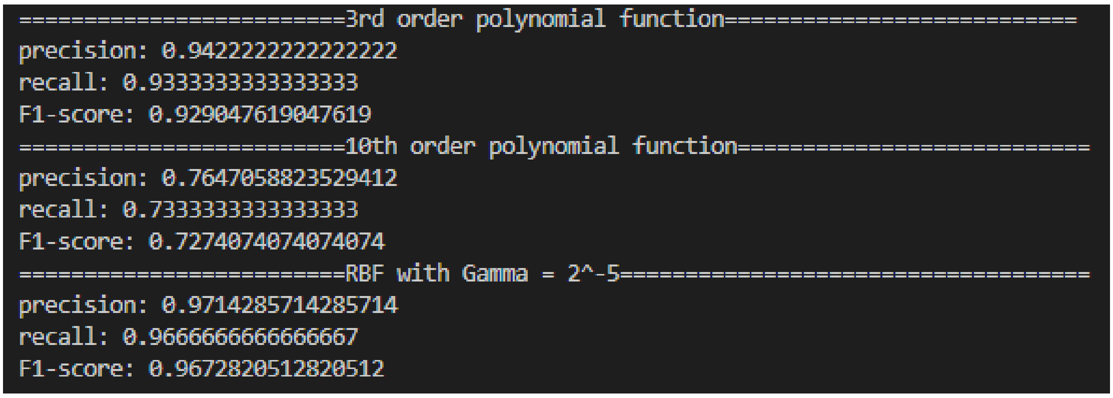
1106022 陳柏嘉

Code 與註解：

```
#%%
import numpy as np
import matplotlib.pyplot as plt
from mlxtend.plotting import plot_decision_regions
import pandas as pd
from sklearn.metrics import confusion_matrix, precision_score,
recall_score, f1_score, accuracy_score
from sklearn import svm
from sklearn import datasets
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split

iris=datasets.load_iris()
X=iris.data
y=iris.target

pca2 = PCA(n_components = 2) # 主成分分析，保留前 2 個 eigenvectors
X_2 = pca2.fit_transform(X) # 將 X 降低為 2 維數據
# 拆分訓練與測試資料，train : test = 80% : 20%
X_train, X_test, y_train, y_test = train_test_split(X_2,
y,test_size=0.2,random_state=0)

#%%
print('=========================3rd order polynomial
function===========================')

clf3=svm.SVC(kernel='poly', C=1, degree=3) # 建立模型，kernel 選用
degree=3 的 polynomial
clf3.fit(X_train,y_train)

y_pred_3 = clf3.predict(X_test) # 測試資料放入模型預測

# Precision Score = TP / (FP + TP)
# Recall Score = TP / (FN + TP)
# F1 Score = 2* Precision Score * Recall Score/ (Precision Score +
Recall Score)

print('precision:', precision_score(y_test, y_pred_3,
average='weighted'))
print('recall:', recall_score(y_test, y_pred_3, average='weighted'))
print('F1-score:', f1_score(y_test, y_pred_3, average='weighted'))

plot_decision_regions(X_train, y_train, clf=clf3, legend=2) # 畫平面圖
plt.xlabel('1st eigenvector')
plt.ylabel('2nd eigenvector')
plt.title('SVM on Iris with 3rd order polynomial function')
plt.show()

#%%
print('=========================10th order polynomial
function===========================')

clf10=svm.SVC(kernel='poly', C=1, degree=10) # 建立模型，kernel 選用
degree=10 的 polynomial
clf10.fit(X_train,y_train)

y_pred_10 = clf10.predict(X_test)

print('precision:', precision_score(y_test, y_pred_10,
average='weighted'))
print('recall:', recall_score(y_test, y_pred_10, average='weighted'))
print('F1-score:', f1_score(y_test, y_pred_10, average='weighted'))

plot_decision_regions(X_train, y_train, clf=clf3, legend=2)
plt.xlabel('1st eigenvector')
plt.ylabel('2nd eigenvector')
plt.title('SVM on Iris with 10th order polynomial function')
plt.show()

#%%
print('=========================RBF with Gamma = 2^-
5====================================')
```

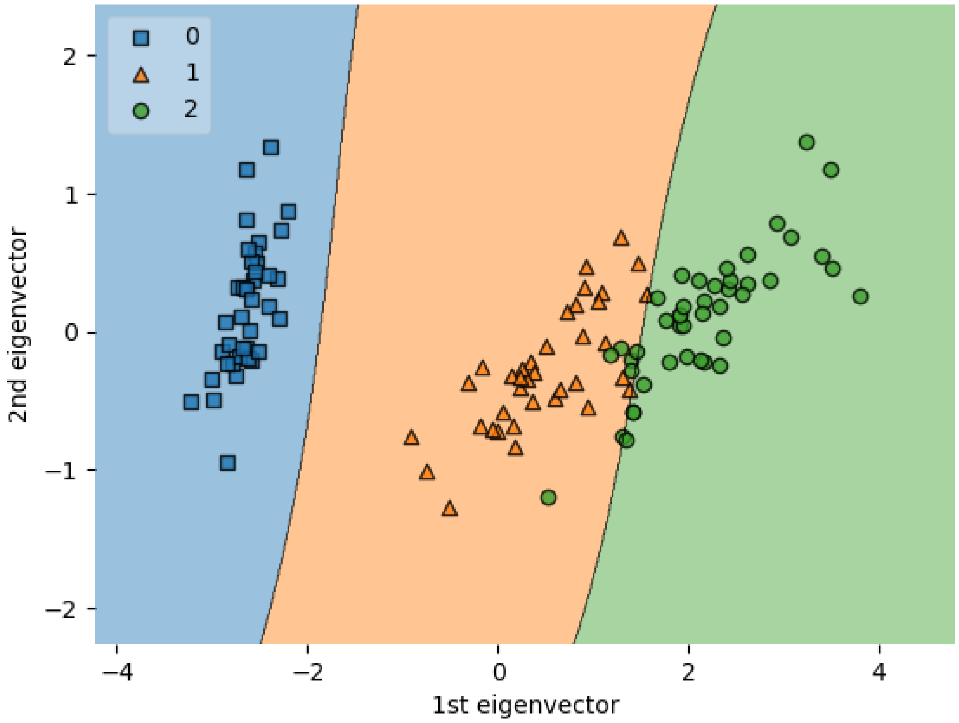
```
clfrbf=svm.SVC(kernel='rbf', C=1, gamma=2**-5) # 建立模型，kernel 選用
gamma=2^-5 的 rbf
clfrbf.fit(X_train,y_train)

y_pred_rbf = clfrbf.predict(X_test)

print('precision:', precision_score(y_test, y_pred_rbf,
average='weighted'))
print('recall:', recall_score(y_test, y_pred_rbf, average='weighted'))
print('F1-score:', f1_score(y_test, y_pred_rbf, average='weighted'))

plot_decision_regions(X_train, y_train, clf=clf3, legend=2)
plt.xlabel('1st eigenvector')
plt.ylabel('2nd eigenvector')
plt.title('SVM on Iris with RBF with Gamma = 2^-5')
plt.show()

# %%
```

輸出結果：

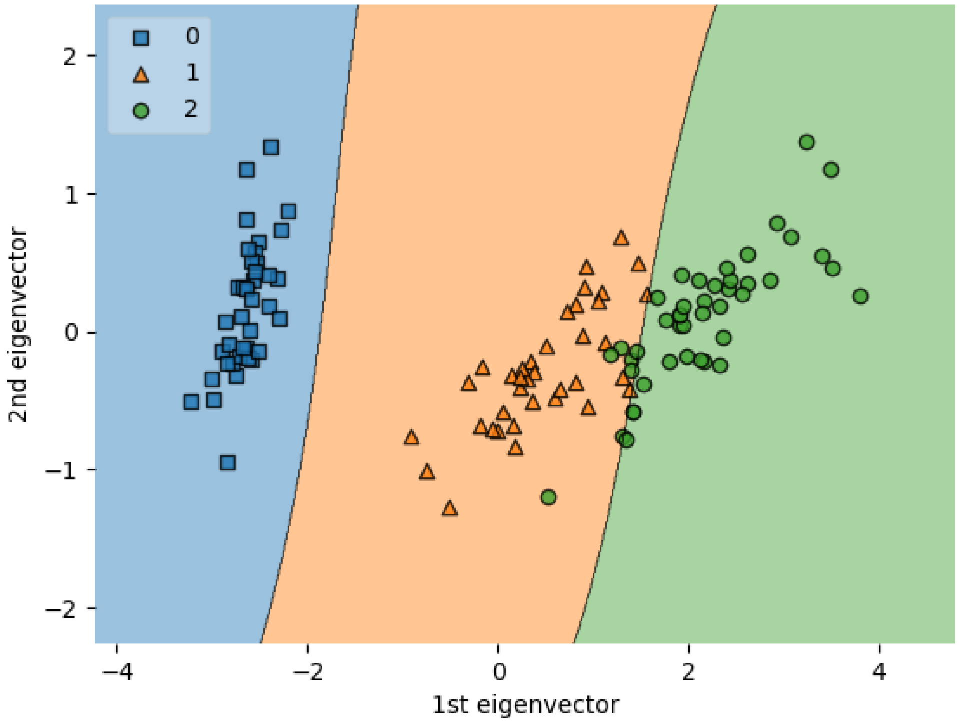


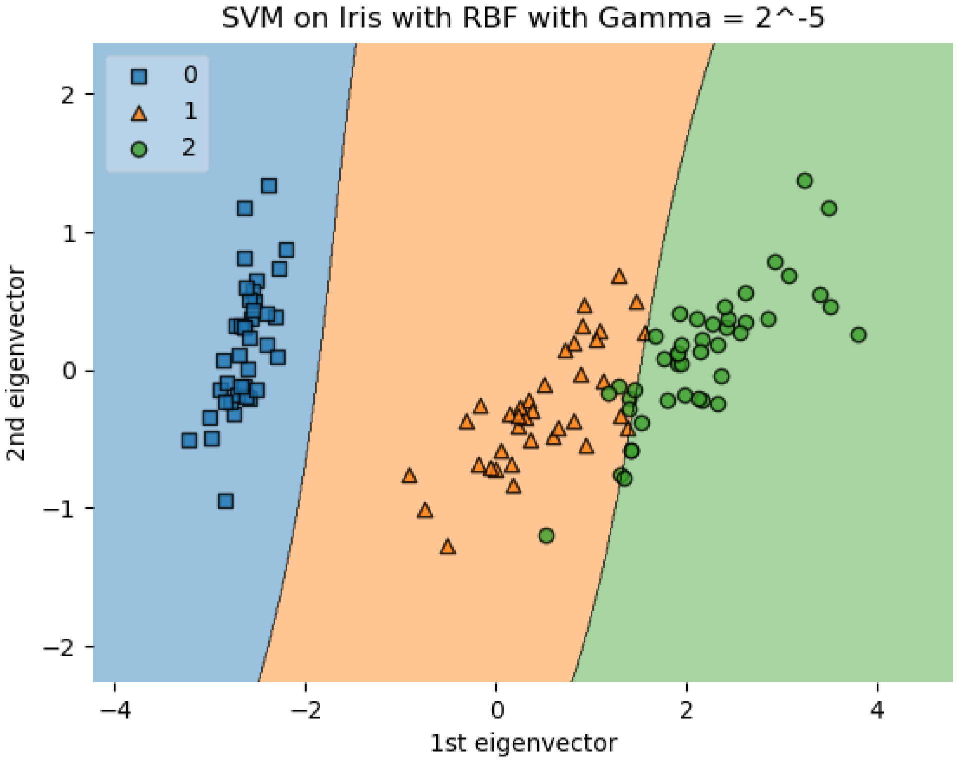
```
=========================3rd order polynomial function===========================
precision: 0.9422222222222222
recall: 0.9333333333333333
F1-score: 0.929047619047619
=========================10th order polynomial function===========================
precision: 0.7647058823529412
recall: 0.7333333333333333
F1-score: 0.7274074074074074
=========================RBF with Gamma = 2^-5====================================
precision: 0.9714285714285714
recall: 0.9666666666666667
F1-score: 0.9672820512820512
```

SVM on Iris with 3rd order polynomial function



SVM on Iris with 10th order polynomial function




SVM on Iris with RBF with Gamma = 2^-5
0
1
2
2
1
0
−1
−2
2nd eigenvector
−4
−2
0
2
4
1st eigenvector