2022 Machine Learning Theory and Practice

期末作業 06/13

姓名：蔡尚哲
學號：B10713138

期末作業提供兩個公開數據集，分別為

Dataset_2 - a dataset for heart attack classification

Dataset_1 - a dataset for diabetes classification

選擇一個資料集，根據這學期學習的各項機器學習知識以及範例程式，設定不同項目進行分類模型效能探討，例如：

1. 不同特徵擷取技術
2. 不同學習演算法
3. 學習演算法參數最佳化
4. 集成學習 - majority-voting classifier, bagging, boosting, …
5. …

注意:

訓練集以及測試集之分割為測試集佔全部數據集的25%，並且random_state設定為1，即

```
X_train, X_test, y_train, y_test = \

   train_test_split(X, y, test_size=0.25, random_state=1)
```

不論自訂對於哪些項目進行討論，最終一定要明確說出你所使用的是哪一個數據集

並且呈現針對上述測試集，你所找到的最佳分類模型，包含數據前處理、學習演算法及其參數設定，以及最終分類效能的指標(accuracy, precision, recal, andl confusion matrix)等結果

目錄

全部使用

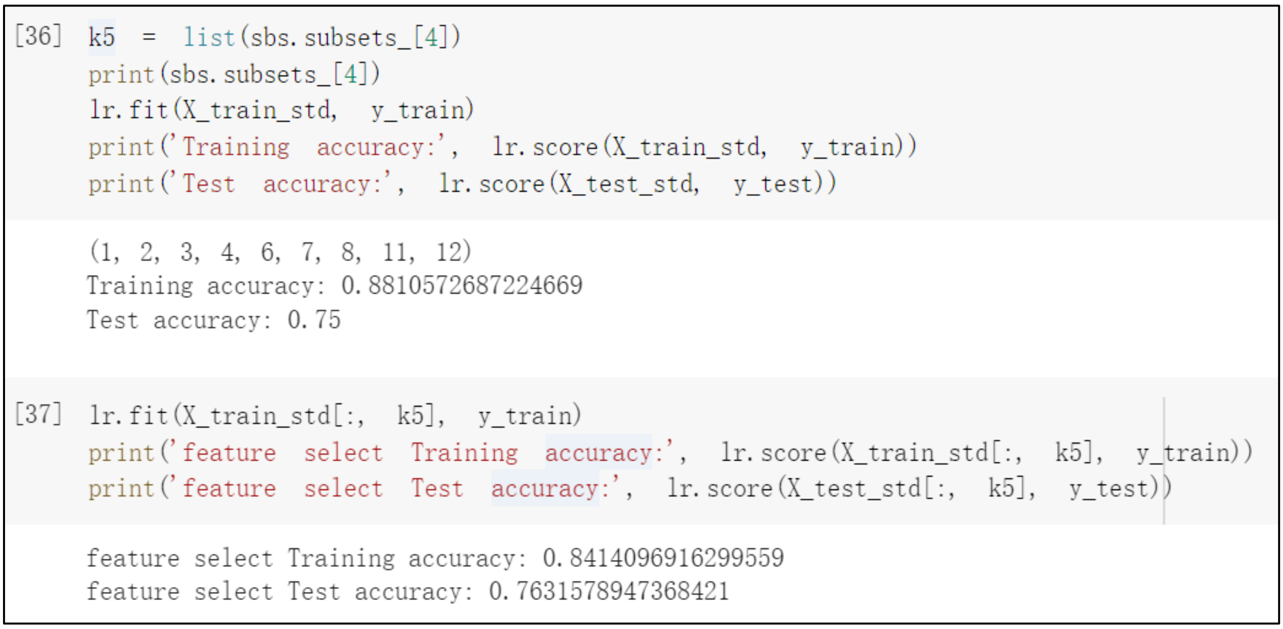
Dataset_2 - a dataset for heart attack classification 資料集

一.Logistic Regression

L1正規化，C=1

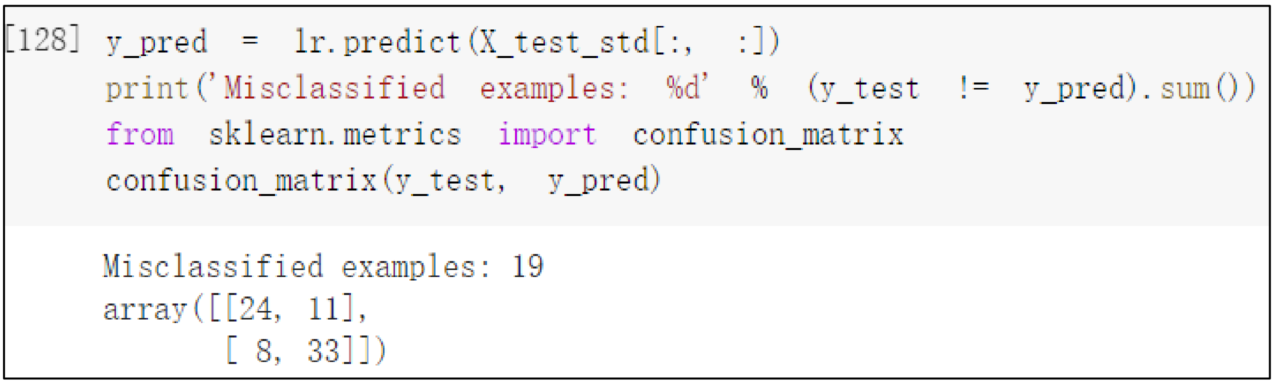
```
lr = LogisticRegression(penalty='l1',C=1,solver='liblinear')
# selecting features
sbs = SBS(lr, k_features=1)
sbs.fit(X_train_std, y_train)
```

正確度(上是所有特徵，下是特徵擷取9個特徵)

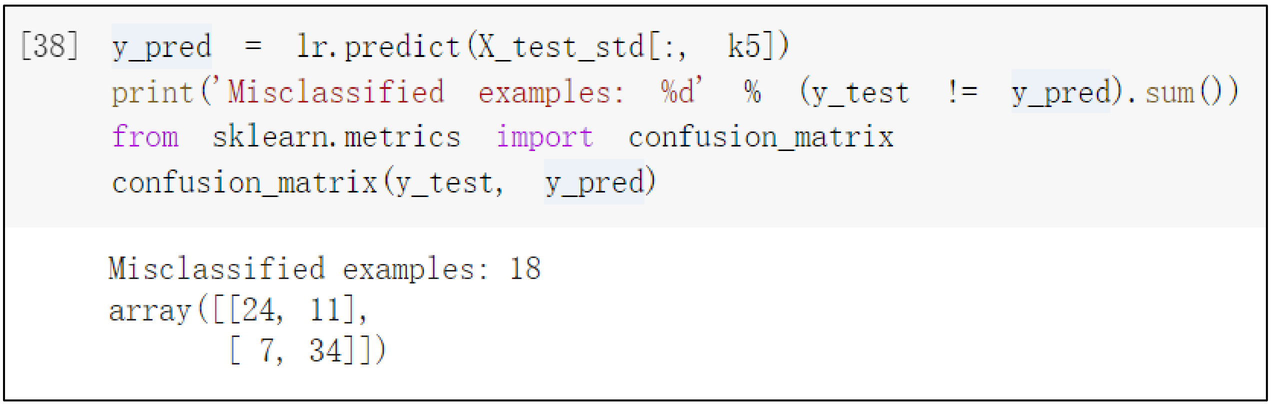


```
[36] k5 = list(sbs.subsets_[4])
     print(sbs.subsets_[4])
     lr.fit(X_train_std, y_train)
     print('Training accuracy:', lr.score(X_train_std, y_train))
     print('Test accuracy:', lr.score(X_test_std, y_test))

     (1, 2, 3, 4, 6, 7, 8, 11, 12)
     Training accuracy: 0.8810572687224669
     Test accuracy: 0.75

[37] lr.fit(X_train_std[:, k5], y_train)
     print('feature select Training accuracy:', lr.score(X_train_std[:, k5], y_train))
     print('feature select Test accuracy:', lr.score(X_test_std[:, k5], y_test))

     feature select Training accuracy: 0.8414096916299559
     feature select Test accuracy: 0.7631578947368421
```

confusion matrix(上是所有特徵，下是特徵擷取9個特徵)



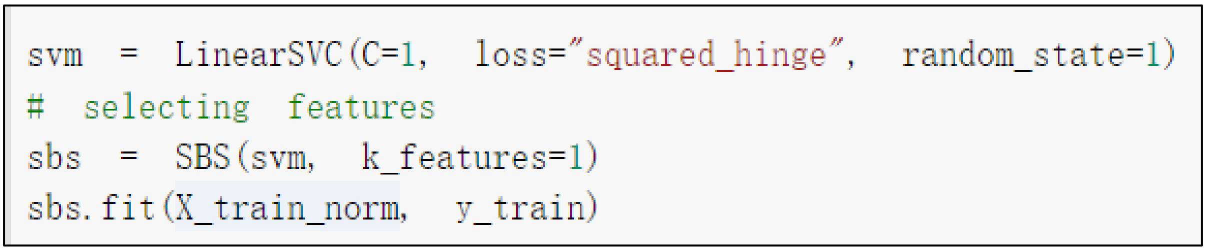
```
[128] y_pred = lr.predict(X_test_std[:, :])
      print('Misclassified examples: %d' % (y_test != y_pred).sum())
      from sklearn.metrics import confusion_matrix
      confusion_matrix(y_test, y_pred)

      Misclassified examples: 19
      array([[24, 11],
             [ 8, 33]])
```



```
[38]  y_pred = lr.predict(X_test_std[:, k5])
      print('Misclassified examples: %d' % (y_test != y_pred).sum())
      from sklearn.metrics import confusion_matrix
      confusion_matrix(y_test, y_pred)

      Misclassified examples: 18
      array([[24, 11],
             [ 7, 34]])
```

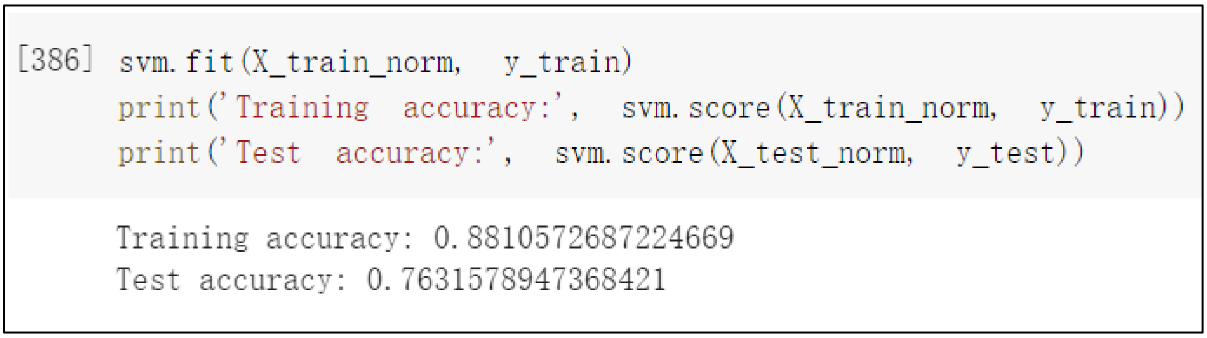
二.Linear SVM

C=1

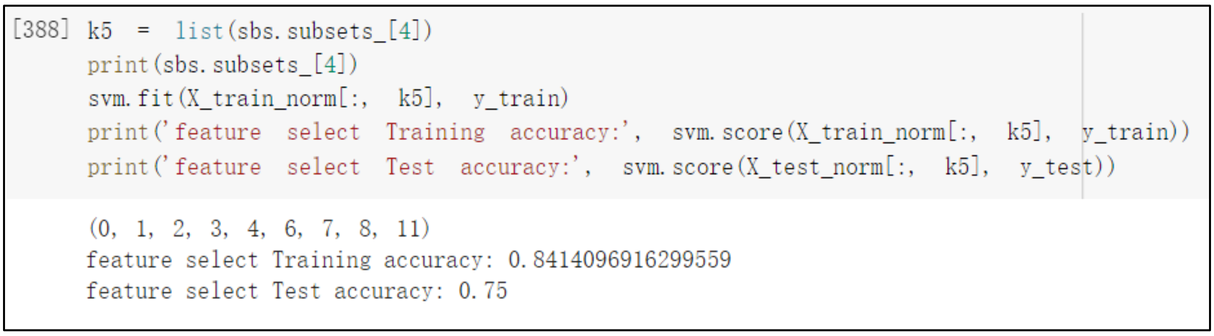


```
svm = LinearSVC(C=1, loss="squared_hinge", random_state=1)
# selecting features
sbs = SBS(svm, k_features=1)
sbs.fit(X_train_norm, y_train)
```

正確度(上是所有特徵，下是特徵擷取9個特徵)

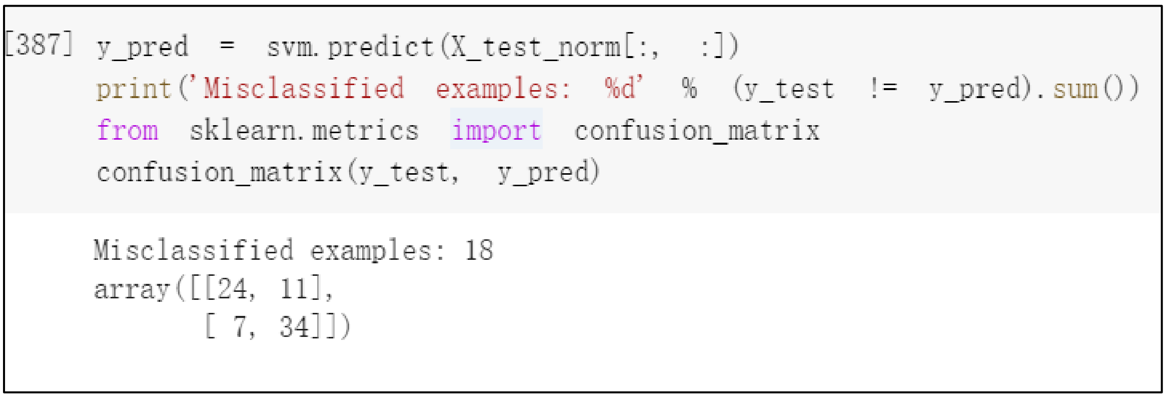


```
[386] svm.fit(X_train_norm, y_train)
      print('Training accuracy:', svm.score(X_train_norm, y_train))
      print('Test accuracy:', svm.score(X_test_norm, y_test))

      Training accuracy: 0.8810572687224669
      Test accuracy: 0.7631578947368421
```

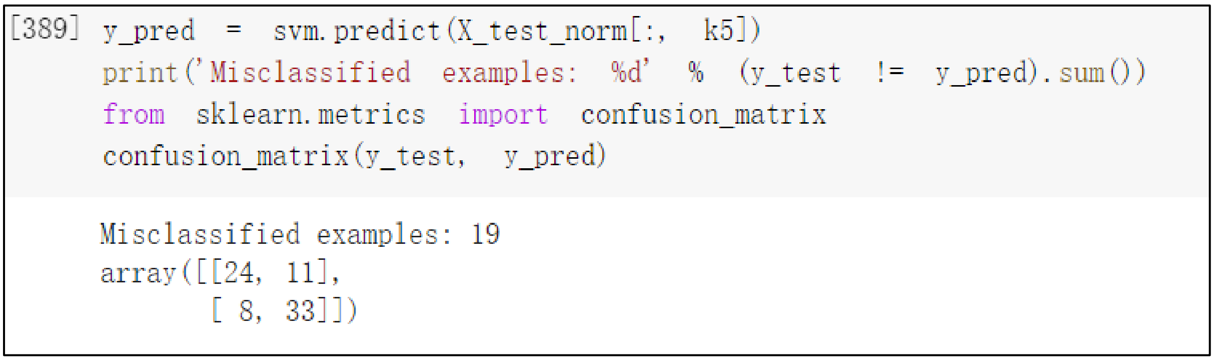


```
[388] k5 = list(sbs.subsets_[4])
      print(sbs.subsets_[4])
      svm.fit(X_train_norm[:, k5], y_train)
      print('feature select Training accuracy:', svm.score(X_train_norm[:, k5], y_train))
      print('feature select Test accuracy:', svm.score(X_test_norm[:, k5], y_test))

      (0, 1, 2, 3, 4, 6, 7, 8, 11)
      feature select Training accuracy: 0.8414096916299559
      feature select Test accuracy: 0.75
```

confusion matrix(上是所有特徵，下是特徵擷取9個特徵)



```
[387] y_pred = svm.predict(X_test_norm[:, :])
      print('Misclassified examples: %d' % (y_test != y_pred).sum())
      from sklearn.metrics import confusion_matrix
      confusion_matrix(y_test, y_pred)

      Misclassified examples: 18
      array([[24, 11],
             [ 7, 34]])
```

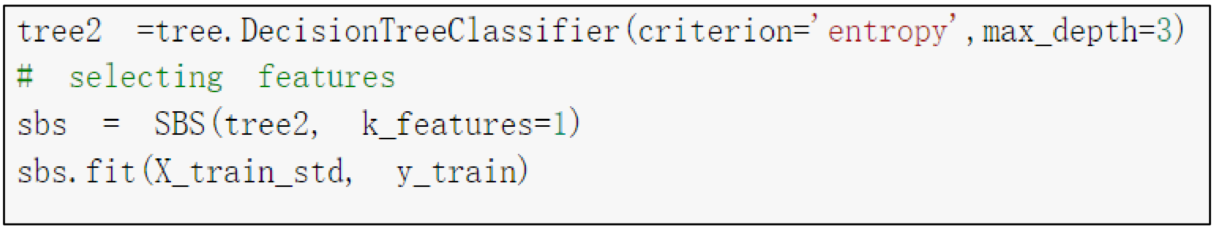


```
[389] y_pred  =  svm.predict(X_test_norm[:,  k5])
      print(’Misclassified  examples:  %d’  %  (y_test  !=  y_pred).sum())
      from  sklearn.metrics  import  confusion_matrix
      confusion_matrix(y_test,  y_pred)
```

```
Misclassified examples: 19
array([[24,  11],
       [ 8,  33]])
```

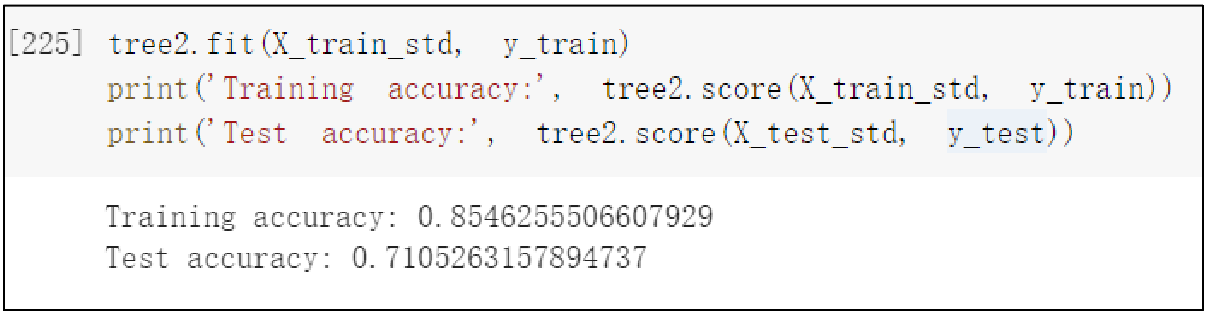
三.Decision Tree

entropy，最多3層

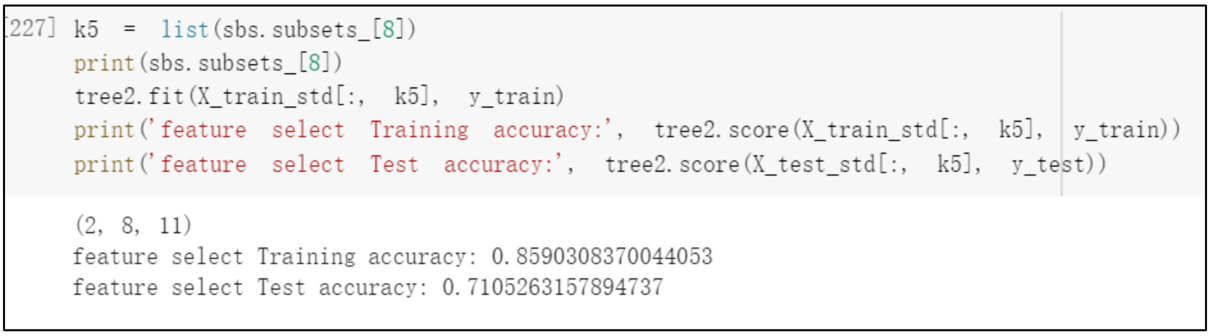


```
tree2 =tree.DecisionTreeClassifier(criterion=’entropy’,max_depth=3)
# selecting features
sbs = SBS(tree2, k_features=1)
sbs.fit(X_train_std, y_train)
```

正確度(上是所有特徵，下是特徵擷取3個特徵)

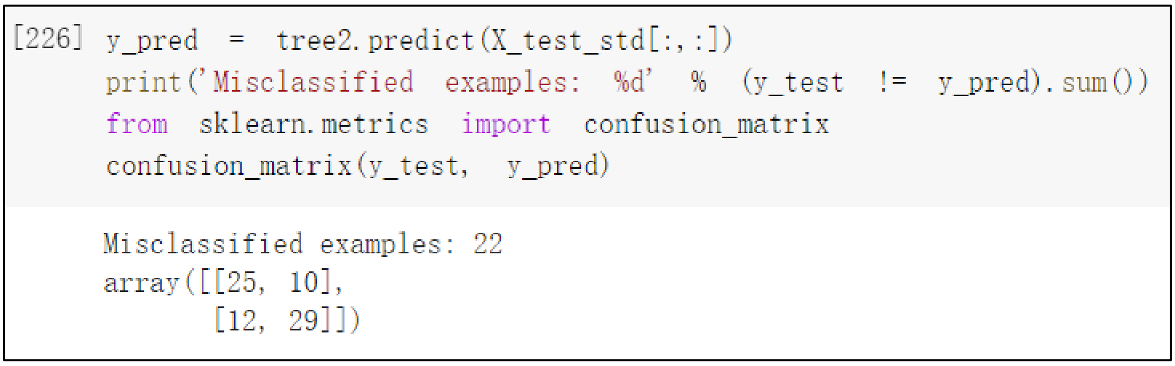


```
[225] tree2.fit(X_train_std, y_train)
      print(’Training accuracy:’, tree2.score(X_train_std, y_train))
      print(’Test accuracy:’, tree2.score(X_test_std, y_test))

      Training accuracy: 0.8546255506607929
      Test accuracy: 0.7105263157894737
```



```
227] k5 = list(sbs.subsets_[8])
     print(sbs.subsets_[8])
     tree2.fit(X_train_std[:, k5], y_train)
     print(’feature select Training accuracy:’, tree2.score(X_train_std[:, k5], y_train))
     print(’feature select Test accuracy:’, tree2.score(X_test_std[:, k5], y_test))

     (2, 8, 11)
     feature select Training accuracy: 0.8590308370044053
     feature select Test accuracy: 0.7105263157894737
```

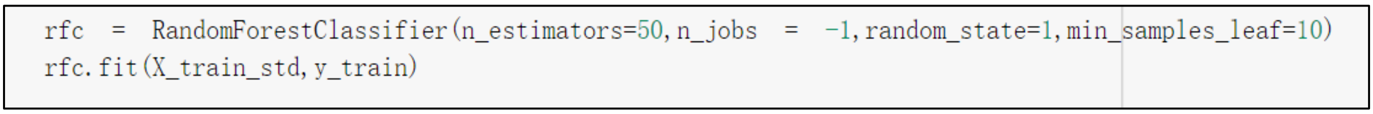
confusion matrix(一樣)



```
[226] y_pred = tree2.predict(X_test_std[:,:])
      print(’Misclassified examples: %d’ % (y_test != y_pred).sum())
      from sklearn.metrics import confusion_matrix
      confusion_matrix(y_test, y_pred)

      Misclassified examples: 22
      array([[25, 10],
             [12, 29]])
```

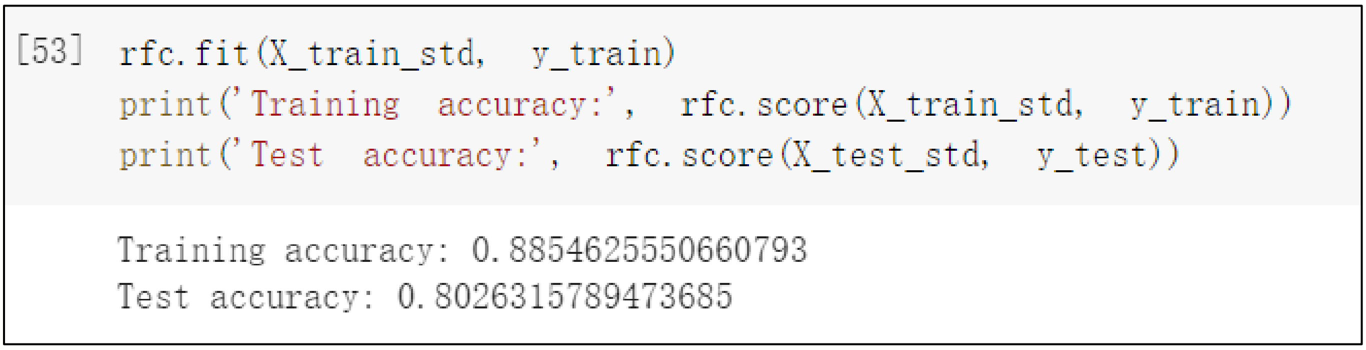
四.Random Forest

n_estimators=50，一個leaf至少十個



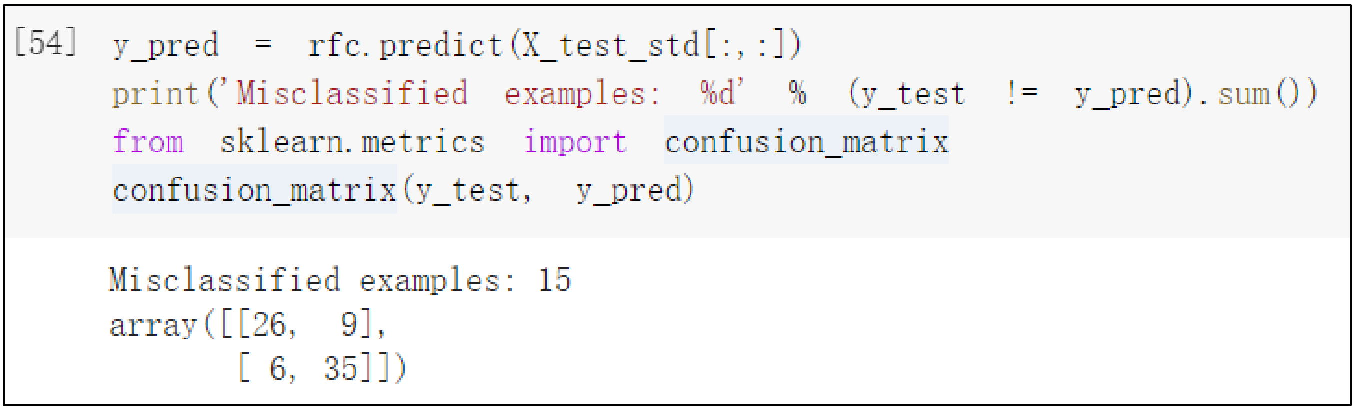
```
rfc = RandomForestClassifier(n_estimators=50,n_jobs = -1,random_state=1,min_samples_leaf=10)
rfc.fit(X_train_std,y_train)
```

正確度



```
[53] rfc.fit(X_train_std, y_train)
     print('Training accuracy:', rfc.score(X_train_std, y_train))
     print('Test accuracy:', rfc.score(X_test_std, y_test))

     Training accuracy: 0.8854625550660793
     Test accuracy: 0.8026315789473685
```

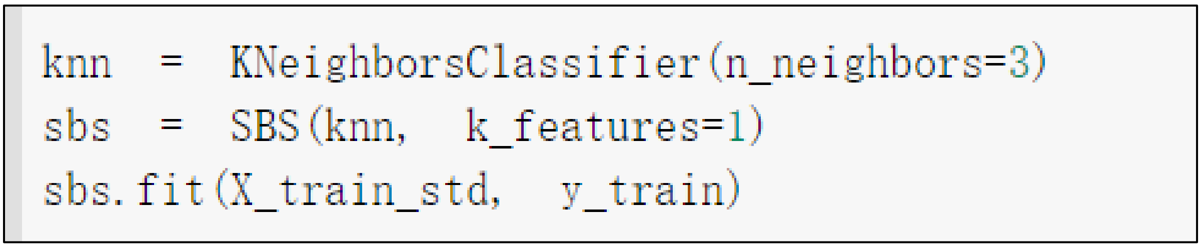
confusion matrix



```
[54] y_pred = rfc.predict(X_test_std[:,:])
     print('Misclassified examples: %d' % (y_test != y_pred).sum())
     from sklearn.metrics import confusion_matrix
     confusion_matrix(y_test, y_pred)

     Misclassified examples: 15
     array([[26,  9],
            [ 6, 35]])
```

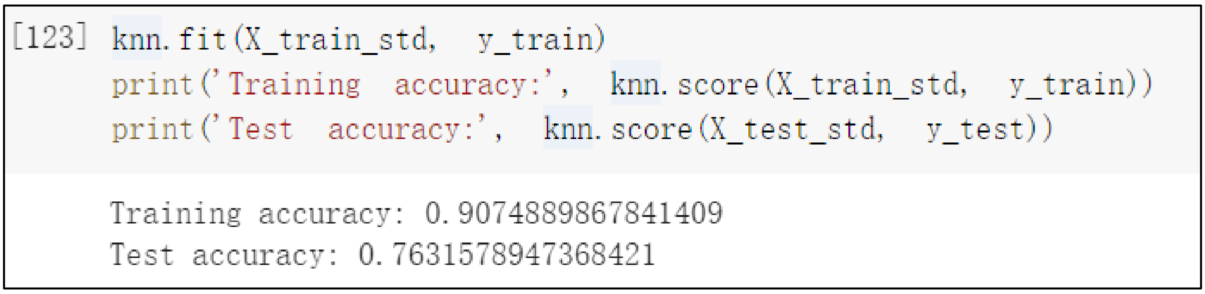
五. knn

n_neighbor=3

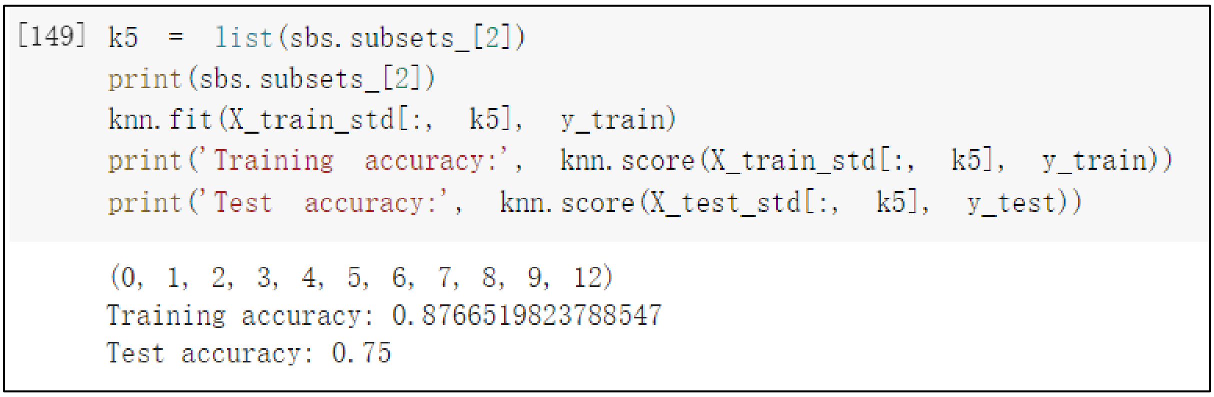


```
knn = KNeighborsClassifier(n_neighbors=3)
sbs = SBS(knn, k_features=1)
sbs.fit(X_train_std, y_train)
```

正確度(上是所有特徵，下是特徵擷取11個特徵)

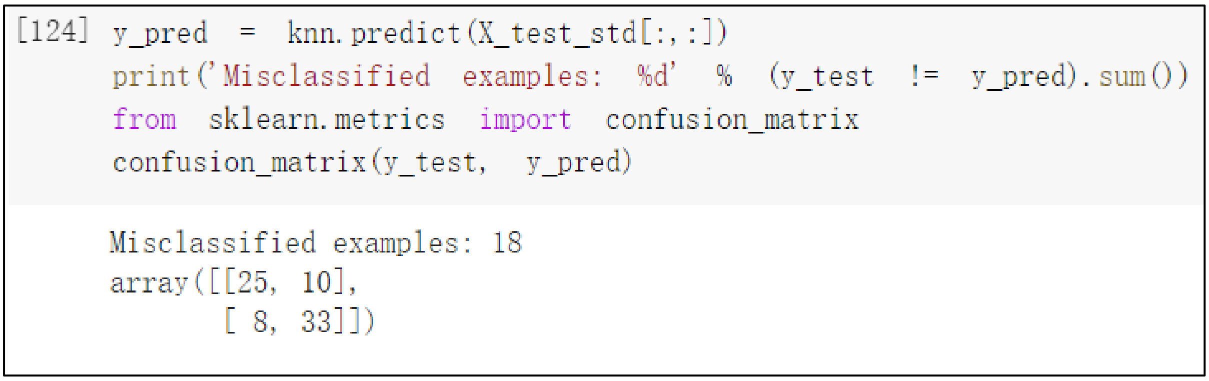


```
[123] knn.fit(X_train_std, y_train)
      print('Training accuracy:', knn.score(X_train_std, y_train))
      print('Test accuracy:', knn.score(X_test_std, y_test))

      Training accuracy: 0.9074889867841409
      Test accuracy: 0.7631578947368421
```

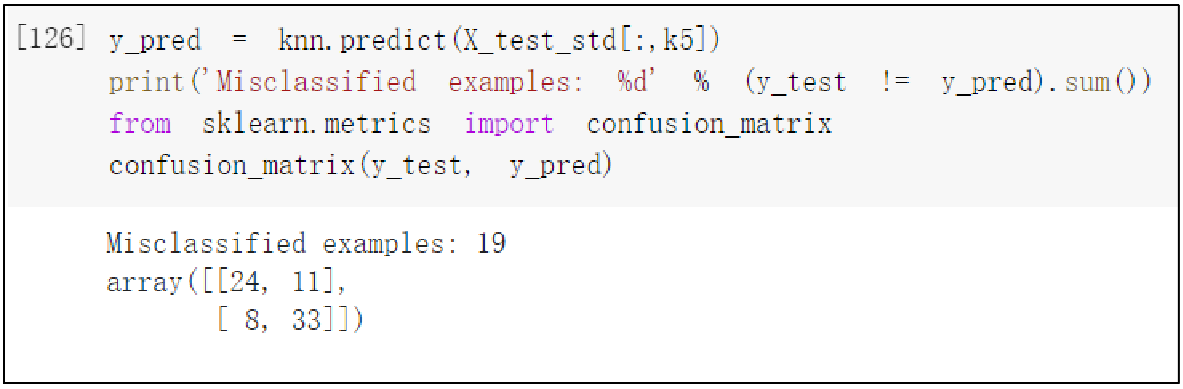


```
[149] k5 = list(sbs.subsets_[2])
      print(sbs.subsets_[2])
      knn.fit(X_train_std[:, k5], y_train)
      print('Training accuracy:', knn.score(X_train_std[:, k5], y_train))
      print('Test accuracy:', knn.score(X_test_std[:, k5], y_test))

      (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 12)
      Training accuracy: 0.8766519823788547
      Test accuracy: 0.75
```

confusion matrix(上是所有特徵，下是特徵擷取5個特徵)



```
[124] y_pred = knn.predict(X_test_std[:,:])
      print('Misclassified examples: %d' % (y_test != y_pred).sum())
      from sklearn.metrics import confusion_matrix
      confusion_matrix(y_test, y_pred)

      Misclassified examples: 18
      array([[25, 10],
             [ 8, 33]])
```

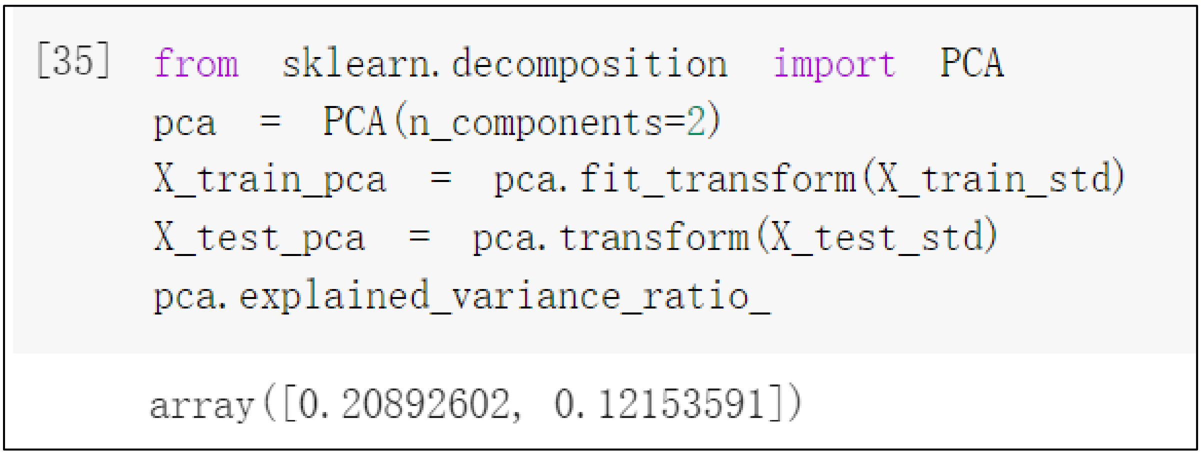


```
[126] y_pred = knn.predict(X_test_std[:,k5])
      print(’Misclassified examples: %d’ % (y_test != y_pred).sum())
      from sklearn.metrics import confusion_matrix
      confusion_matrix(y_test, y_pred)
```

```
      Misclassified examples: 19
      array([[24, 11],
             [ 8, 33]])
```

六. PCA:

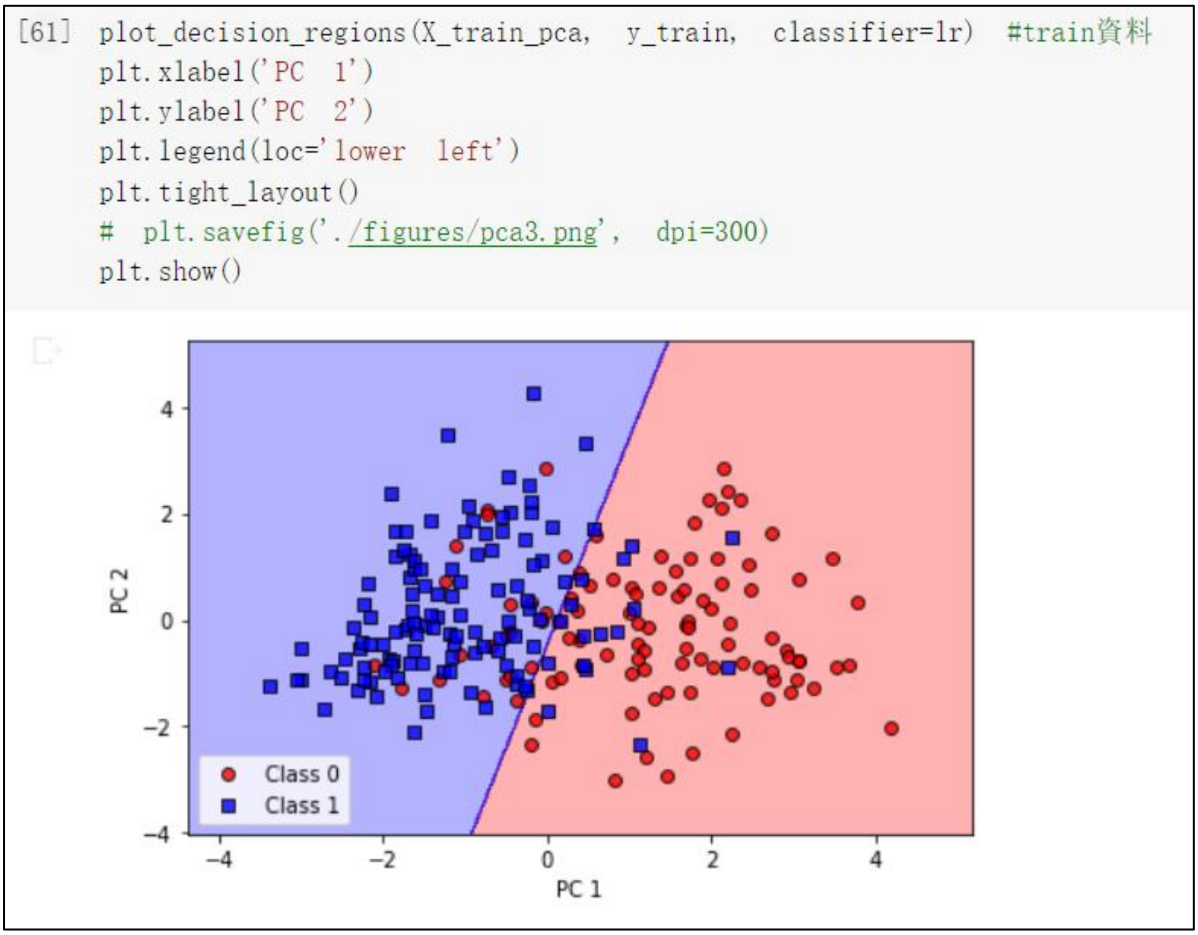
explained_variance_ratio(降到2個特徵)



```
[35] from sklearn.decomposition import PCA
     pca = PCA(n_components=2)
     X_train_pca = pca.fit_transform(X_train_std)
     X_test_pca = pca.transform(X_test_std)
     pca.explained_variance_ratio_
```

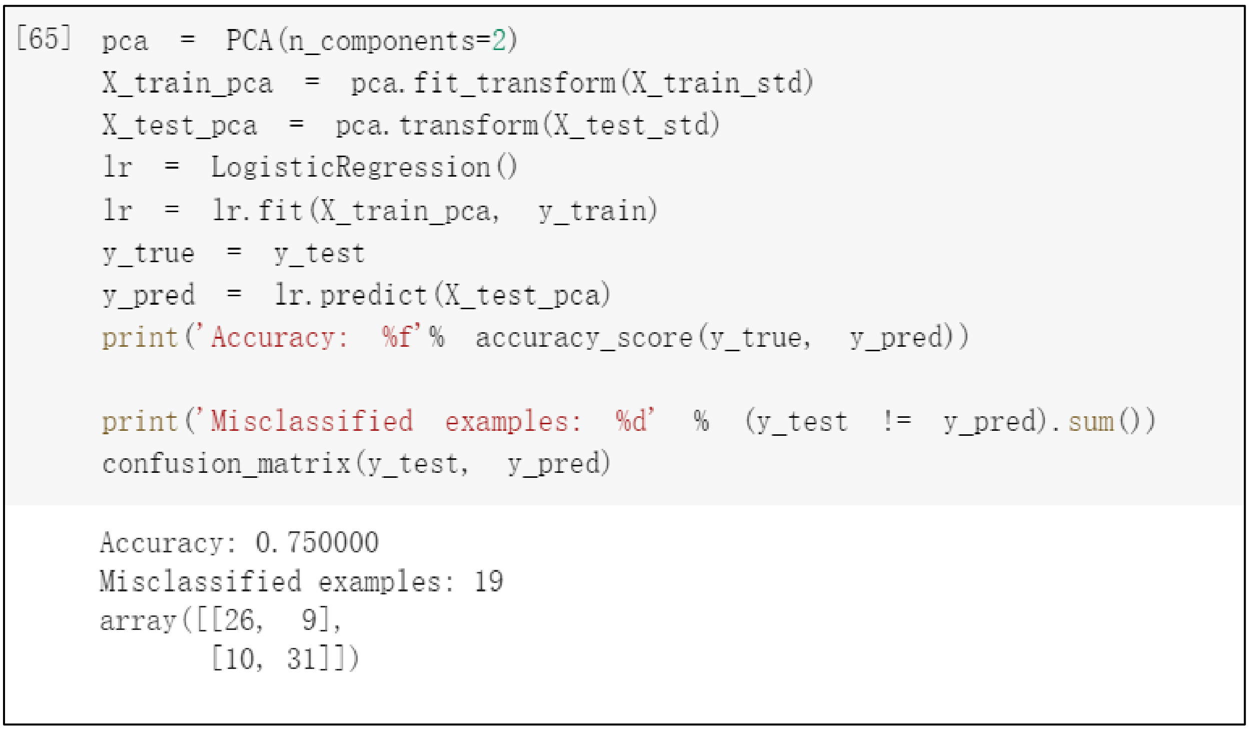
```
array([0.20892602, 0.12153591])
```

train data（降到2個特徵的分布圖）



```
[61] plot_decision_regions(X_train_pca, y_train, classifier=lr) #train資料
     plt.xlabel(’PC 1’)
     plt.ylabel(’PC 2’)
     plt.legend(loc=’lower left’)
     plt.tight_layout()
     # plt.savefig(’./figures/pca3.png’, dpi=300)
     plt.show()
```

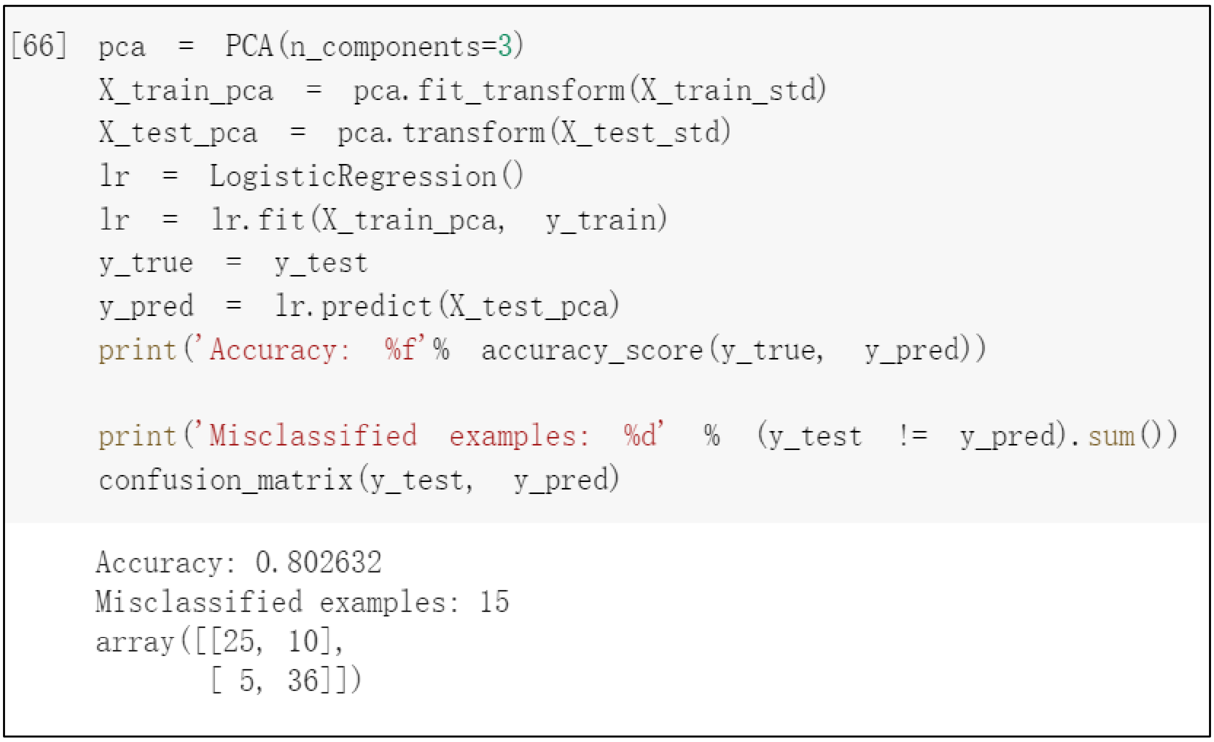
2個特徵正確率與confusion matrix(使用logistic regression，2個特徵)



```
[65] 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 = lr.fit(X_train_pca, y_train)
     y_true = y_test
     y_pred = lr.predict(X_test_pca)
     print('Accuracy: %f' % accuracy_score(y_true, y_pred))

     print('Misclassified examples: %d' % (y_test != y_pred).sum())
     confusion_matrix(y_test, y_pred)
```

```
     Accuracy: 0.750000
     Misclassified examples: 19
     array([[26,  9],
            [10, 31]])
```

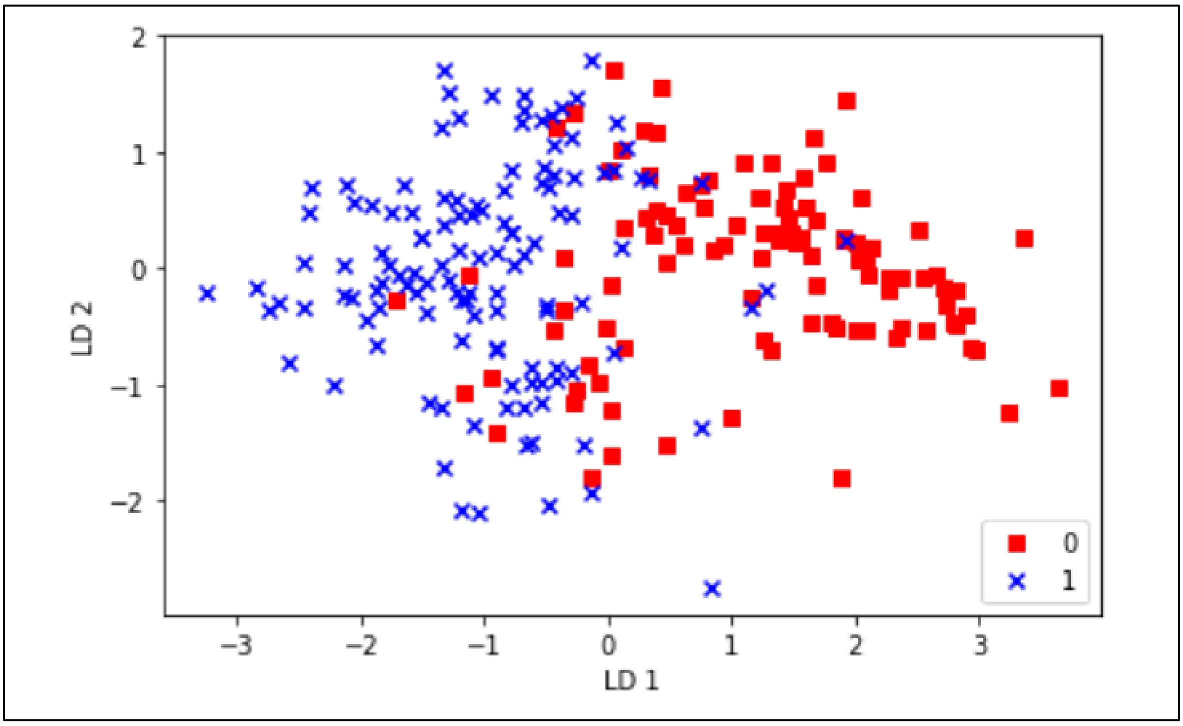
最終維度為3~8為最高正確率，其Accuracy與confusion matrix都一樣

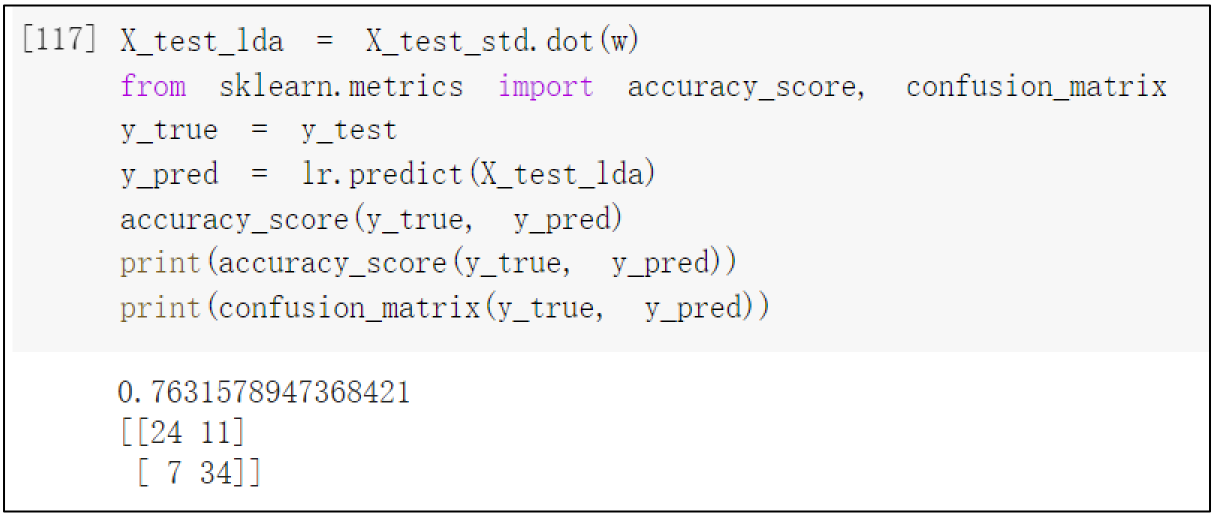


```
[66] pca = PCA(n_components=3)
     X_train_pca = pca.fit_transform(X_train_std)
     X_test_pca = pca.transform(X_test_std)
     lr = LogisticRegression()
     lr = lr.fit(X_train_pca, y_train)
     y_true = y_test
     y_pred = lr.predict(X_test_pca)
     print('Accuracy: %f' % accuracy_score(y_true, y_pred))

     print('Misclassified examples: %d' % (y_test != y_pred).sum())
     confusion_matrix(y_test, y_pred)
```

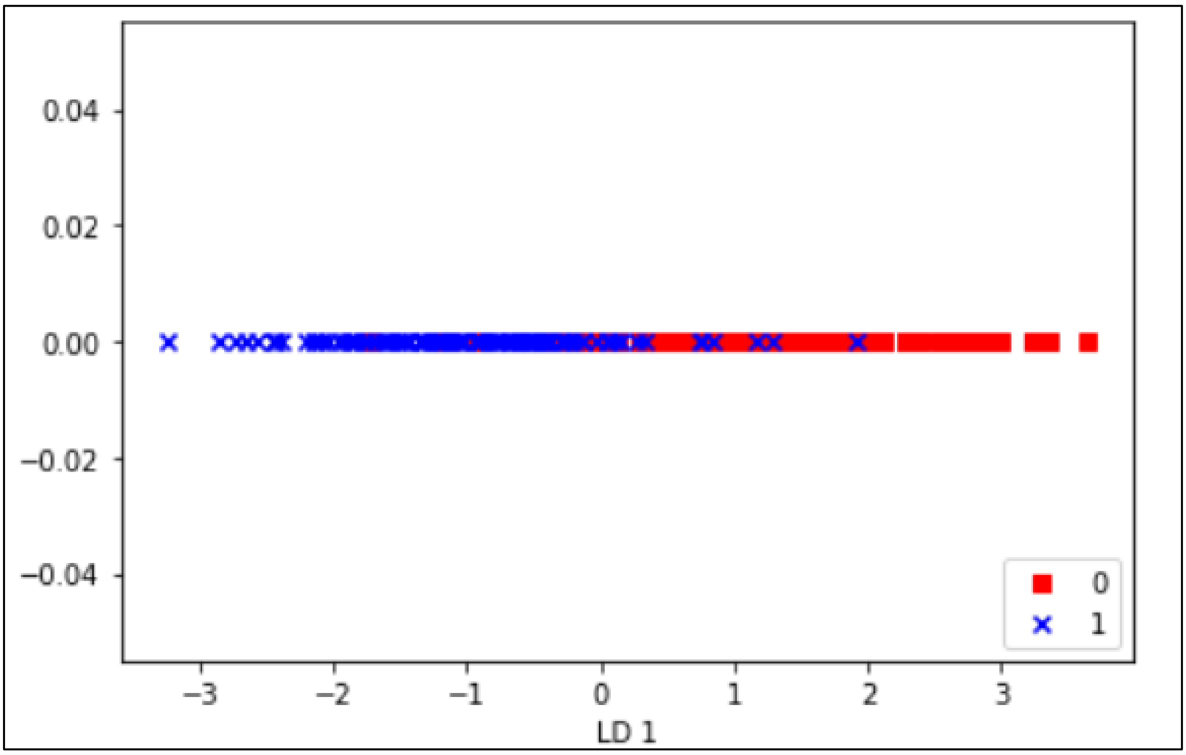
```
     Accuracy: 0.802632
     Misclassified examples: 15
     array([[25, 10],
            [ 5, 36]])
```

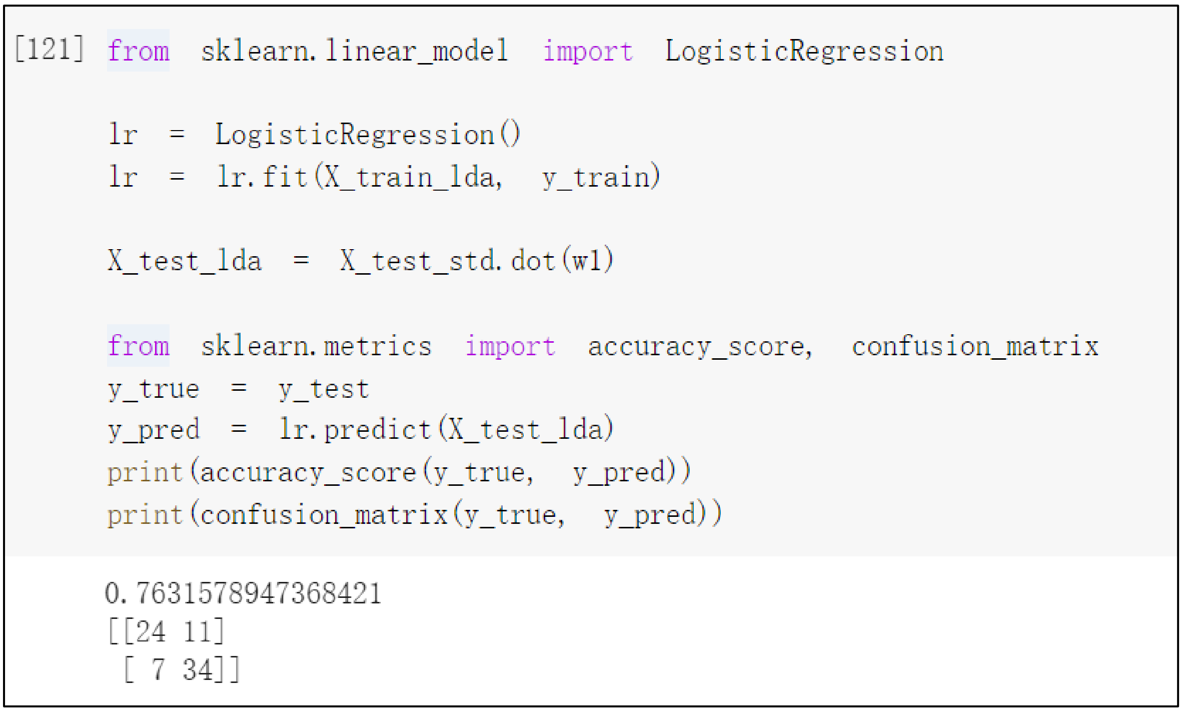
七.LDA:





```
[117] X_test_lda = X_test_std.dot(w)
      from sklearn.metrics import accuracy_score, confusion_matrix
      y_true = y_test
      y_pred = lr.predict(X_test_lda)
      accuracy_score(y_true, y_pred)
      print(accuracy_score(y_true, y_pred))
      print(confusion_matrix(y_true, y_pred))

      0.7631578947368421
      [[24 11]
       [ 7 34]]
```



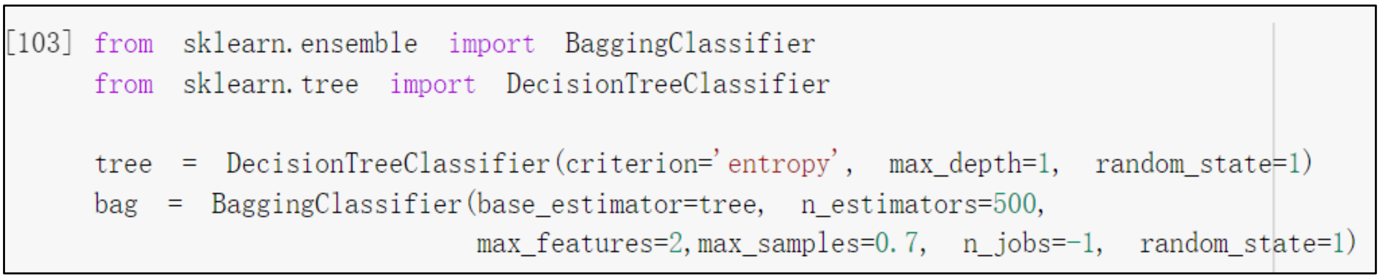


```
[121] from sklearn.linear_model import LogisticRegression

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

      X_test_lda = X_test_std.dot(w1)

      from sklearn.metrics import accuracy_score, confusion_matrix
      y_true = y_test
      y_pred = lr.predict(X_test_lda)
      print(accuracy_score(y_true, y_pred))
      print(confusion_matrix(y_true, y_pred))
```

```
      0.7631578947368421
      [[24 11]
       [ 7 34]]
```

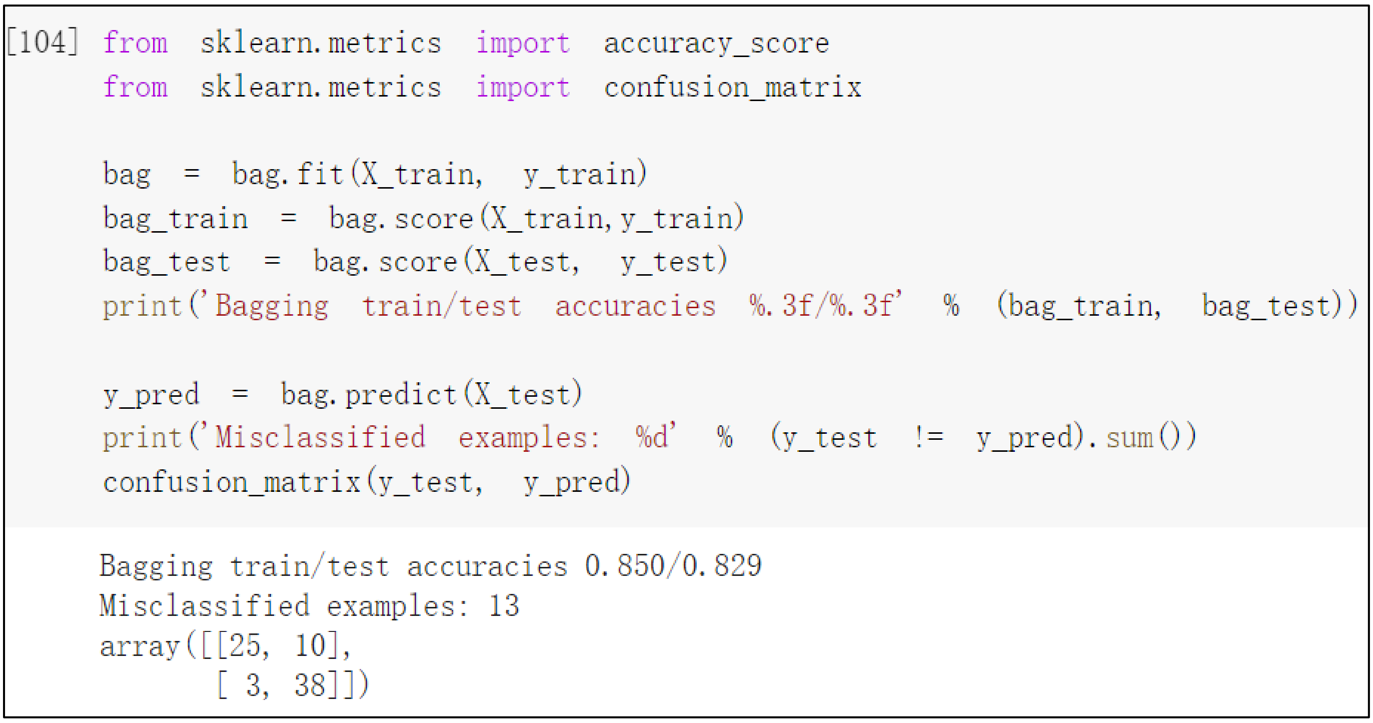
八. Bagging

n_estimator=500,max_features=2,max_samples=0.7



```
[103] from sklearn.ensemble import BaggingClassifier
      from sklearn.tree import DecisionTreeClassifier

      tree = DecisionTreeClassifier(criterion='entropy', max_depth=1, random_state=1)
      bag = BaggingClassifier(base_estimator=tree, n_estimators=500,
                              max_features=2,max_samples=0.7, n_jobs=-1, random_state=1)
```

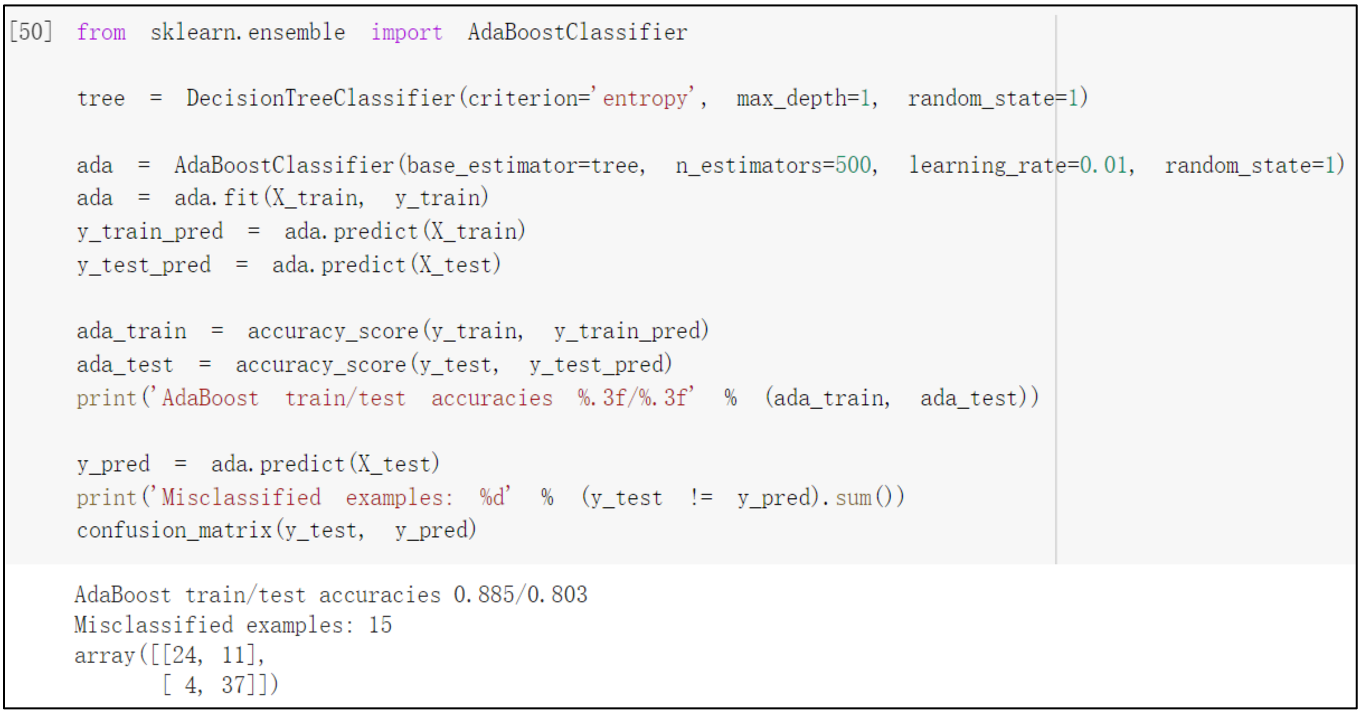
正確率



```
[104] from sklearn.metrics import accuracy_score
      from sklearn.metrics import confusion_matrix

      bag = bag.fit(X_train, y_train)
      bag_train = bag.score(X_train,y_train)
      bag_test = bag.score(X_test, y_test)
      print('Bagging train/test accuracies %.3f/%.3f' % (bag_train, bag_test))

      y_pred = bag.predict(X_test)
      print('Misclassified examples: %d' % (y_test != y_pred).sum())
      confusion_matrix(y_test, y_pred)

      Bagging train/test accuracies 0.850/0.829
      Misclassified examples: 13
      array([[25, 10],
             [ 3, 38]])
```

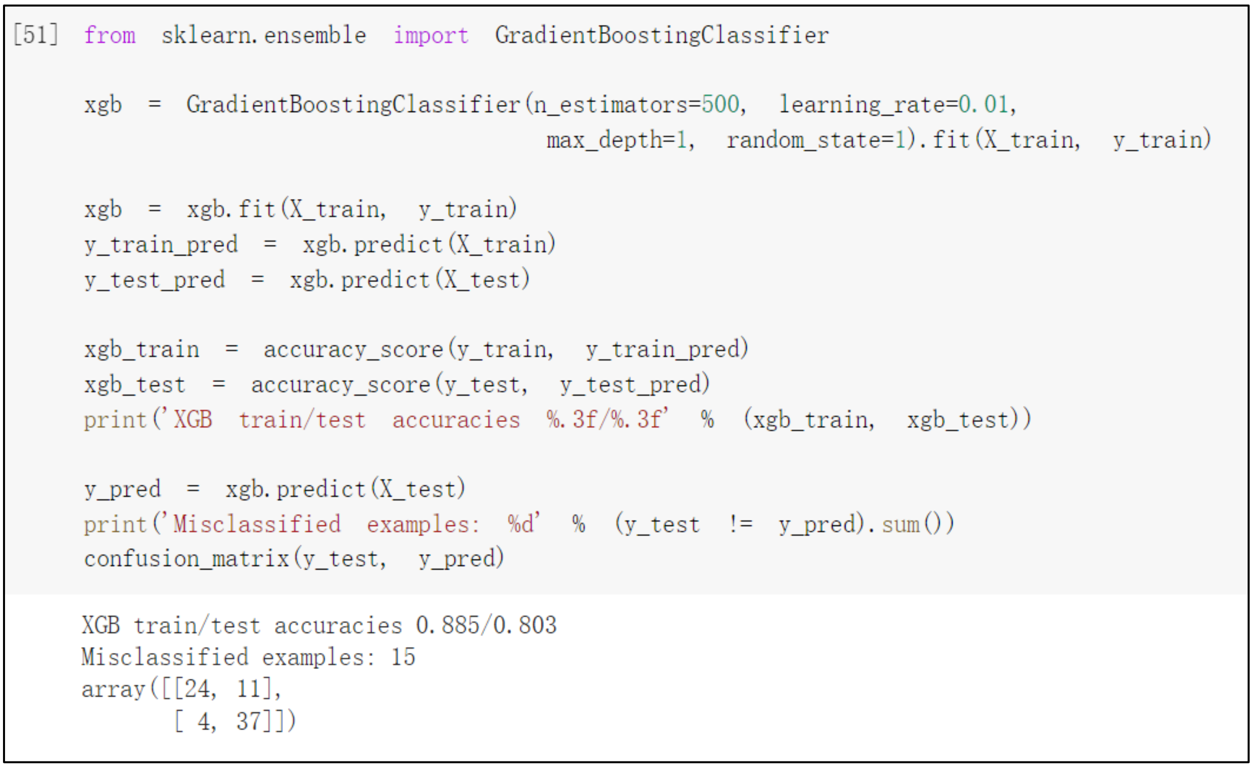
九.Boosting

AdaBoost
n_estimators=500, learning_rate=0.01



```
[50] from sklearn.ensemble import AdaBoostClassifier

     tree = DecisionTreeClassifier(criterion='entropy', max_depth=1, random_state=1)

     ada = AdaBoostClassifier(base_estimator=tree, n_estimators=500, learning_rate=0.01, random_state=1)
     ada = ada.fit(X_train, y_train)
     y_train_pred = ada.predict(X_train)
     y_test_pred = ada.predict(X_test)

     ada_train = accuracy_score(y_train, y_train_pred)
     ada_test = accuracy_score(y_test, y_test_pred)
     print('AdaBoost train/test accuracies %.3f/%.3f' % (ada_train, ada_test))

     y_pred = ada.predict(X_test)
     print('Misclassified examples: %d' % (y_test != y_pred).sum())
     confusion_matrix(y_test, y_pred)

     AdaBoost train/test accuracies 0.885/0.803
     Misclassified examples: 15
     array([[24, 11],
            [ 4, 37]])
```

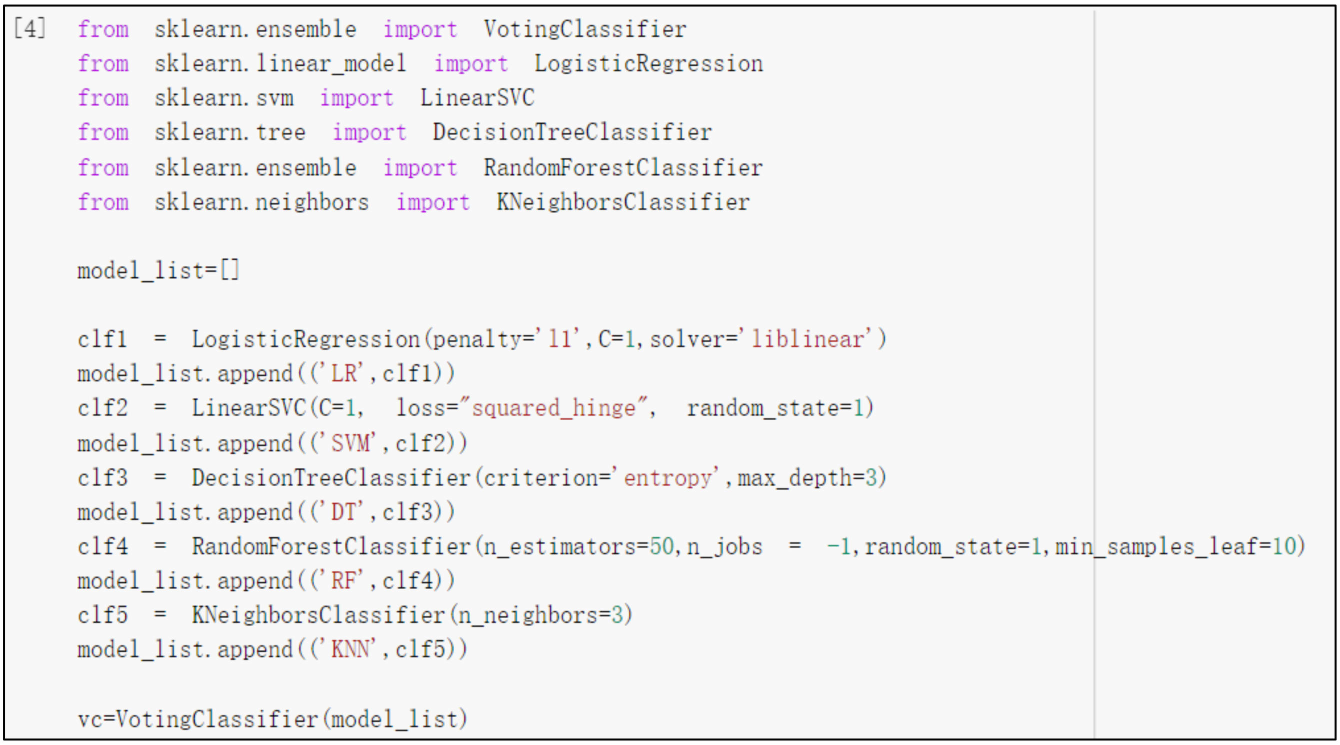
GradientBoost
n_estimators=500, learning_rate=0.01



```
[51] from sklearn.ensemble import GradientBoostingClassifier

     xgb = GradientBoostingClassifier(n_estimators=500, learning_rate=0.01,
                                      max_depth=1, random_state=1).fit(X_train, y_train)

     xgb = xgb.fit(X_train, y_train)
     y_train_pred = xgb.predict(X_train)
     y_test_pred = xgb.predict(X_test)

     xgb_train = accuracy_score(y_train, y_train_pred)
     xgb_test = accuracy_score(y_test, y_test_pred)
     print('XGB train/test accuracies %.3f/%.3f' % (xgb_train, xgb_test))

     y_pred = xgb.predict(X_test)
     print('Misclassified examples: %d' % (y_test != y_pred).sum())
     confusion_matrix(y_test, y_pred)

     XGB train/test accuracies 0.885/0.803
     Misclassified examples: 15
     array([[24, 11],
            [ 4, 37]])
```

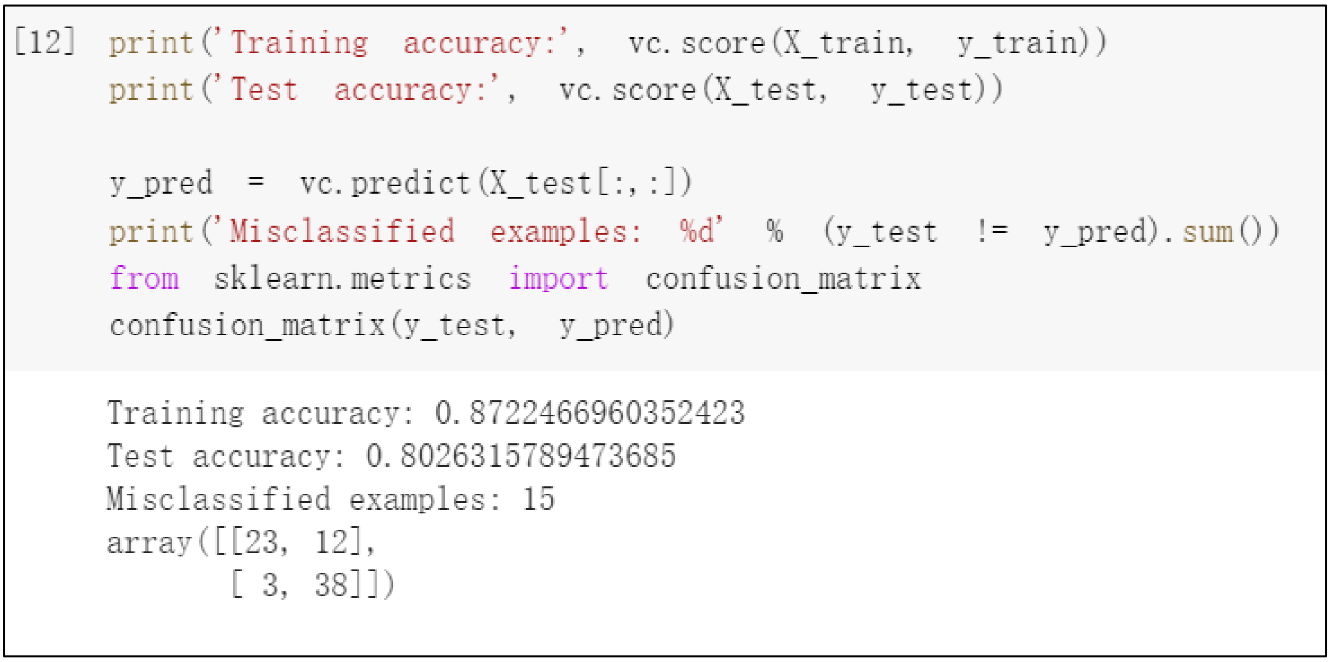
十.Voting

使用Logistic Regression,Linear SVM,Decision Tree,Random Forest,KNN



```
[4] from sklearn.ensemble import VotingClassifier
    from sklearn.linear_model import LogisticRegression
    from sklearn.svm import LinearSVC
    from sklearn.tree import DecisionTreeClassifier
    from sklearn.ensemble import RandomForestClassifier
    from sklearn.neighbors import KNeighborsClassifier

    model_list=[]

    clf1 = LogisticRegression(penalty='l1',C=1,solver='liblinear')
    model_list.append(('LR',clf1))
    clf2 = LinearSVC(C=1, loss="squared_hinge", random_state=1)
    model_list.append(('SVM',clf2))
    clf3 = DecisionTreeClassifier(criterion='entropy',max_depth=3)
    model_list.append(('DT',clf3))
    clf4 = RandomForestClassifier(n_estimators=50,n_jobs = -1,random_state=1,min_samples_leaf=10)
    model_list.append(('RF',clf4))
    clf5 = KNeighborsClassifier(n_neighbors=3)
    model_list.append(('KNN',clf5))

    vc=VotingClassifier(model_list)
```

正確率與confusion matrix



```
[12] print('Training accuracy:', vc.score(X_train, y_train))
     print('Test accuracy:', vc.score(X_test, y_test))

     y_pred = vc.predict(X_test[:,:])
     print('Misclassified examples: %d' % (y_test != y_pred).sum())
     from sklearn.metrics import confusion_matrix
     confusion_matrix(y_test, y_pred)
```

```
Training accuracy: 0.8722466960352423
Test accuracy: 0.8026315789473685
Misclassified examples: 15
array([[23, 12],
       [ 3, 38]])
```

總結與心得:

Logistic Regression-0.75
SVM-0.763
Decision Tree-0.71
Random Forest-0.80
knn-0.76
PCA LR-0.80
LDA-0.76
Bagging-0.829
Boosting-0.803
Voting-0.802

從數據上來看，正確率分布在0.75~0.83之間，可能特徵對於結果有些地方是預測不到的，或是特徵不會絕對影響到結果，有些不明因素，才會使正確率不能達到高峰，如果絕對透過特徵去學習，可能導致失準，所以上面在關於隨機選取特徵與樣本的學習法才會比較出色，會分散特徵的影響，像是Random Forest和Ensemble learing。

透過這學期的修課大致了解了各個sklearn不同的學習法，大致了解如何運作與數學統計概念。但其中搞不太懂SBS特徵選取，有點不太懂，好像選取幾個特徵出來不會比全部特徵下去學習還好，而如果是用sklearn模組的feature_selection的SelectKBest好像也是，因為時間的問題就沒有再多去研究，日後如果領域再接觸到會再詳細探查其原理，透過這短短的一學期時間，稍微了解了機器學習的架構與操作，雖然無法對於數學原理深入探討，但對於未來相信有莫大的幫助。

程式碼

https://drive.google.com/drive/folders/1RqZunw1EU6_pjXwilxs_7HTJav3U23oI?usp=sharing