Lab 8 Dimensionality Reduction (17/6/2022)

ລະຫັດນັກສຶກສາ: 205Q0010.19

ຊື່ ແລະ ນາມສະກຸນ: ທ້າວ ນຸຊົ່ວ ເຮີ 3CW1

ຈົ່ງນຳໃຊ້ຄຳສັ່ງຂອງ Python ດ້ວຍ principal_component_analysis.ipynb ເພື່ອຕອບຄຳຖາມຕໍ່ໄປນີ້ໃຫ້ ສຳເລັດ:

1.ຈາກຊຸດຂໍ້ມູນ Wine..csv. ໃຫ້ X ເປັນຄຸນລັກສະນະ (Features) ຍົກ Y (Label:Customer_Segment).

```
+ Code  + Markdown

dataset = pd.read_csv('Wine.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
print(dataset.head())

[24]  ✓ 0.1s                                                    Python
```

```
   Alcohol  Malic_Acid   Ash  Ash_Alcanity  Magnesium  Total_Phenols  \
0    14.23        1.71  2.43          15.6        127           2.80
1    13.20        1.78  2.14          11.2        100           2.65
2    13.16        2.36  2.67          18.6        101           2.80
3    14.37        1.95  2.50          16.8        113           3.85
4    13.24        2.59  2.87          21.0        118           2.80

   Flavanoids  Nonflavanoid_Phenols  Proanthocyanins  Color_Intensity   Hue  \
0        3.06                  0.28             2.29             5.64  1.04
1        2.76                  0.26             1.28             4.38  1.05
2        3.24                  0.30             2.81             5.68  1.03
3        3.49                  0.24             2.18             7.80  0.86
4        2.69                  0.39             1.82             4.32  1.04

   OD280  Proline  Customer_Segment
0   3.92     1065                 1
1   3.40     1050                 1
2   3.17     1185                 1
3   3.45     1480                 1
4   2.93      735                 1
```

2. ຈົ່ງທຳການແບ່ງຊຸດຂໍ້ມູນອອກເປັນຊຸດຝຶກ 70 ແລະ ຊຸດທົດສອບ 30 .

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 0)

✓ 0.9s                                                          Python
```

3. ຈົ່ງທຳການປະມວນຜົນຊຸດຂໍ້ມູນໃນຂໍ້ທີ 2 ດ້ວຍແບບຈຳລອງ KneighborsClassifier, GaussianNB ແລະ DecisionTreeClassifier ພ້ອມທັງລາຍງານຜົນດ້ວຍ confusion_matrix.

```python
algo = [
    [KNeighborsClassifier(n_neighbors=10), 'KNeighborsClassifier'],
    [GaussianNB(), 'GaussianNB'],
    [DecisionTreeClassifier(), 'DecisionTreeClassifier'],
]
model_score=[]
for a in algo:
    model=a[0]
    model.fit(X_train, y_train) # step 2: fit
    y_pred=model.predict(X_test) # step 3: predict
    score=model.score(X_test, y_test)
    model_score.append([score, a[1]])
    print(f'{a[1]} score = {score}') # step 4: score
    print(metrics.confusion_matrix(y_test, y_pred))
    print(metrics.classification_report(y_test, y_pred))
    print('-' * 100)
print(model_score)
```

[41] ✓ 0.1s Python

```
KNeighborsClassifier score = 0.9814814814814815
[[19  0  0]
 [ 1 21  0]
 [ 0  0 13]]
              precision    recall  f1-score   support

           1       0.95      1.00      0.97        19
           2       1.00      0.95      0.98        22
           3       1.00      1.00      1.00        13

    accuracy                           0.98        54
   macro avg       0.98      0.98      0.98        54
weighted avg       0.98      0.98      0.98        54

----------------------------------------------------------------------------------------------------
GaussianNB score = 0.9814814814814815
[[19  0  0]
 [ 1 21  0]
 [ 0  0 13]]
              precision    recall  f1-score   support

           1       0.95      1.00      0.97        19
           2       1.00      0.95      0.98        22
           3       1.00      1.00      1.00        13
...
   macro avg       1.00      1.00      1.00        54
weighted avg       1.00      1.00      1.00        54

----------------------------------------------------------------------------------------------------
[[0.9814814814814815, 'KNeighborsClassifier'], [0.9814814814814815, 'GaussianNB'], [1.0,
'DecisionTreeClassifier']]
```

4. ຈົ່ງທຳການປັບຄ່າຂອງຊຸດຂໍ້ມູນໃນຂໍ້ທີ 2 ດ້ວຍ StandardScaler.

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(X)
print(X)
```

[25] ✓ 0.7s Python

```
[[ 1.51861254 -0.5622498   0.23205254 ...  0.36217728  1.84791957
   1.01300893]
 [ 0.24628963 -0.49941338 -0.82799632 ...  0.40605066  1.1134493
   0.96524152]
 [ 0.19687903  0.02123125  1.10933436 ...  0.31830389  0.78858745
   1.39514818]
 ...
 [ 0.33275817  1.74474449 -0.38935541 ... -1.61212515 -1.48544548
   0.28057537]
 [ 0.20923168  0.22769377  0.01273209 ... -1.56825176 -1.40069891
   0.29649784]
 [ 1.39508604  1.58316512  1.36520822 ... -1.52437837 -1.42894777
  -0.59516041]]
```

5. ຈົ່ງທຳການປະມວນຜົນຊຸດຂໍ້ມູນໃນຂໍ້ທີ 4 ດ້ວຍແບບຈຳລອງ KneighborsClassifier, GaussianNB ແລະ DecisionTreeClassifier ພ້ອມທັງລາຍງານຜົນດ້ວຍ confusion_matrix.

```
algo = [
    [KNeighborsClassifier(n_neighbors=10), 'KNeighborsClassifier'],
    [GaussianNB(), 'GaussianNB'],
    [DecisionTreeClassifier(), 'DecisionTreeClassifier'],
]
model_score=[]
for a in algo:
    model=a[0]
    model.fit(X_train, y_train) # step 2: fit
    y_pred=model.predict(X_test) # step 3: predict
    score=model.score(X_test, y_test)
    model_score.append([score, a[1]])
    print(f'{a[1]} score = {score}') # step 4: score
    print(metrics.confusion_matrix(y_test, y_pred))
    print(metrics.classification_report(y_test, y_pred))
    print('-' * 100)
print(model_score)
```

Python

```
KNeighborsClassifier score = 0.9722222222222222
[[14  0  0]
 [ 1 15  0]
 [ 0  0  6]]
              precision    recall  f1-score   support

           1       0.93      1.00      0.97        14
           2       1.00      0.94      0.97        16
           3       1.00      1.00      1.00         6

    accuracy                           0.97        36
   macro avg       0.98      0.98      0.98        36
weighted avg       0.97      0.97      0.97        36

----------------------------------------------------------------------------------------------------
GaussianNB score = 0.9722222222222222
[[14  0  0]
 [ 1 15  0]
 [ 0  0  6]]
              precision    recall  f1-score   support

           1       0.93      1.00      0.97        14
           2       1.00      0.94      0.97        16
           3       1.00      1.00      1.00         6
…
   macro avg       0.98      0.98      0.98        36
weighted avg       0.97      0.97      0.97        36

----------------------------------------------------------------------------------------------------
[[0.9722222222222222, 'KNeighborsClassifier'], [0.9722222222222222, 'GaussianNB'], [0.9722222222222222,
'DecisionTreeClassifier']]
```

6. ຈົ່ງອະທິບາຍ ແລະ ສົມທຽບຜົນການປະມວນຜົນຂໍ້ທີ 3 ແລະ 5.

ຂໍ້ທີ 3 :ແມ່ນ: KNeighborsClassifier score = 0.9722222222222222 ແລະ confusion_matrix ແມ່ນ :

[[14 0 0]
 [1 15 0]
 [0 0 6]]

GaussianNB score = 0.9722222222222222

[[14 0 0]
 [1 15 0]
 [0 0 6]]

ຂໍ້ທີ 5 ແມ່ນ: KNeighborsClassifier score = 0.9814814814814815ແລະ confusion_matrix ແມ່ນ :

[[19 0 0]
 [1 21 0]
 [0 0 13]]

GaussianNB score = 0.9814814814814815

[[19 0 0]
 [1 21 0]
 [0 0 13]]

3

```
----------------------------------------------------------------------------------------------------
[[0.9814814814814815, 'KNeighborsClassifier'], [0.9814814814814815, 'GaussianNB'], [1.0,
'DecisionTreeClassifier']]
```

5

```
----------------------------------------------------------------------------------------------------
[[0.9722222222222222, 'KNeighborsClassifier'], [0.9722222222222222, 'GaussianNB'], [0.9722222222222222,
'DecisionTreeClassifier']]
```

7. ຈົ່ງທຳການຫຼຸດຜ່ອນຂະໜາດຂໍ້ມູນ (Dimensionality Reduction) ຂອງຊຸດຂໍ້ມູນໃນຂໍ້ທີ 4 ດ້ວຍ PCA ໂດຍ ໃຫ້ n_components = 2.

```
from sklearn.decomposition import PCA
pca = PCA(n_components = 2)
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
explained_variance = pca.explained_variance_ratio_
print(explained_variance)
```

[26] ✓ 0.7s Python

```
[0.3685082 0.1858638]
```

8. ຈົ່ງທຳການປະມວນຜົນຊຸດຂໍ້ມູນໃນຂໍ້ທີ 7 ດ້ວຍແບບຈຳຮອງ KneighborsClassifier, GaussianNB ແລະ DecisionTreeClassifier ພ້ອມທັງລາຍງານຜົນດ້ວຍ confusion_matrix.

```
algo = [
    [KNeighborsClassifier(n_neighbors=10), 'KNeighborsClassifier'],
    [GaussianNB(), 'GaussianNB'],
    [DecisionTreeClassifier(), 'DecisionTreeClassifier'],
]
model_score=[]
for a in algo:
    model=a[0]
    model.fit(X_train, y_train) # step 2: fit
    y_pred=model.predict(X_test) # step 3: predict
    score=model.score(X_test, y_test)
    model_score.append([score, a[1]])
    print(f'{a[1]} score = {score}') # step 4: score
    print(metrics.confusion_matrix(y_test, y_pred))
    print(metrics.classification_report(y_test, y_pred))
    print('-' * 100)
print(model_score)
```

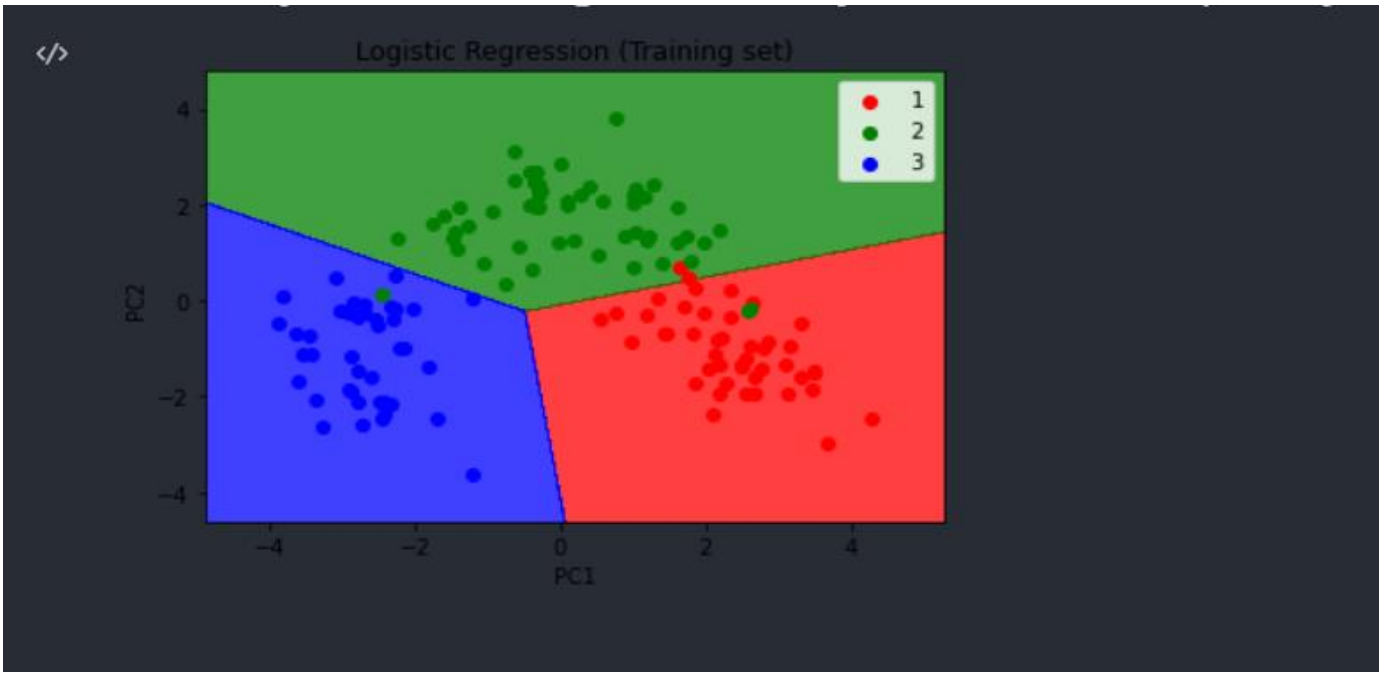
Python

```
KNeighborsClassifier score = 0.7037037037037037
[[18  1  0]
 [ 1 15  6]
 [ 2  6  5]]
              precision    recall  f1-score   support

           1       0.86      0.95      0.90        19
           2       0.68      0.68      0.68        22
           3       0.45      0.38      0.42        13

    accuracy                           0.70        54
   macro avg       0.66      0.67      0.67        54
weighted avg       0.69      0.70      0.69        54

----------------------------------------------------------------------------------------------------
GaussianNB score = 0.7962962962962963
[[17  0  2]
 [ 1 17  4]
 [ 1  3  9]]
              precision    recall  f1-score   support

           1       0.89      0.89      0.89        19
           2       0.85      0.77      0.81        22
           3       0.60      0.69      0.64        13
…
   macro avg       0.62      0.63      0.62        54
weighted avg       0.65      0.67      0.66        54

----------------------------------------------------------------------------------------------------
[[0.7037037037037037, 'KNeighborsClassifier'], [0.7962962962962963, 'GaussianNB'], [0.6666666666666666,
'DecisionTreeClassifier']]
```

9. ຈົ່ງທຳການສ້າງ from matplotlib.colors import ListedColormap

```
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green', 'blue')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green', 'blue'))(i), label = j)
plt.title('Logistic Regression (Training set)')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.legend()
plt.show()
```

[9] ✓ 1.1s Python



```python
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green', 'blue')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green', 'blue'))(i), label = j)
plt.title('Logistic Regression (Test set)')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.legend()
plt.show()
```

✓ 0.6s Python