ACTIVITY #3

PCA : Principal Component Analysis

Topics

3.1 Data Exploration

3.2 PCA

3.3 KMEAN Clustering

Libraries

1. import numpy as np
2. import pandas as pd
3. import matplotlib.pyplot as plt
4. import seaborn as sns
5. from sklearn import preprocessing
6. from sklearn.decomposition import PCA
7. from sklearn.cluster import Kmeans
8. from sklearn.metrics import accuracy_score
9. from scipy.stats import mode

3.1

DATA EXPLORATION AND TRANSFORM

3.1 Data exploration and Transform

1
- Read .csv file
  - df_wine = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data', header=None)

2
- df_wine.columns = ['Class label', 'Alcohol', 'Malic acid', 'Ash',
- 'Alcalinity of ash', 'Magnesium', 'Total phenols',
- 'Flavanoids', 'Nonflavanoid phenols', 'Proanthocyanins',
- 'Color intensity', 'Hue',
- 'OD280/OD315 of diluted wines', 'Proline']

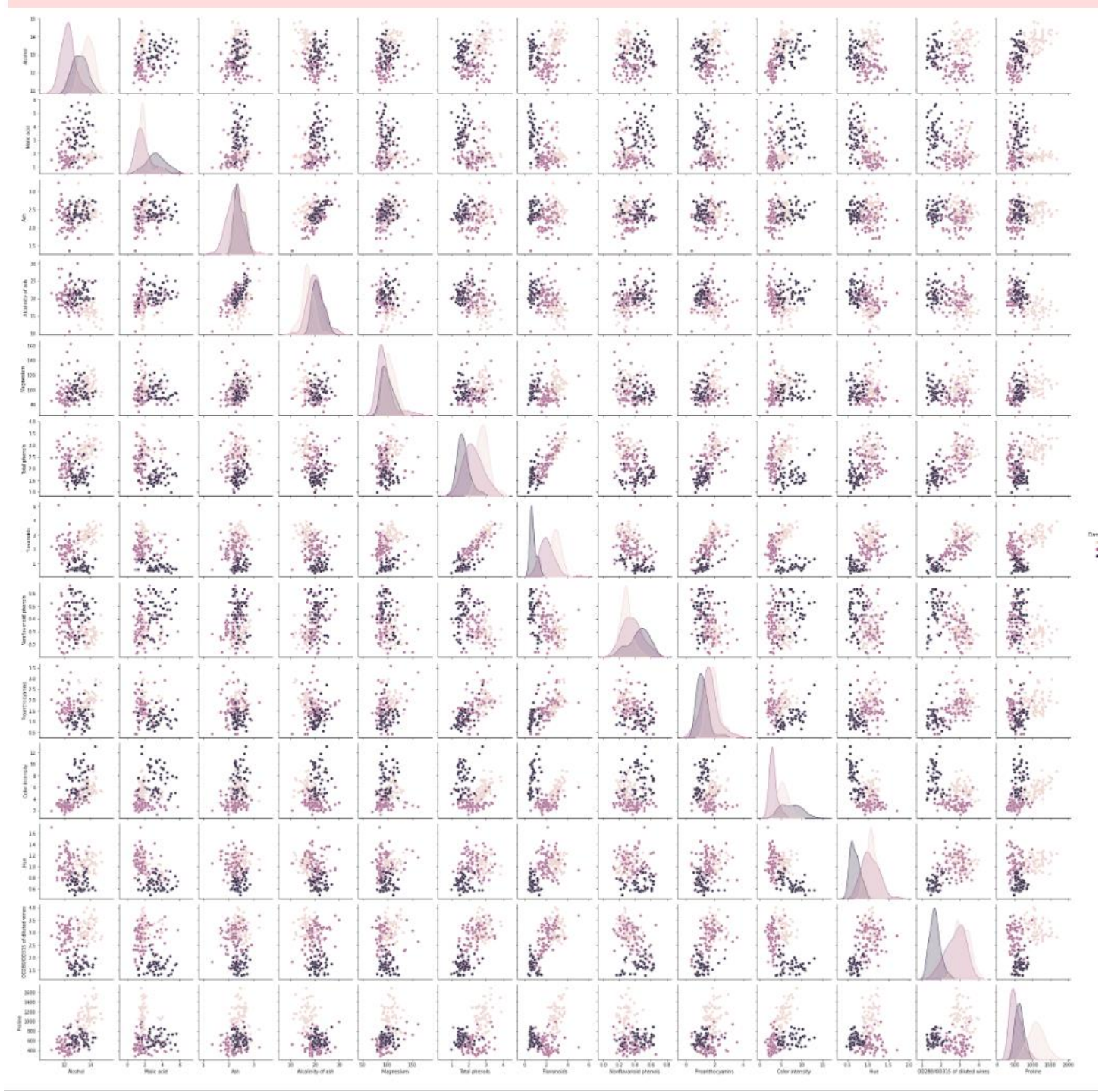
3
- #Data Explore and Cleaning
  - Fillna()
  - describe()

4
- # Split variables / output
  - X = all except [‘Class label’]
  - Y = df_wine[‘Class label’]

5
- #Data Transformation
  - Standardized data (z-transform) of X

6
- #Visualize Variable (scatter plot pairs of variables)
  - sns.pairplot(df_wine, hue='Class label', size=2.5);

#Visualize Variable (scatter plot pairs of variables)



3.2

PCA

PRINCIPAL COMPONENT ANALYSIS

3.2 PCA of all variables

1

- # PCA all variables (after standardized data)
  - pca = PCA()
  - X_pca = pca.fit_transform(X_standard)

2

- print('Explained Variance ratio = ', pca.explained_variance_ratio_)
- Visualize Explained Variance (eigenvalues)
  - plt.bar() ค่าของ pca.explained_variance_ratio_

3

- print('Explained Variance (eigenvalues) = ', pca.explained_variance_)
- print('----------')
- print('PCA components (eigenvectors) ')
- print(pca.components_[0:2,:])

3.2 PCA of 2 components

1

- # PCA all variables (after standardized data)
  - pca2 = PCA(n_components=2)
  - X_pca_2 = pca2.fit_transform(X)

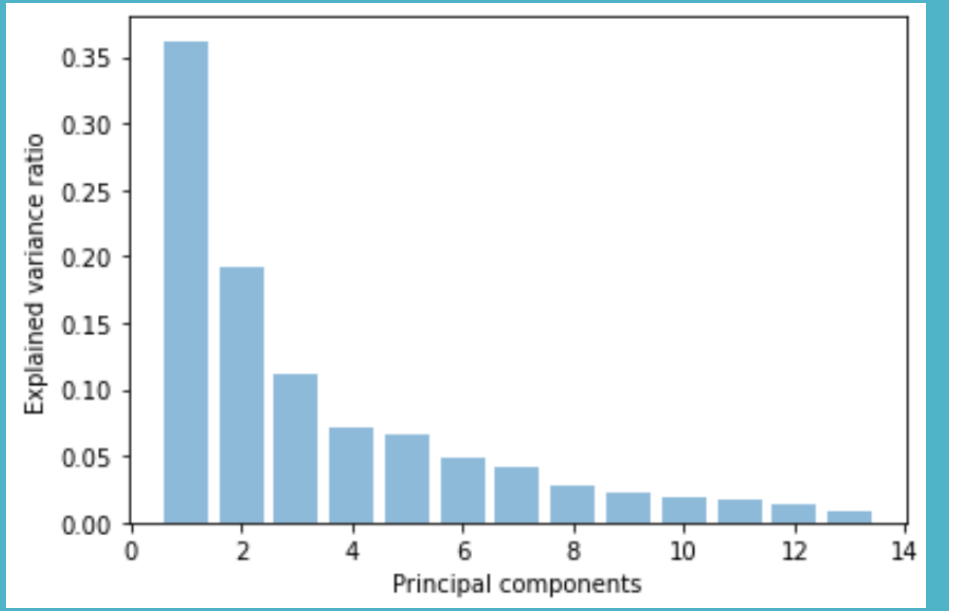
2

- print('Explained Variance ratio = ', pca2.explained_variance_ratio_)
- Visualize Explained Variance (eigenvalues)
  - plt.bar() ค่าของ pca.explained_variance_ratio_

3

- print('Explained Variance (eigenvalues) = ', pca2.explained_variance_)
- print('----------')
- print('PCA components (eigenvectors) ')
- print(pca2.components_[0:2,:])

3.2 PCA Results



pca = PCA()

```
Explained Variance ratio =  [0.36198848 0.1920749  0.11123631 0.0706903  0.06563294 0.04935823
 0.04238679 0.02680749 0.02222153 0.01930019 0.01736836 0.01298233
 0.00795215]


Explained Variance (eigenvalues) =  [4.73243698 2.51108093 1.45424187 0.92416587 0.85804868 0.64528221
 0.55414147 0.35046627 0.29051203 0.25232001 0.22706428 0.16972374
 0.10396199]
-----------------------------------------
PCA components (eigenvectors) along row
[ 0.1443294  -0.24518758 -0.00205106 -0.23932041  0.14199204  0.39466085
  0.4229343  -0.2985331   0.31342949 -0.0886167   0.29671456  0.37616741
  0.28675223]
[-0.48365155 -0.22493093 -0.31606881  0.0105905  -0.299634   -0.06503951
  0.00335981 -0.02877949 -0.03930172 -0.52999567  0.27923515  0.16449619
 -0.36490283]
[-0.20738262  0.08901289  0.6262239   0.61208035  0.13075693  0.14617896
  0.1506819   0.17036816  0.14945431 -0.13730621  0.08522192  0.16600459
 -0.12674592]
```

pca2 = PCA(n_components=2)

```
Explained Variance ratio =  [0.36198848 0.1920749 ]
Explained Variance (eigenvalues) =  [4.73243698 2.51108093]
(2, 13)
[ 0.1443294  -0.24518758 -0.00205106 -0.23932041  0.14199204  0.39466085
  0.4229343  -0.2985331   0.31342949 -0.0886167   0.29671456  0.37616741
  0.28675223]
[-0.48365155 -0.22493093 -0.31606881  0.0105905  -0.299634   -0.06503951
  0.00335981 -0.02877949 -0.03930172 -0.52999567  0.27923515  0.16449619
 -0.36490283]
(178,)
```

3.3

KMEAN CLUSTERING

PCA VS ALL_VARIABLES

3.3 Kmean Clustering of All variables and Accuracy

1

- # Kmean all variables
- # Compute the n_clusters np.unique(Y)
  - kmeans = KMeans(n_clusters, random_state=0)
  - clusters = kmeans.fit_predict(X)

2

- # Scatter Plot cluster center
  - col_x , col_y เป็น column คู่ ใดๆ ใน X input variables
  - plt.scatter(X[col_x], X[col_y], c=clusters, edgecolors='m',alpha=0.75,s=150)
  - centers = kmeans.cluster_centers_
  - plt.scatter(centers[:, 0], centers[:, 1], c='red', s=200, alpha=0.5);

3

- # map clusters to real label
  - labels = np.zeros_like(clusters)
  - for i in range(10):
    - mask = (clusters == i)
    - labels[mask] = mode(Y[mask])[0]

4

- # Calculate Accuracy
  - accuracy_score(labels, Y)

3.3 Kmean Clustering of PCA and Accuracy

1

- # Kmean all variables
- # Compute the n_clusters np.unique(Y)
  - kmeans_PCA = KMeans(n_clusters, random_state=0)
  - clusters_PCA = kmeans_PCA.fit_predict(X_PCA_2)

2

- # Scatter Plot cluster center
  - plt.scatter(X_PCA_2[:, 0], X_PCA_2[:, 1], c=clusters, edgecolors='m',alpha=0.75,s=150)
  - centers_pca = kmeans.cluster_centers_
  - plt.scatter(centers_pca[:, 0], centers_pca[:, 1], c='red', s=200, alpha=0.5);

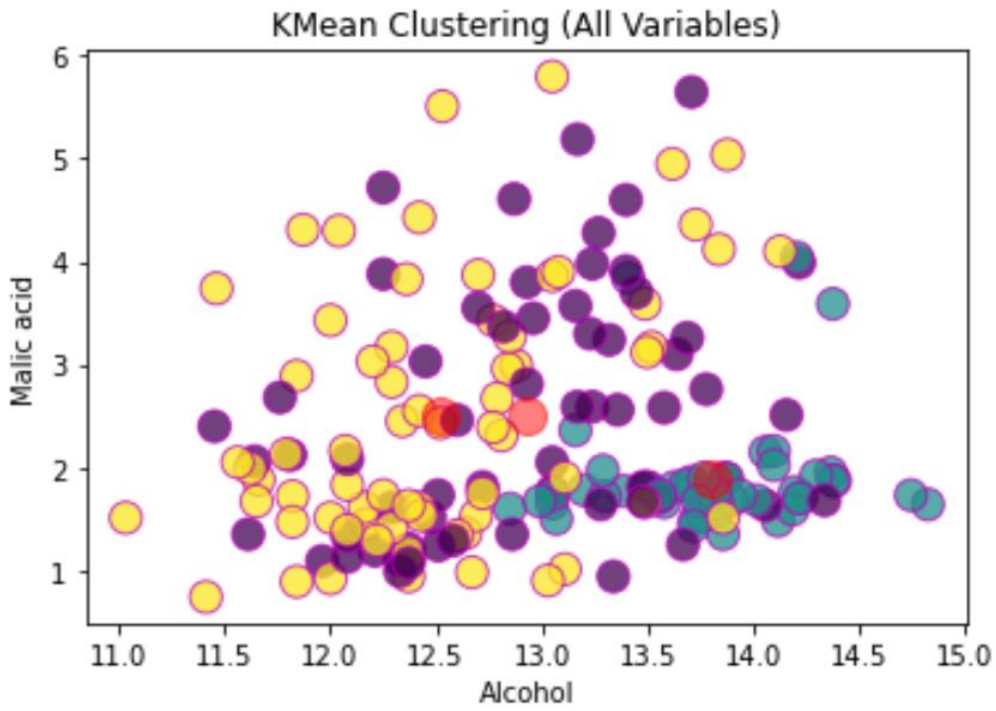
3

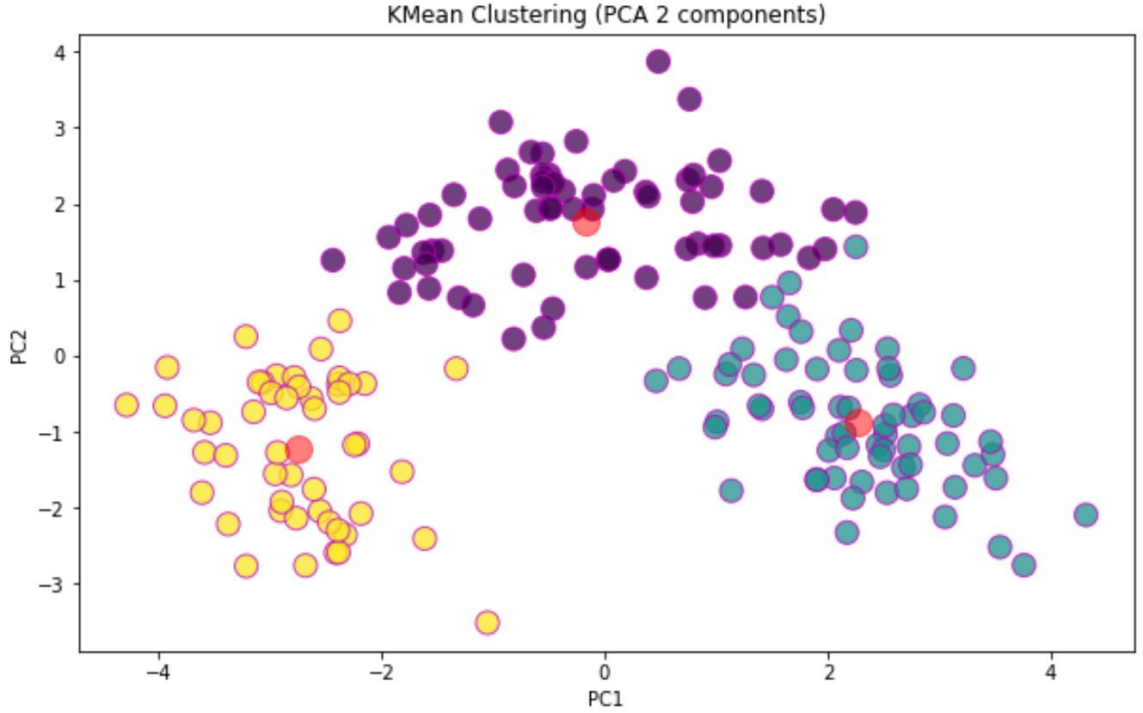
- # map clusters to real label
  - Labels_pca = np.zeros_like(clusters_pca)
  - for i in range(10):
    - mask = (clusters_pca == i)
    - Labels_pca[mask] = mode(Y[mask])[0]

4

- # Calculate Accuracy
  - accuracy_score(labels_pca, Y)

Kmean Clustering Results






```
# Kmean Accuracy all variables
accuracy_score(labels, Y)
```

```
0.702247191011236
```

```
# Kmean Accuracy
# Compute the accuracy
accuracy_score(labels_PCA, Y)
```

```
0.9662921348314607
```