Computer Engineering Department National University of Technology Islamabad, Pakistan

Introduction to Data Mining

Practice Exercise 06

Name: Muhammad Sami Uddin Rafay

Roll Number: F18604013

Submitted To: Dr. Kamran Javed

Date: 17 December 2020

Practice Exercise 06

Principle Component Analysis

Objective:

- To implement Principal Component Analysis
- The principal components of a collection of points in a real p-space are a sequence of direction vectors, where the vector is the direction of a line that best fits the data while being orthogonal to the first vectors.

Equipment/Software Required:

- Python (Spyder 4.0 Anaconda Distribution)

Background:

Tasks:

Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn import datasets
from sklearn.decomposition import PCA
from pyod.models.copod import COPOD

#from pca import pca

# Load the iris data from sklearn
iris = datasets.load_iris()
iris=pd.DataFrame(iris.data)
#print(iris)

# Get input variables (from column 1 to 4)

X=iris[iris.columns[0:4]]
#Y = iris[iris[:,:5]]
print(X)
#print(Y)

# De-mean data by subtrating mean form each point
X1=X-np.mean(X)
#print(X1)
#[COEFF, SCORE, LATENT,TSQUARED, EXPLAINED]=PCA.fit_transform(X1)


pca = PCA()
pca.fit(X1)
```

```
coeff = np.transpose(pca.components_)
print(coeff)

latent=(pca.explained_variance_)
print(latent)
#print(pca.score(X))
explained=pca.explained_variance_ratio_
print(explained*100)
#d=pca.singular_values_
#print(d)
f=pca.n_samples_
print(f)

numberOfDimentions=4;
reducedDimention=coeff[0,0:4]
print(reducedDimention)
reducedFeatureMatrix=X-reducedDimention
print(reducedFeatureMatrix)

plt.figure(3,figsize=(5,10))
X = np.arange(4)
col=['b','g','r','y']
for i in range(1,5,1):
#plt.hist(coeff[3])
    plt.subplot(2,2,i)
    plt.bar(X,coeff[i-1], color = col[i-1] )
    plt.title("Eigen Vector "+str(i))
    plt.grid()
plt.show()


iris = datasets.load_iris()
X = iris.data[:, :4]  # we only take the first two features.
y = iris.target

x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5

plt.figure(2, figsize=(8, 6))
plt.clf()

# Plot the training points
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set1,
            edgecolor='k')
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')

plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())

# To getter a better understanding of interaction of the dimensions
# plot the first three PCA dimensions
fig = plt.figure(1, figsize=(8, 6))
```

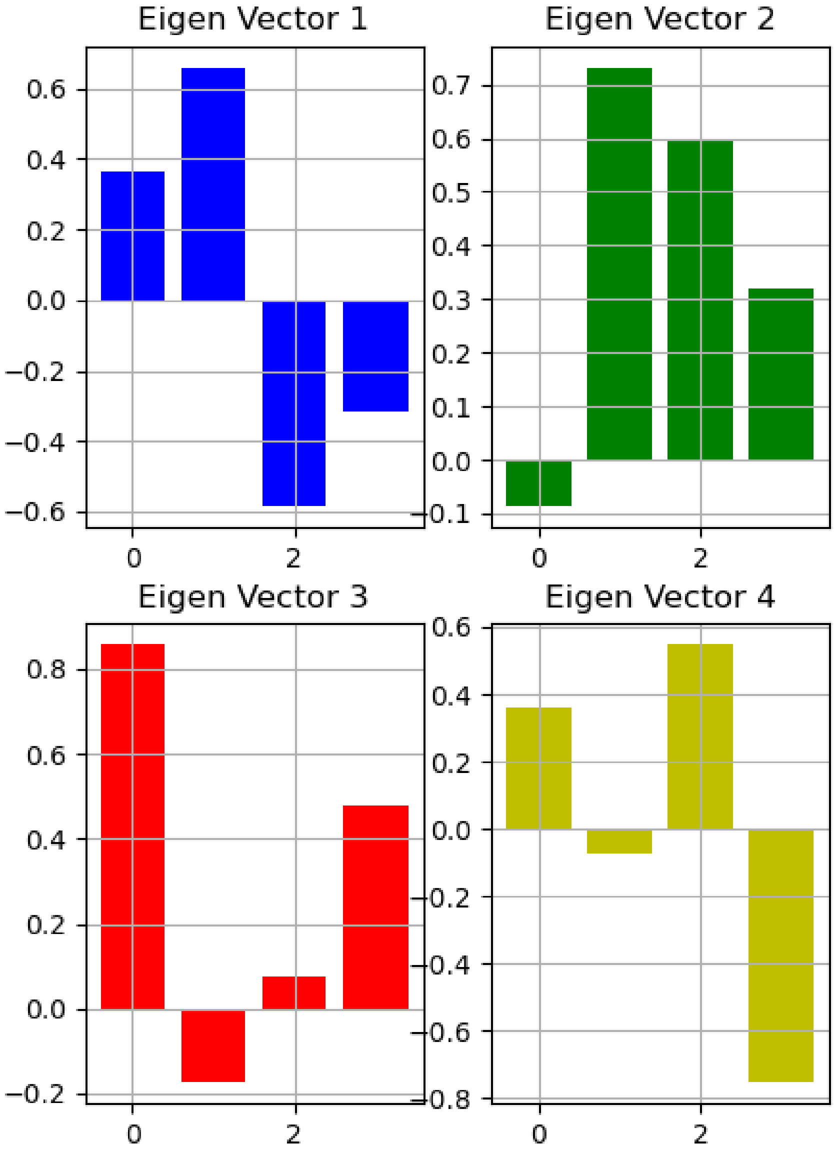
```
ax = Axes3D(fig, elev=-150, azim=110)
X_reduced = PCA(n_components=3).fit_transform(iris.data)
ax.scatter(X_reduced[:, 0], X_reduced[:, 1], X_reduced[:, 2], c=y,
          cmap=plt.cm.Set1, edgecolor='k', s=40)
ax.set_title("First three PCA directions")
ax.set_xlabel("1st eigenvector")
ax.w_xaxis.set_ticklabels([])
ax.set_ylabel("2nd eigenvector")
ax.w_yaxis.set_ticklabels([])
ax.set_zlabel("3rd eigenvector")
ax.w_zaxis.set_ticklabels([])

plt.show()
```

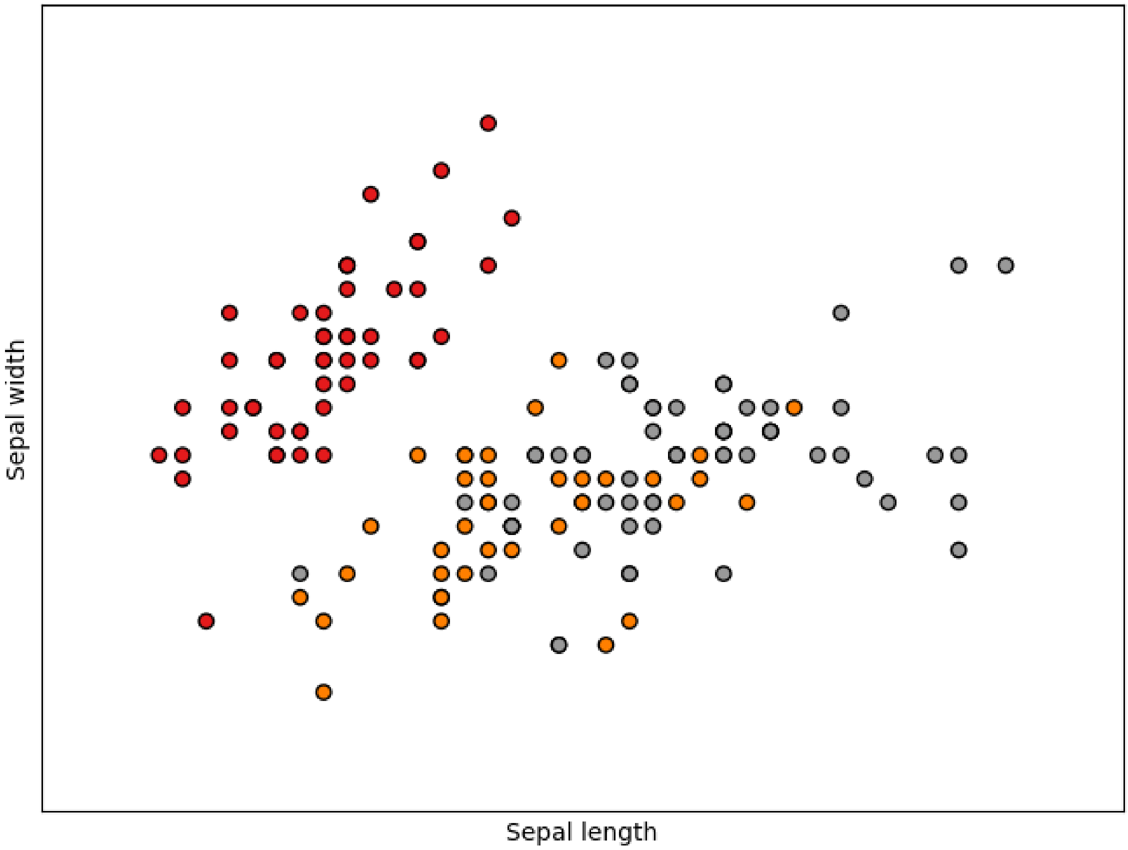
Output:

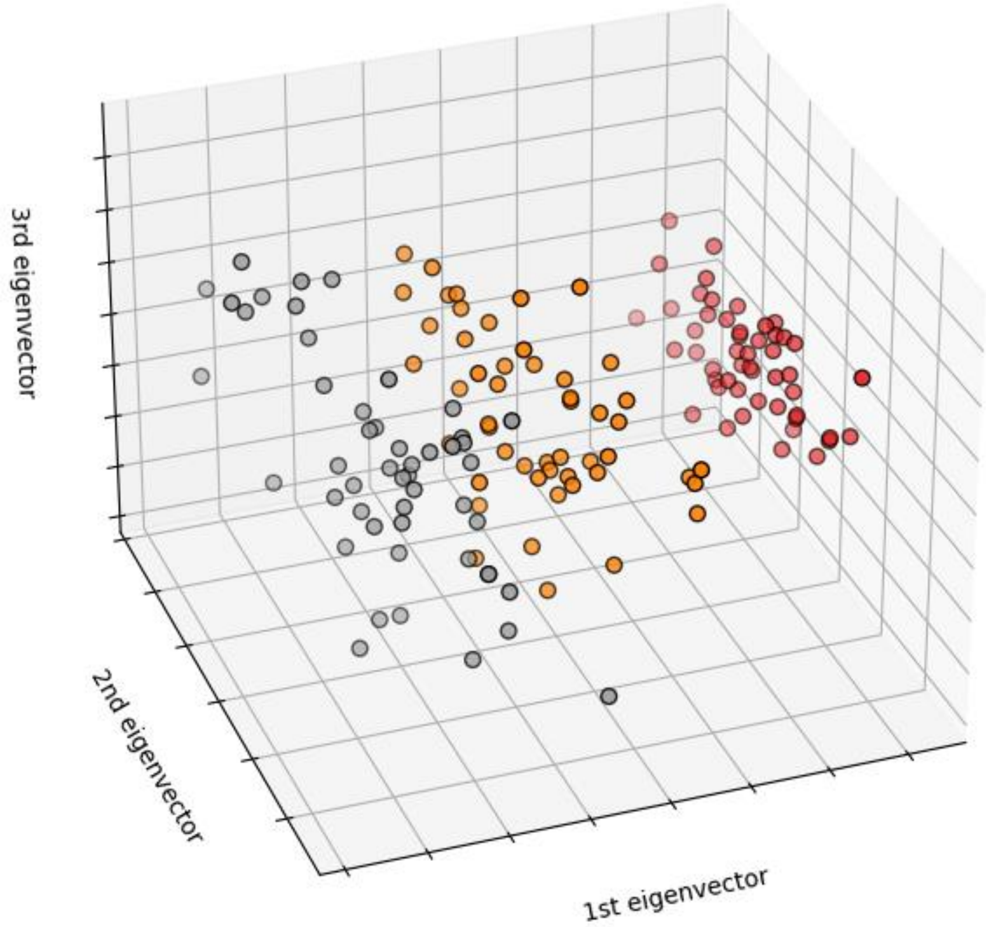
```
       0    1    2    3
0    5.1  3.5  1.4  0.2
1    4.9  3.0  1.4  0.2
2    4.7  3.2  1.3  0.2
3    4.6  3.1  1.5  0.2
4    5.0  3.6  1.4  0.2
..   ...  ...  ...  ...
145  6.7  3.0  5.2  2.3
146  6.3  2.5  5.0  1.9
147  6.5  3.0  5.2  2.0
148  6.2  3.4  5.4  2.3
149  5.9  3.0  5.1  1.8

[150 rows x 4 columns]
[[ 0.36138659  0.65658877 -0.58202985 -0.31548719]
 [-0.08452251  0.73016143  0.59791083  0.3197231 ]
 [ 0.85667061 -0.17337266  0.07623608  0.47983899]
 [ 0.3582892  -0.07548102  0.54583143 -0.75365743]]
[4.22824171 0.24267075 0.0782095  0.02383509]
[92.46187232  5.30664831  1.71026098  0.52121839]
150
[ 0.36138659  0.65658877 -0.58202985 -0.31548719]
            0         1        2         3
0    4.738613  2.843411  1.98203  0.515487
1    4.538613  2.343411  1.98203  0.515487
2    4.338613  2.543411  1.88203  0.515487
3    4.238613  2.443411  2.08203  0.515487
4    4.638613  2.943411  1.98203  0.515487
..        ...       ...      ...       ...
145  6.338613  2.343411  5.78203  2.615487
146  5.938613  1.843411  5.58203  2.215487
147  6.138613  2.343411  5.78203  2.315487
148  5.838613  2.743411  5.98203  2.615487
149  5.538613  2.343411  5.68203  2.115487

[150 rows x 4 columns]

runfile('D:/PCA in Python.py', wdir='D:')
       0    1    2    3
0    5.1  3.5  1.4  0.2
```

```
1    4.9  3.0  1.4  0.2
2    4.7  3.2  1.3  0.2
3    4.6  3.1  1.5  0.2
4    5.0  3.6  1.4  0.2
..   ...  ...  ...  ...
145  6.7  3.0  5.2  2.3
146  6.3  2.5  5.0  1.9
147  6.5  3.0  5.2  2.0
148  6.2  3.4  5.4  2.3
149  5.9  3.0  5.1  1.8

[150 rows x 4 columns]
[[ 0.36138659  0.65658877 -0.58202985 -0.31548719]
 [-0.08452251  0.73016143  0.59791083  0.3197231 ]
 [ 0.85667061 -0.17337266  0.07623608  0.47983899]
 [ 0.3582892  -0.07548102  0.54583143 -0.75365743]]
[4.22824171 0.24267075 0.0782095  0.02383509]
[92.46187232  5.30664831  1.71026098  0.52121839]
150
[ 0.36138659  0.65658877 -0.58202985 -0.31548719]
            0         1        2         3
0    4.738613  2.843411  1.98203  0.515487
1    4.538613  2.343411  1.98203  0.515487
2    4.338613  2.543411  1.88203  0.515487
3    4.238613  2.443411  2.08203  0.515487
4    4.638613  2.943411  1.98203  0.515487
..        ...       ...      ...       ...
145  6.338613  2.343411  5.78203  2.615487
146  5.938613  1.843411  5.58203  2.215487
147  6.138613  2.343411  5.78203  2.315487
148  5.838613  2.743411  5.98203  2.615487
149  5.538613  2.343411  5.68203  2.115487

[150 rows x 4 columns]
```

Graphs:


Eigen Vector 1
Eigen Vector 2
Eigen Vector 3
Eigen Vector 4
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0.7
0.5
0.3
0.1
−0.1
0.8
−0.8
0
2


Sepal width
Sepal length



Results and Discussions:

The main purposes of a principal component analysis are the analysis of data to identify patterns and finding patterns to reduce the dimensions of the dataset with minimal loss of information.

The python packages I used in this practical: -

- ✓ NumPy
- ✓ matplotlib
- ✓ pandas
- ✓ Scikit-learn

Conclusion:

We can easily avoid curse of dimensionality with help of PCA algorithm.