# Week4 (PCA) Lab Homework

연세대학교 통계데이터사이언스 석사과정

August 8, 2021

## 1 EX1

#### 1.0.1 : prove PCA

PC를 만드는 vector가 공분산행렬의 eigenvector이고, 그 때의 variance가 eigenvalue라는 것을 증명해보세요 !

$$
\begin{aligned}
\max_{\delta} Var(\delta^T X) &= \delta^T Var(X)\delta \quad \text{s.t} \quad \delta^T\delta = 1 \\
&= \delta^T \Sigma \delta
\end{aligned}
$$

$$
\begin{aligned}
&\Longrightarrow L = \delta^T\Sigma\delta - \lambda(\delta^T\delta - 1) \quad \text{where} \quad \frac{\partial}{\partial\delta}L = 0 \\
&\Longrightarrow \frac{\partial}{\partial\delta}L = 2\Sigma\delta - 2\lambda\delta \\
&\Longrightarrow \Sigma\delta = \lambda\delta
\end{aligned}
$$

$$
\begin{aligned}
\max_{\delta} Var(\delta^T X) &= \delta^T\Sigma\delta \\
&= \delta^T\lambda\delta \\
&= \lambda\delta^T\delta \\
&= \lambda
\end{aligned}
$$

## 2 HW2

#### 2.0.1 : PCA from scratch

- iris 데이터에 numpy만을 사용해서 PCA를 해봅시다!
- We can find PCA makes visualization easier!

[1]:
```
import warnings

import numpy as np
import pandas as pd

from sklearn.preprocessing import StandardScaler

import matplotlib.pyplot as plt
```

```
import seaborn as sns

warnings.simplefilter(action='ignore', category=FutureWarning)
plt.style.use('ggplot')
```

[2]:
```
df = pd.read_csv('https://raw.githubusercontent.com/uiuc-cse/data-fa14/gh-pages/
↪data/iris.csv')
df.head()
```

[2]:

| | sepal_length | sepal_width | petal_length | petal_width | species |
|---|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 4 | 5.0 | 3.6 | 1.4 | 0.2 | setosa |

[3]:
```
X = df.iloc[:,:-1]
label = df.iloc[:,-1]
X.head()
```

[3]:

| | sepal_length | sepal_width | petal_length | petal_width |
|---|---|---|---|---|
| 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 |

### 2.0.2 Using Covariance Matrix

[4]:
```
# Step 1. Center Data
X_scaled = StandardScaler().fit_transform(X)
X_scaled[:5]
```

[4]:
```
array([[-0.90068117,  1.03205722, -1.3412724 , -1.31297673],
       [-1.14301691, -0.1249576 , -1.3412724 , -1.31297673],
       [-1.38535265,  0.33784833, -1.39813811, -1.31297673],
       [-1.50652052,  0.10644536, -1.2844067 , -1.31297673],
       [-1.02184904,  1.26346019, -1.3412724 , -1.31297673]])
```

[5]:
```
# Step 2. Compute Covariance Matrix
cov_matrix = X_scaled.T @ X_scaled / (X_scaled.shape[0]-1) #TODO
cov_matrix
```

[5]:
```
array([[ 1.00671141, -0.11010327,  0.87760486,  0.82344326],
       [-0.11010327,  1.00671141, -0.42333835, -0.358937  ],
       [ 0.87760486, -0.42333835,  1.00671141,  0.96921855],
```

```
        [ 0.82344326, -0.358937  ,  0.96921855,  1.00671141]])
```

[6]:
```python
# Step 3. Eigenvalue decomposition
eigvals, eigvecs = np.linalg.eig(cov_matrix) #TODO
eigvals
```

[6]:
```
array([2.93035378, 0.92740362, 0.14834223, 0.02074601])
```

[7]:
```python
# Ratio of explained variance per PC
explained_variances = []
for i in range(len(eigvals)):
    explained_variances.append(eigvals[i] / np.sum(eigvals))

print(np.sum(explained_variances), '\n', explained_variances)
```

```
1.0
 [0.7277045209380132, 0.2303052326768066, 0.03683831957627393,
0.005151926808906313]
```

첫 번째, 두 번째 PC가 이미 variance의 95% 이상을 설명함을 확인할 수 있다!

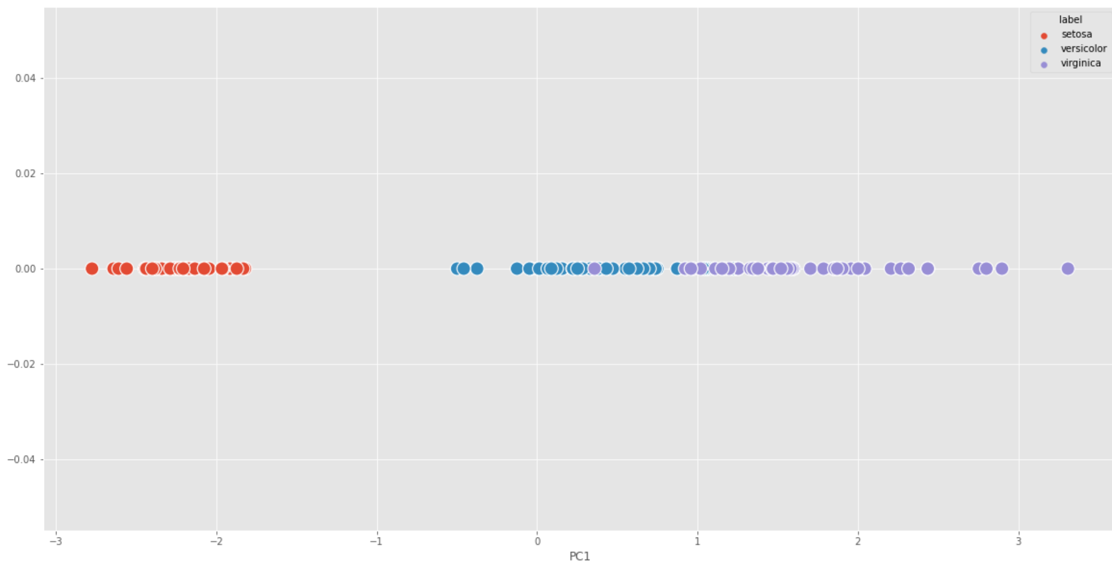
[8]:
```python
# Visualization (Embedding)
pc1 = np.dot(X_scaled, eigvecs[:,0]) #TODO
pc2 = np.dot(X_scaled, eigvecs[:,1]) #TODO
res = pd.DataFrame(pc1, columns=['PC1'])
res['PC2'] = pc2
res['label'] = label
res.head()
```

[8]:

| | PC1 | PC2 | label |
|---|---|---|---|
| 0 | -2.264542 | -0.505704 | setosa |
| 1 | -2.086426 | 0.655405 | setosa |
| 2 | -2.367950 | 0.318477 | setosa |
| 3 | -2.304197 | 0.575368 | setosa |
| 4 | -2.388777 | -0.674767 | setosa |

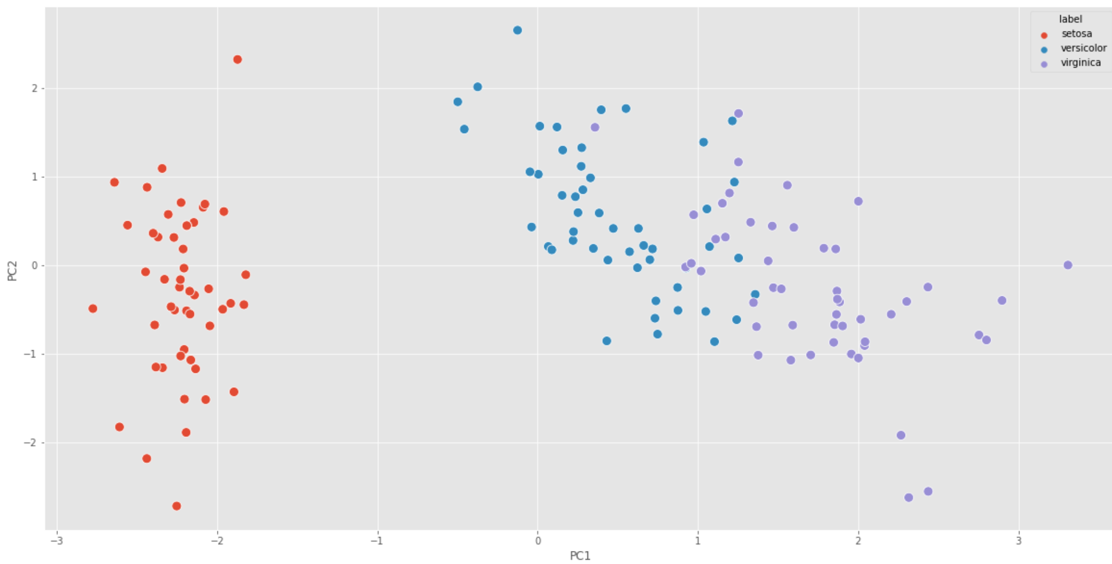
[9]:
```python
# Projection on 1-dim subspace
plt.figure(figsize=(20, 10))
sns.scatterplot(res['PC1'], [0] * len(res), hue=res['label'], s=200)
```

[9]:
```
<AxesSubplot:xlabel='PC1'>
```




[10]:
```python
# Projection on 2-dim subspace
plt.figure(figsize=(20, 10))
sns.scatterplot(res['PC1'], res['PC2'], hue=res['label'], s=100)
```

[10]: <AxesSubplot:xlabel='PC1', ylabel='PC2'>



### 2.0.3 Shortcut

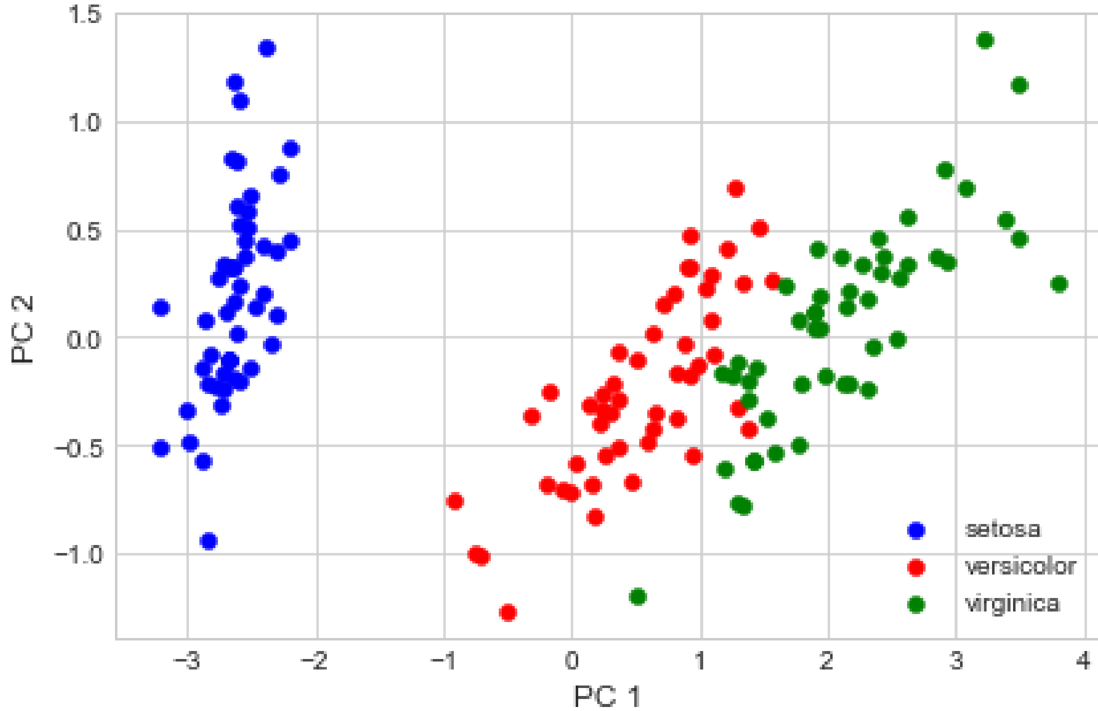
[11]:
```python
from sklearn.decomposition import PCA as sklearnPCA

sklearn_pca = sklearnPCA(n_components=2)
projection = sklearn_pca.fit_transform(X, y=label)

sklearn_pca.explained_variance_ratio_
```

[11]:
```
array([0.92461621, 0.05301557])
```

[12]:
```python
sklearn_pca.components_
```

[12]:
```
array([[ 0.36158968, -0.08226889,  0.85657211,  0.35884393],
       [ 0.65653988,  0.72971237, -0.1757674 , -0.07470647]])
```

[13]:
```python
with plt.style.context('seaborn-whitegrid'):
    plt.figure(figsize=(6, 4))
    for lab, col in zip(('setosa', 'versicolor', 'virginica'),
                        ('blue', 'red', 'green')):
        plt.scatter(projection[label==lab, 0],
                    projection[label==lab, 1],
                    label=lab,
                    c=col)
    plt.xlabel('PC 1')
    plt.ylabel('PC 2')
    plt.legend(loc='lower right')
    plt.tight_layout()
    plt.show()
```


1.5
1.0
0.5
0.0
−0.5
−1.0
PC 2
−3
−2
−1
0
1
2
3
4
PC 1
setosa
versicolor
virginica