Statistical Machine Learning

7주차

담당 : 11기 명재성

Feature Selection and Extraction

- Feature Selection

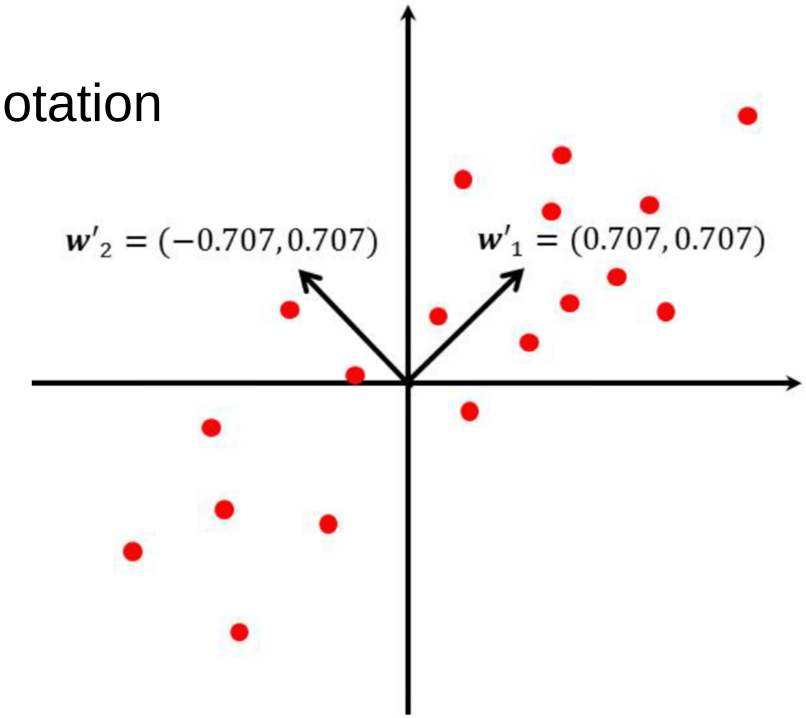
  - Subset selection, Stepwise method, LASSO, Least Angle Regression etc..

- Feature Extraction (Dimension Reduction)

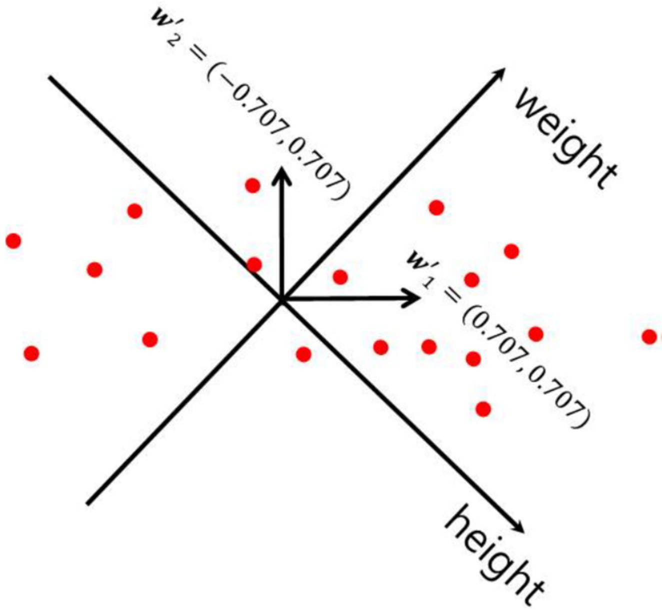
  - Principal Component Analysis, Partial Least Square, Discriminant Analysis, Factor Analysis, Latent Class Analysis, etc..

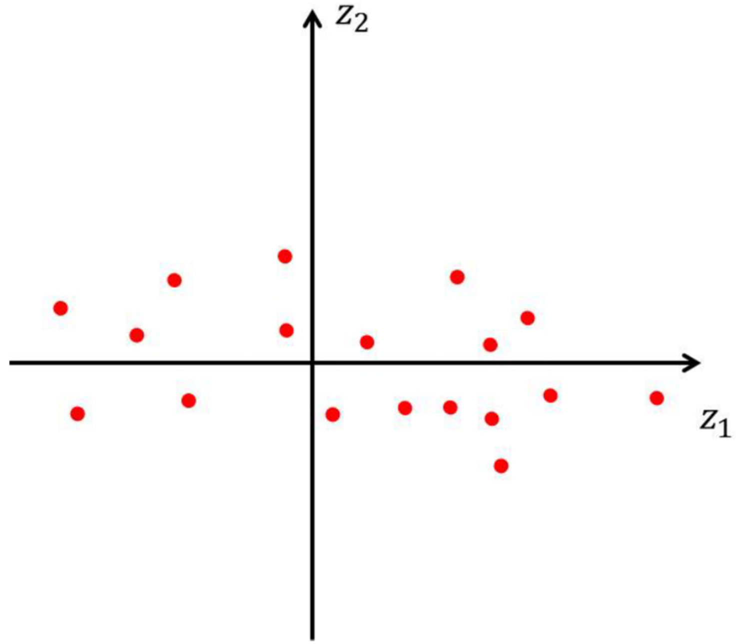
Principal Component Analysis

- Matrix Multiplication : Rotation



Principal Component Analysis





Principal Component Analysis

- Eigenvalues, Eigenvectors

$$A\mathbf{v} = \lambda\mathbf{v} \qquad \text{for} \quad \mathbf{v} \neq \mathbf{0}$$

Principal Component Analysis

- Eigenvalues, Eigenvectors

$$A\mathbf{v} = \lambda\mathbf{v} \qquad \text{for} \quad \mathbf{v} \neq \mathbf{0}$$

- If $A_{n\times n}$ is symmetric, then
  1. A has exactly n (not necessarily distinct) eigenvalues.
  2. There exists a set of n eigenvectors, one for each eigenvalue, that are mutually orthogonal.

Principal Component Analysis

- Eigenvalue Decomposition (= Spectral Decomposition)

$$
\begin{aligned}
AU &= A[\mathbf{u}_1\ \mathbf{u}_2\ \cdots\ \mathbf{u}_n] \\
&= [A\mathbf{u}_1\ A\mathbf{u}_2\ \cdots\ A\mathbf{u}_n] = [\lambda_1\mathbf{u}_1\ \lambda_2\mathbf{u}_2\ \cdots\ \lambda_n\mathbf{u}_n] \\
&= [\mathbf{u}_1\ \mathbf{u}_2\ \cdots\ \mathbf{u}_n]\begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix} \\
&= UD
\end{aligned}
$$

Principal Component Analysis

- Eigenvalue Decomposition (= Spectral Decomposition)

$$AU = UD \quad \Rightarrow \quad A = UDU^T$$

Principal Component Analysis

- Singular Value Decomposition

$$A_{n\times p} = \mathrm{U}_{n\times n}\Lambda_{n\times p}\mathrm{V}^{T}{}_{\mathrm{p}\times p}$$

$$(\mathrm{A}^{T}A)\mathrm{V} = \mathrm{V}D \quad \Rightarrow \quad (\mathrm{A}^{T}A) = \mathrm{V}D\mathrm{V}^{T}$$

$$(A\mathrm{A}^{T})U = UD' \quad \Rightarrow \quad (A\mathrm{A}^{T}) = UD'U^{T}$$

Principal Component Analysis

- Singular Value Decomposition

$$\mathrm{X}_{n\times p} = Z_{s_{n\times p}} D^{\frac{1}{2}}{}_{\mathrm{p}\times p} \mathrm{U}^{T}{}_{\mathrm{p}\times p}$$

$$X \sim (0, \Sigma)$$

$$\Sigma = \frac{1}{n} X^T X = \frac{1}{n} U D^{\frac{1}{2}} Z_s Z_s^T D^{\frac{1}{2}} U^T = U D U^T$$

Principal Component Analysis

- We want to find principal components which maximize the variances.

$$Var(\mathbf{z}_1) = Var(X\mathbf{u}_1) = \mathbf{u}_1^T Var(X)\mathbf{u}_1 = \mathbf{u}_1^T \Sigma \mathbf{u}_1$$

$$\max_{\mathbf{u}_1} \quad \mathbf{u}_1^T \Sigma \mathbf{u}_1$$

subject to $\mathbf{u}_1^T \mathbf{u}_1 = 1$

$\mathbf{u}_1 \Rightarrow$ eigenvector of Σ corresponding to λ_1

Principal Component Analysis

- From the fact that $\sum Var(X_i) = \sum Var(Z_i)$,

$$Var(\mathbf{z}_1) = \mathbf{u}_1^T \Sigma \mathbf{u}_1 = \mathbf{u}_1^T \lambda_1 \mathbf{u}_1 = \lambda_1 \mathbf{u}_1^T \mathbf{u}_1 = \lambda_1$$

$$Var(\mathbf{z}_\mathrm{p}) = \mathbf{u}_\mathrm{p}^T \Sigma \mathbf{u}_\mathrm{p} = \mathbf{u}_\mathrm{p}^T \lambda_\mathrm{p} \mathbf{u}_\mathrm{p} = \lambda_\mathrm{p} \mathbf{u}_\mathrm{p}^T \mathbf{u}_\mathrm{p} = \lambda_\mathrm{p}$$

$$\sum Var(X_i) = \lambda_1 + \cdots + \lambda_\mathrm{p}$$

Principal Component Analysis

- If we choose first k ($< p$) principal components

$$\mathrm{X}_{n\times p} \approx Z_{s_{n\times k}} D^{\frac{1}{2}}{}_{\mathrm{k}\times k} \mathrm{U}^{T}{}_{\mathrm{k}\times p}$$

$$\Rightarrow \quad Dimension\ Reduction$$

Principal Component Analysis

```
from sklearn.datasets import fetch_olivetti_faces
import matplotlib.pyplot as plt

faces_all = fetch_olivetti_faces()
K = 7 # 7번 인물의 사진만 선택
faces = faces_all.images[faces_all.target == K]
X3 = faces_all.data[faces_all.target == K]
print(faces.shape)
print(X3.shape)

N = 2
M = 5
fig = plt.figure(figsize=(10, 5))
plt.subplots_adjust(top=1, bottom=0, hspace=0, wspace=0.05)
for i in range(N):
  for j in range(M):
    k = i * M + j
    ax = fig.add_subplot(N, M, k+1)
    ax.imshow(faces[k], cmap=plt.cm.bone)
    ax.grid(False)
    ax.xaxis.set_ticks([])
    ax.yaxis.set_ticks([])
plt.suptitle("faces of the figure")
plt.tight_layout()
plt.show()
```

```
(10, 64, 64)
(10, 4096)
```

Principal Component Analysis

faces of the figure

Principal Component Analysis

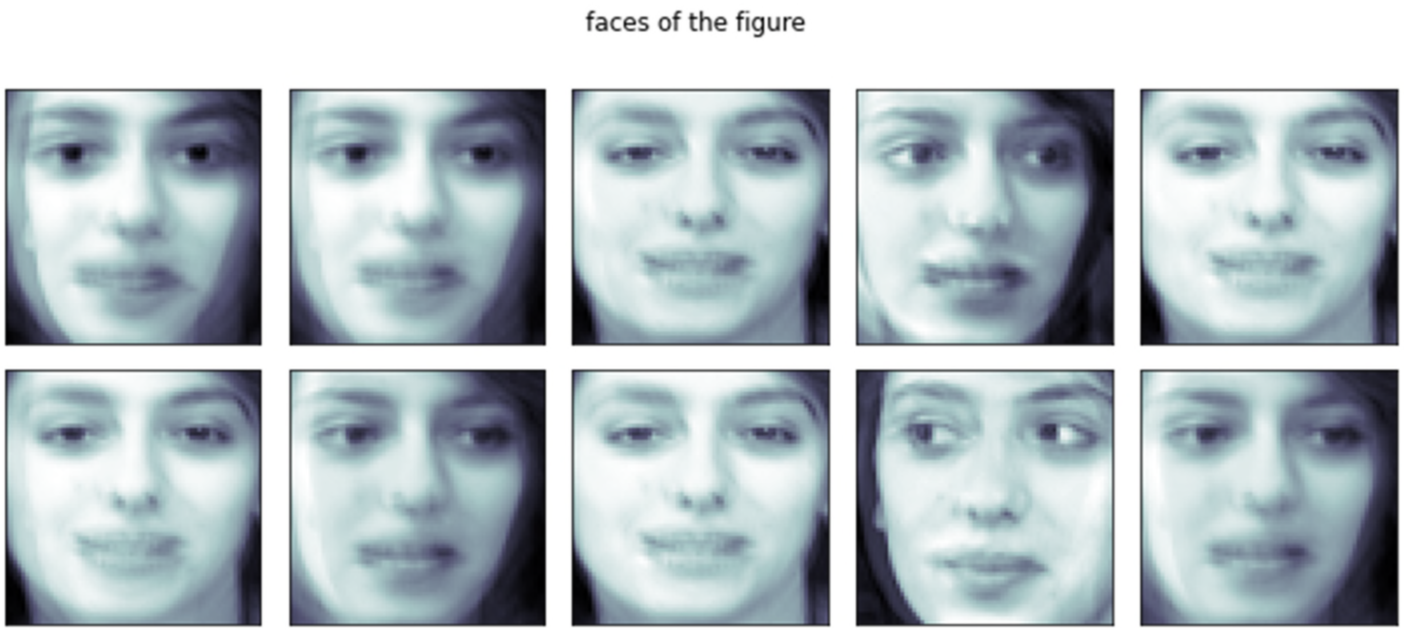
```
from sklearn.decomposition import PCA

pca3 = PCA(n_components=2)
W3 = pca3.fit_transform(X3)
X32 = pca3.inverse_transform(W3)
print(W3.shape)
print(X32.shape)
```

```
(10, 2)
(10, 4096)
```

```
N = 2
M = 5
fig = plt.figure(figsize=(10, 5))
plt.subplots_adjust(top=1, bottom=0, hspace=0, wspace=0.05)
for i in range(N):
  for j in range(M):
    k = i * M + j
    ax = fig.add_subplot(N, M, k+1)
    ax.imshow(X32[k].reshape(64, 64), cmap=plt.cm.bone)
    ax.grid(False)
    ax.xaxis.set_ticks([])
    ax.yaxis.set_ticks([])
plt.suptitle("faces of the figure")
plt.tight_layout()
plt.show()
```

Principal Component Analysis



reference

자료

19-2 STAT424 통계적 머신러닝 - 박유성 교수님

교재

파이썬을 이용한 통계적 머신러닝 (2020) – 박유성

ISLR (2013) - G. James, D. Witten, T. Hastie, R. Tibshirani

The elements of Statistical Learning (2001) - J. Friedman, T. Hastie, R. Tibshirani

Hands on Machine Learning (2017) - Aurelien Geron