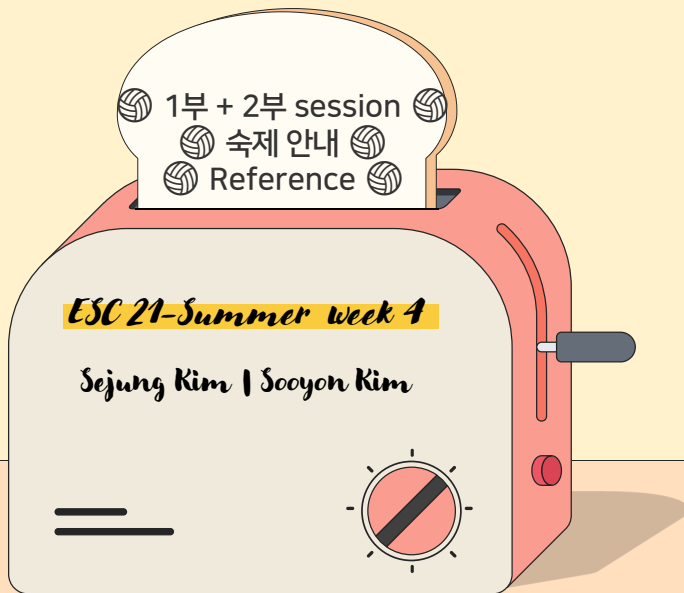


Principal Component Analysis



Principal Component Analysis

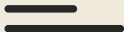


1부 암기빵 남남

*Introduction of PCA
Mathematical Background
PCA algorithm
PLS algorithm*

Contents

Principal Component Analysis





Introduction of Principal Component Analysis

Dimensionality Reduction

변수 선택 (feature selection)

기존의 설명 변수들 중 분석 목적에 부합하는 소수의
예측 변수만을 선택하는 방식

ex. Lasso ...

변수 추출 (feature extraction)

기존의 설명 변수들의 변환을 통해 새로운 예측 변수를
추출하는 방식

ex. PCA, PLS ...

장점 : 변수의 개수를 크게 줄일 수 있음
단점 : 추출된 변수들의 해석이 어려움

변수 추출 (feature extraction)

PCA (Principal Component Analysis, 주성분 분석) : **unsupervised** feature extraction

PLS (Partial Least Squares, 부분 최소 제곱법) : **supervised** feature extraction

종속변수 Y 존재 여부

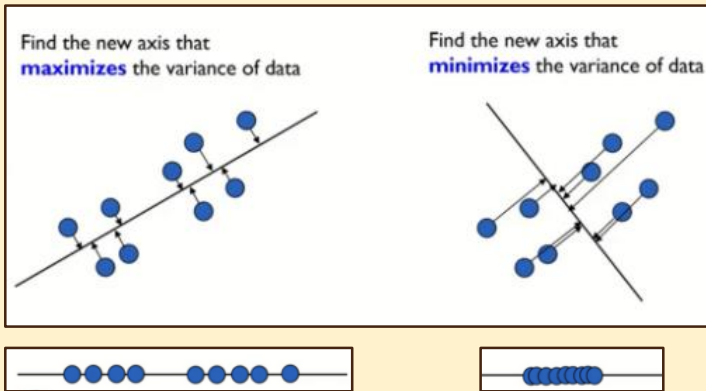
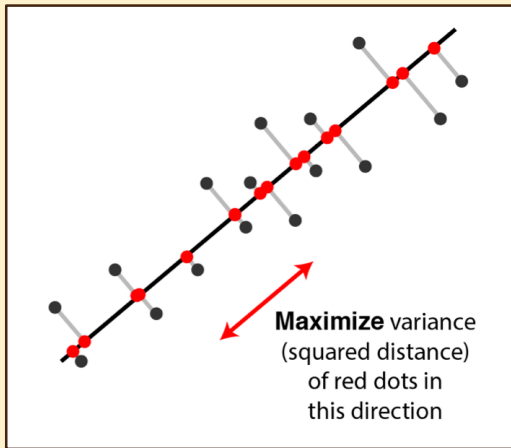


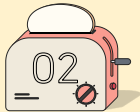
Introduction of Principal Component Analysis

PCA (Principal Component Analysis, 주성분 분석)

N개의 관측치와 d개의 변수로 구성된 데이터를 상관관계가 없는 k개의 변수로 구성된 데이터로 요약하는 방식
이 때, k개의 변수는 기존 변수의 선형 조합으로 생성

원래의 데이터의 분산을 최대한 보존하는 새로운 축을 찾고, 그 축에 데이터를 사영(projection)시키는 방법





Mathematical Background for PCA

Eigenvector & Eigenvalue

정사각행렬 A 에 대하여 $Ax = \lambda x$ 를 만족하는 상수 λ 와 벡터 x 를 각각 eigenvalue, eigenvector라고 한다.

- 벡터에 행렬을 곱하는 것은 선형 변환하는 것. 고유벡터는 이러한 변환에 의해 방향이 변하지 않는 벡터를 의미.
- 한 eigenvalue에 대응하는 eigenvector는 유일하지 않다.
- A 가 $n \times n$ 행렬이라면 선형독립인 eigenvector를 최대 n 개까지 가질 수 있다.

Spectral Decomposition

$$A \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix} = \begin{pmatrix} Ax_1 & \dots & Ax_n \end{pmatrix} = \begin{pmatrix} \lambda_1 x_1 & \dots & \lambda_n x_n \end{pmatrix} = \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix} \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}$$

$$AX = XD$$

$$A \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix} = \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}$$

$$\Rightarrow A = XDX^{-1}$$

$$A = \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix} \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix}^{-1}$$

- 정사각행렬에 한해서만 가능



Mathematical Background for PCA

Singular Vector Decomposition

The *singular value decomposition* (SVD) of the centered input matrix \mathbf{X} gives us some additional insight into the nature of ridge regression. This decomposition is extremely useful in the analysis of many statistical methods. The SVD of the $N \times p$ matrix \mathbf{X} has the form

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T. \quad (3.45)$$

$$\mathbf{X} = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_n \mathbf{u}_n \mathbf{v}_n^T$$

- 정사각행렬이 아니어도 가능
- 한 행렬을 여러 행렬의 합으로 표현
- \mathbf{U} 와 \mathbf{V} 는 orthogonal matrix
- 이 중 가장 중요한 q 개의 layer를 더해 행렬 \mathbf{X} 를 요약할 수 있음

Covariance matrix

$$\Sigma = \text{Var}(\mathbf{X}) = \Gamma \Lambda \Gamma^T$$

- 공분산행렬의 대각성분은 각 변수의 분산과 같으며, 비대각성분은 대응하는 두 변수의 공분산과 같음

$$\mathbf{S} = \frac{1}{n-1} \mathbf{X}^T \mathbf{X}$$

- 표본 공분산행렬은 \mathbf{S} 이며, 때에 따라 n 이나 $n-1$ 이 달라짐

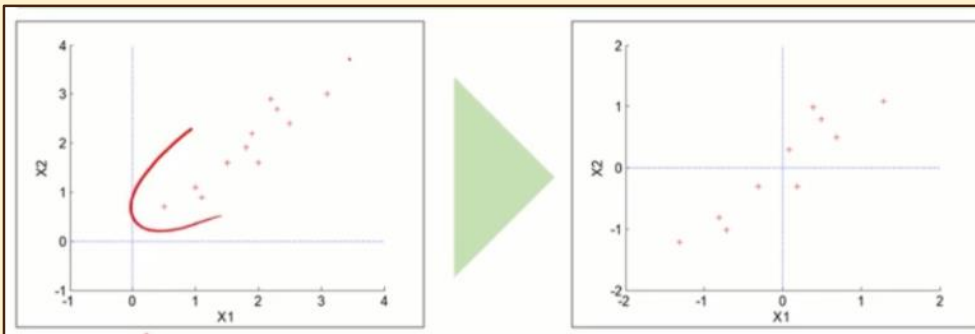


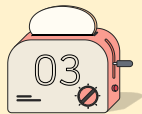
PCA algorithm

Goal: 가장 큰 **variance**를 가지는 주성분을 만드는 새로운 벡터를 찾자!

Assumptions

- 기존 변수들을 선형 결합하여 새로운 변수를 만드는 것
 - 데이터들은 centered and scaled
- (centering과 scaling을 한다고 하더라도 데이터 자체의 구조는 변하지 않음)
→ scale invariant





PCA algorithm

Goal : 가장 큰 variance를 가지는 주성분을 만드는 새로운 벡터를 찾자 !

$$\delta^\top X = \sum_{j=1}^p \delta_j X_j, \quad \text{such that} \quad \sum_{j=1}^p \delta_j^2 = 1. \quad (11.1)$$

The weighting vector $\delta = (\delta_1, \dots, \delta_p)^\top$ can then be optimised to investigate and to detect specific features. We call (11.1) a **standardised linear combination** (SLC). Which SLC should we choose? One aim is to **maximise the variance of the projection $\delta^\top X$** , i.e. to choose δ according to

$$\max_{\{\delta: \|\delta\|=1\}} \text{Var}(\delta^\top X) = \max_{\{\delta: \|\delta\|=1\}} \delta^\top \text{Var}(X) \delta. \quad (11.2)$$

The interesting “**directions**” of δ are found through the **spectral decomposition** of the **covariance matrix**. Indeed, from Theorem 2.5, the **direction δ** is given by the **eigenvector γ_1** corresponding to the **largest eigenvalue λ_1** of the covariance matrix $\Sigma = \text{Var}(X)$.

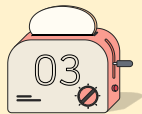


PCA algorithm

가장 큰 variance가 eigenvalue라는 것을 증명해봅시다 !

$$\max_{\{\delta: \|\delta\|=1\}} \text{Var}(\delta^\top X) = \max_{\{\delta: \|\delta\|=1\}} \delta^\top \text{Var}(X) \delta. \quad (11.2)$$

The interesting “directions” of δ are found through the spectral decomposition of the covariance matrix. Indeed, from Theorem 2.5, the direction δ is given by the eigenvector γ_1 corresponding to the largest eigenvalue λ_1 of the covariance matrix $\Sigma = \text{Var}(X)$.



PCA algorithm

In practice the PC transformation has to be replaced by the respective estimators: μ becomes \bar{x} , Σ is replaced by S , etc. If g_1 denotes the first eigenvector of S , the first principal component is given by $y_1 = (\mathcal{X} - \mathbf{1}_n \bar{x}^\top) g_1$. More generally if $S = \mathcal{G} \mathcal{L} \mathcal{G}^\top$ is the spectral decomposition of S , then the PCs are obtained by \hookrightarrow eigen vector of S

$$\mathcal{Y} = (\underbrace{\mathcal{X} - \mathbf{1}_n \bar{x}^\top}_{\hookrightarrow \text{centred } X}) \underbrace{\mathcal{G}}_{\substack{\text{1.2차 행렬,} \\ \rightarrow \text{eigen vector}}} \quad (11.10)$$

Summary !!!

Maximising the variance of $\delta^\top X$ leads to the choice $\delta = \gamma_1$, the eigenvector corresponding to the largest eigenvalue λ_1 of $\Sigma = \text{Var}(X)$.

This is a projection of X into the one-dimensional space, where the components of X are weighted by the elements of γ_1 . $Y_1 = \gamma_1^\top (X - \mu)$ is called the first principal component (PC).

This projection can be generalised for higher dimensions. The PC transformation is the linear transformation $Y = \Gamma^\top (X - \mu)$, where $\Sigma = \text{Var}(X) = \Gamma \Lambda \Gamma^\top$ and $\mu = \mathbf{E} X$.

Y_1, Y_2, \dots, Y_p are called the first, second, ..., and p -th PCs.

- 실제 sample에서는 모집단의 특성을 다 알 수 없기 때문에 $\mu \rightarrow \bar{x}$, $\text{var}(X) \rightarrow S$ 로 대체



PCA algorithm

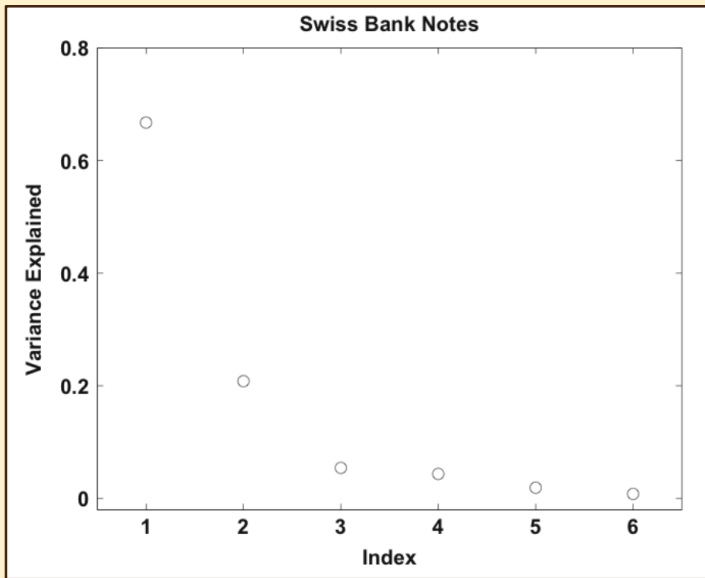
Interpretation of PC

$$\psi_q = \frac{\sum_{j=1}^q \lambda_j}{\sum_{j=1}^p \lambda_j} = \frac{\sum_{j=1}^q \text{Var}(Y_j)}{\sum_{j=1}^p \text{Var}(Y_j)}.$$

- 분산비를 이용해서 몇 개의 주성분을 이용하는 것이 좋을지 판단할 수 있음

- 선택 방식 1 : 고유 값 감소율이 유의미하게 낮아지는 Elbow point에 해당하는 주성분 수를 선택
- 선택 방식 2 : 일정 수준 이상의 분산 비를 보존하는 최소의 주성분을 선택 (보통 70%)

분산 비 : q차원으로 차원 축소하였을 때 그들의 주성분들이 original variable X를 얼마나 잘 설명할 수 있는지 나타내는 식





PCA algorithm

Interpretation of PC

$$\rho_{X_i Y_j} = \frac{\gamma_{ij} \lambda_j}{(\sigma_{X_i X_i} \lambda_j)^{1/2}} = \gamma_{ij} \left(\frac{\lambda_j}{\sigma_{X_i X_i}} \right)^{1/2} \quad (11.14)$$

↳ λ_j 의 값은 eigen value

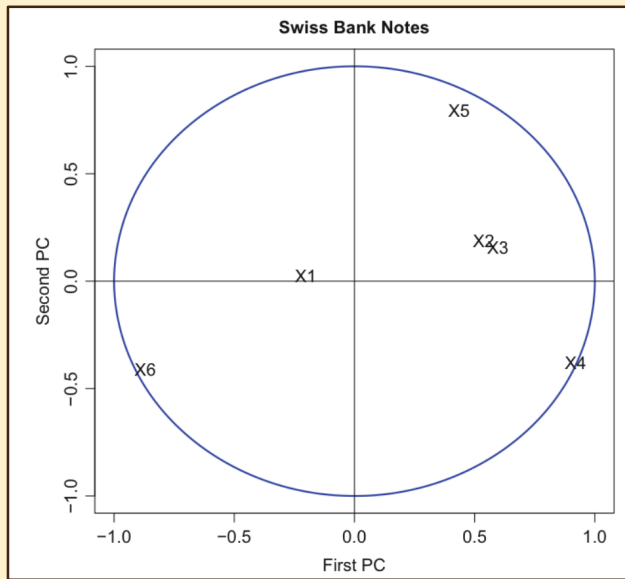
Using actual data, this of course translates into

$$r_{X_i Y_j} = g_{ij} \left(\frac{\ell_j}{s_{X_i X_i}} \right)^{1/2} \quad (11.15)$$

↑
eigen vector ↑
eigen value

The correlations can be used to evaluate the relations between the PCs Y_j where $j = 1, \dots, q$, and the original variables X_i where $i = 1, \dots, p$. Note that

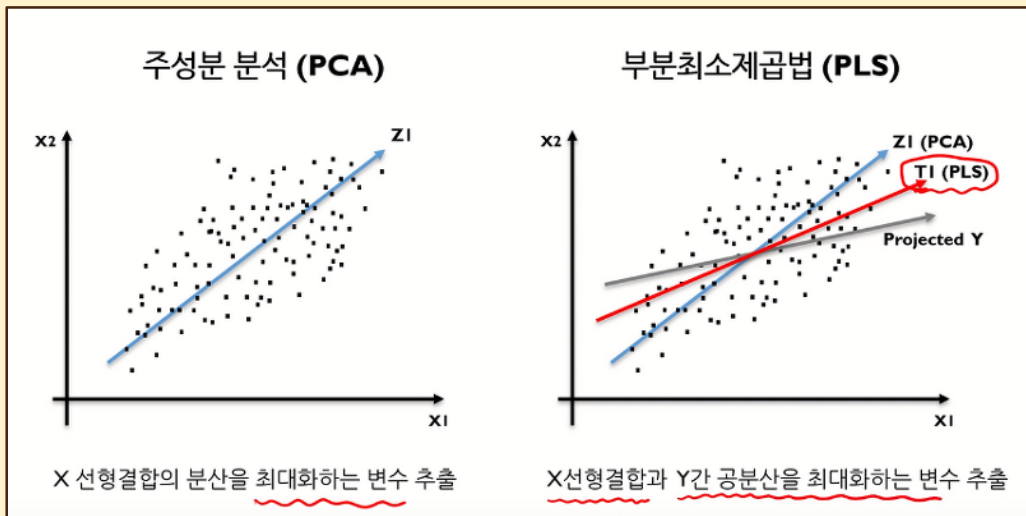
From (11.16) it obviously follows that $r_{X_i Y_1}^2 + r_{X_i Y_2}^2 \leq 1$ so that the points are always inside the circle of radius 1. In the bank notes example, the variables X_4 , X_5 and X_6 correspond to correlations near the periphery of the circle and are thus well explained by the first two PCs. Recall that we have interpreted the first PC as being





PLS algorithm

Supervised feature extraction PLS



- 종속 변수 Y와 상관관계가 높은 q개의 선형 조합 변수를 추출하는 방식
- 선형조합으로 추출된 변수가 설명하지 못하는 부분에 대해 지속적으로 최소제곱법을 사용하는 것에서 유래
 - 추출된 변수가 PCA에서는 반영하지 못했던 Y와의 상관관계를 반영하는 특징
 - 적은 수의 추출된 변수로 효율적인 모델 구축 가능



PLS algorithm



Goal : 가장 큰 상관관계를 가지는 선형 결합을 만드는 벡터를 찾자!

X : 기존의 설명 변수

Y : 기존의 종속 변수

t : 기존 설명변수들의 선형 결합

w : 기존 설명 변수들에게 부여되는 가중치

$$t = Xw$$

$$\begin{aligned} \text{cov}(t, Y) &= \frac{\text{cov}(t, Y)}{\sqrt{\text{var}(t)}\sqrt{\text{var}(Y)}} \sqrt{\text{var}(t)}\sqrt{\text{var}(Y)} \\ &= \text{corr}(t, Y)\sqrt{\text{var}(t)}\sqrt{\text{var}(Y)} \end{aligned}$$

$$\therefore \text{Maximize } \text{cov}(t, Y) \propto \text{Maximize } \text{corr}(t, Y)\text{var}(t)$$



PLS algorithm



Then, 가중치는 어떻게 설정해야 할까?

$$\text{Maximize } \text{cov}(t, Y) = \text{cov}(Xw, Y) = E[(Xw - E[Xw])(Y - E[Y])] = E(Xw \cdot Y)$$



내적인 값이 최대가 되도록 만들어야해!

$$E(Xw \cdot Y) = \frac{1}{n} \sum_1^n (Xw)_i \cdot Y_i = \frac{1}{n} (Xw)^T Y = \frac{1}{n} w^T (X^T Y)$$

$$* w^T (X^T Y) = \|w\| \|X^T Y\| \cos \theta$$

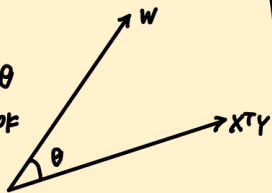
⇒ w와 $X^T Y$ 가 같은 방향으로 설정되어야

θ가 0이 되며 내적값이 최대화

될 수 있음!

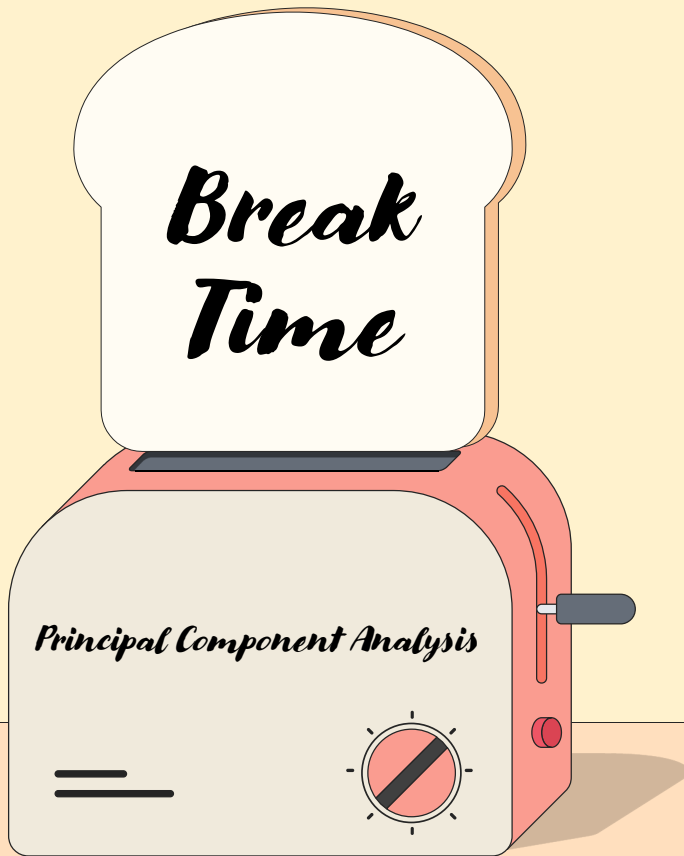
$$\therefore w = X^T Y \left(\frac{X^T Y}{\|X^T Y\|} \right)$$

↳ 크기를 1로 맞춰주기!!



*Break
Time*

Principal Component Analysis



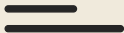
2부 암기빵 남남

*PCA in terms of
Reconstruction error
Principal Components
PCR*

*PCA In a nutshell
Using PCA*

Contents

Principal Component Analysis





PCA in terms of Reconstruction Error

Linear Projection \leftrightarrow Reconstruction

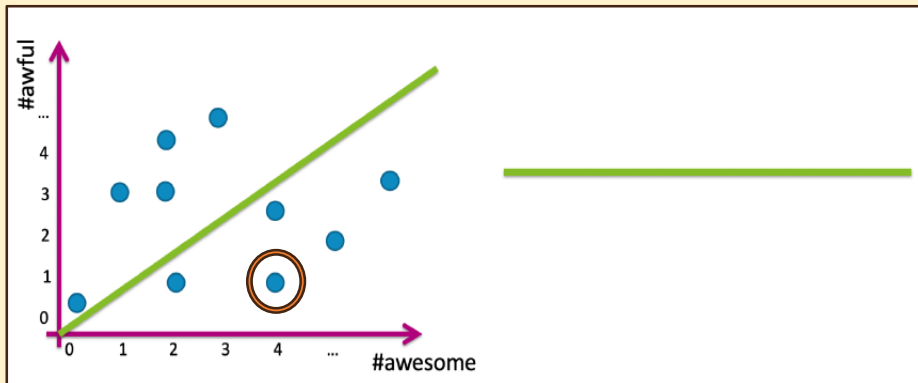
- PCA uses a linear projection from d-dim data to q-dim data ($q < d$)
- Choose projection that minimizes reconstruction error
- Reconstruction error : the information lost when you "undo" the projection
- using PCs and projected data, you should be able to reconstruct the data well on the original data!
(이 말이 곧 PC가 original data의 정보를 잘 보존하고 있다는 의미)
- Choose u_1, \dots, u_q such that minimizes $\|\hat{x} - x\|_2^2$

$$\hat{x}_i = \bar{X} + \sum_{j=1}^q z_{ij} u_j$$



PCA in terms of Reconstruction Error

Computing Reconstruction Error



$$z = \text{projection of } x \text{ on } u = \frac{u^T x}{||u||^2} = u^T x \quad (||u||^2 = 1)$$

Let $u = (4, 3)$. Then $x = (4, 1) \rightarrow z = 4 * 4 + 3 * 1 = 19$

Reconstruction : $\hat{x} = zu$



PCA in terms of Reconstruction Error

Computing Reconstruction Error



$$z_1 = \text{projection of } x \text{ on } u_1 = u_1^T x$$

$$z_2 = \text{projection of } x \text{ on } u_2 = u_2^T x$$

$$\hat{x} = z_1 u_1 + z_2 u_2$$

- PCs should be "perpendicular" to each other so that we can get maximum amount of new information (i.e. capture more variation)
- In this case, $d=q=2$. Thus, reconstruction error is 0.

We can perfectly capture all the information from original data.

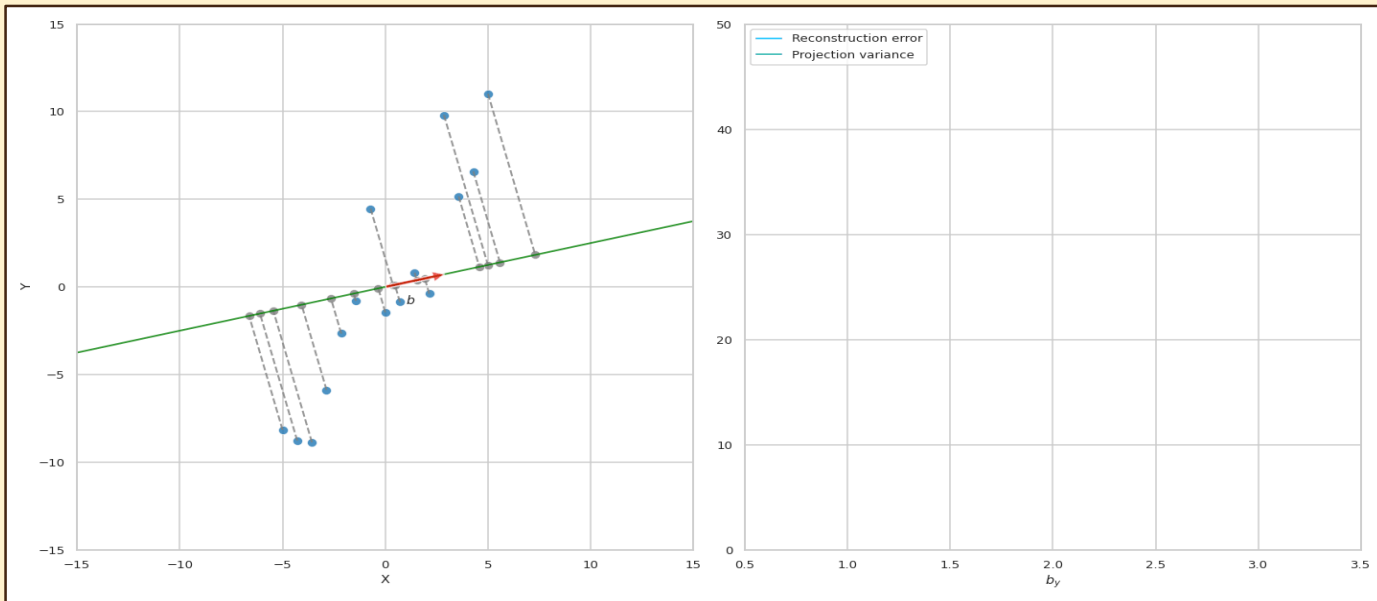
- In fact, this is nothing but rotation / orthogonalization (change of basis)

 visualization : <https://setosa.io/ev/principal-component-analysis/>



PCA in terms of Reconstruction Error

- maximizing variance = minimizing reconstruction error

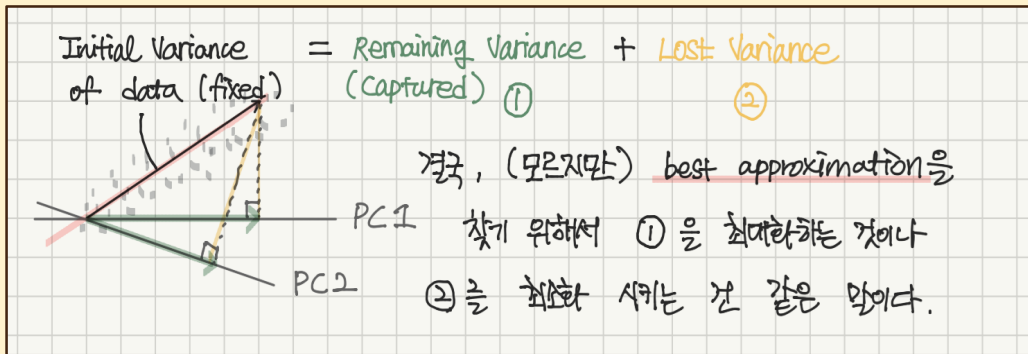




PCA in terms of Reconstruction Error

Geometry : What does "minimizing reconstruction error" mean?

① 분산 최대화 → 공분산 행렬 → $S = \frac{1}{n-1} X^T X$





Principal Components [ESL 14.5.1]

What is Principal Components?

- sequences of projections of data, which are mutually uncorrelated and ordered in variance
- provides a sequence of best linear approximations to the data of rank q !

$$f(\lambda) = \mu + V_q \lambda$$

☐ f : rank q linear model of representing X of dimension $(n \times d)$
: representation of affine hyperplane of rank q

☐ μ : location vector in R^d

☐ V_q : $d \times q$ matrix with q orthogonal unit vectors as columns

☐ λ : q vectors of parameters (일종의 가중치)

Fitting $f(\lambda)$ to the data by Least Squares amounts to minimizing reconstruction error.

$$\min_{\mu, \{\lambda_i\}, V_q} \sum_{i=1}^n \|x_i - \mu - V_q \lambda_i\|^2$$



Principal Components

Deriving equation (continued)

$$\min_{\mu, \{\lambda_i\}, V_q} \sum_{i=1}^n ||x_i - \mu - V_q \lambda_i||^2$$

→ partial optimization by solving normal equation (회귀분석에서 β coefficient 구하는 것처럼!)

$$\hat{\mu} = \bar{x}$$

$$\hat{\lambda} = V_q^T (x_i - \bar{x})$$

Then, all we need to find is V_q^T !

$$\min_{V_q} \sum_{i=1}^n ||(x_i - \bar{x}) - V_q V_q^T (x_i - \bar{x})||^2$$

$$V_q V_q^T = H_q$$

(projection matrix; idempotent)

Let $\tilde{x}_i = x_i - \bar{x}$ be centered data

$H_q \tilde{x}_i$: rank q reconstruction

(i.e. orthogonal projection of x onto the subspace spanned by the columns of V_q)



Principal Components

Deriving equation (continued)

$$\min_{V_q} \sum_{i=1}^n ||(x_i - \bar{x}) - V_q V_q^T (x_i - \bar{x})||^2$$

Given above, we can solve V_q by constructing SVD of $X = UDV^T$

- V_q consists of the first q columns of V
- columns of UD : principal components of X
- n optimal $\hat{\lambda}_i$ are given by the first q PCs (n rows of $n \times q$ matrix $U_q D_q$)

 orthogonal projection : Anton 7.7 / 7.9

M : column vector가 projection 시키고자 하는 부분공간 W 의 basis vector인 아무 행렬

$P = M(M^T M)^{-1} M^T$ where P is idempotent.

근데 여기서 M 의 열벡터들이 W 의 orthonormal basis들로 이루어져 있다면 $M^T M = I$



Principal Components

PCA in terms of Unsupervised Learning

In terms of finding latent representation of given data

☐ projection = encoding

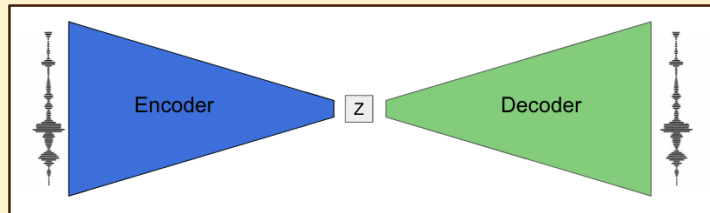
$$\rightarrow z = V^T x$$

☐ reconstruction = decoding

$$\rightarrow \hat{x} = Vz$$

☐ we want \hat{x} to be close to x (in L2 distance) \rightarrow our objective function

$$\rightarrow \mathbf{L}(V) = \frac{1}{n} \sum_{i=1}^n ||x_i - \text{decode}(\text{encode}(x_i; V); V)||_2^2$$



(reference) Probabilistic Machine Learning (Kevin P. Murphy) 20.1



PCR

ESL 3.5.1 Principal Components Regression

$$\hat{y}_{PCR} = \bar{y} + \sum_{j=1}^q \hat{\theta}_j z_j = \bar{y} + \sum_{j=1}^q \hat{\theta}_j X v_j = \bar{y} + X \hat{\beta}_{PCR}$$

$$\hat{\beta}_{PCR} = \sum_{j=1}^q \hat{\theta}_j v_j$$

$\hat{\theta}_j = \frac{\langle z_j, y \rangle}{\langle z_j, z_j \rangle}$: univariate regression coefficients

z_j : linear combination of original x_i 's

Thus, PCR is a sum of univariate regressions



PCR

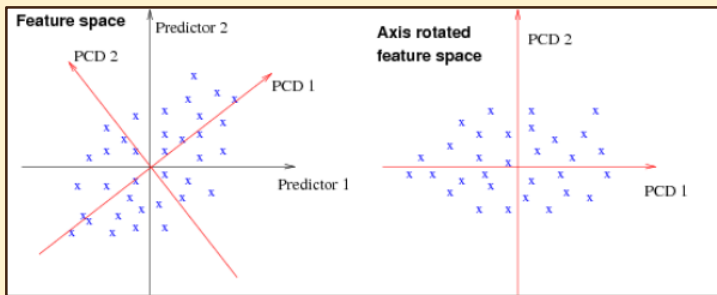
Ridge vs. PCR

Ridge : shrinks coefficients

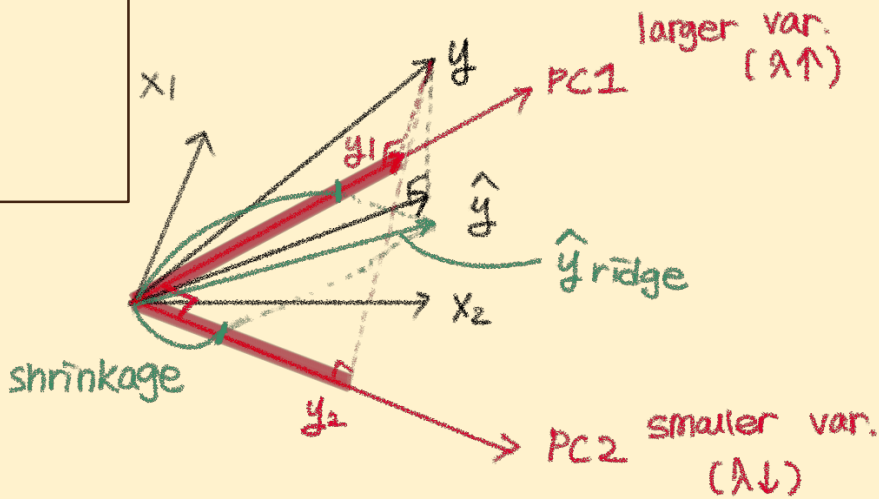
→ variance / eigenvalue 작은 부분 shrink (smooth)

$$\begin{aligned}\hat{\beta}_{ridge} &= (X^T X + \lambda I)^{-1} X^T Y \\ \hat{Y}_{ridge} &= X \hat{\beta}_{ridge} = X(X^T X + \lambda I)^{-1} X^T Y \\ &= U D (D D^T + \lambda I)^{-1} D U^T Y \\ &= \sum_{j=1}^q u_j \frac{d_j^2}{d_j^2 + \lambda} u_j^T Y\end{aligned}$$

$$\begin{aligned}\beta_1 &= \frac{\|y_1\|}{\|PC1\|} & \beta_1^{ridge} &= \frac{\|PC1\|^2}{\|PC1\|^2 + \lambda} \\ \beta_2 &= \frac{\|y_2\|}{\|PC2\|} & \beta_2^{ridge} &= \frac{\|PC2\|^2}{\|PC2\|^2 + \lambda}\end{aligned}$$



(reference) <https://youtu.be/onwOXVv8sks>





PCR

Ridge vs. PCR

PCA : discards $d-q$ components (predictors) with smallest eigenvalue

→ variance 작은 부분 아예 0으로 만들어버림!

$$X = UDV^T \rightarrow X_{PCA} = U_q D_q$$

$$\hat{Y} = X\hat{\beta} = X(X^T X)^{-1} X^T Y$$

$$\hat{Y}_{PCA} = U_q U_q^T Y = U \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & 0 & \\ & & & \ddots \\ & & & & 0 \end{bmatrix} U^T Y = \sum_{j=1}^q u_j 1 u_j^T Y + \sum_{j=q+1}^d u_j 0 u_j^T Y$$

Y를 PC들이 span하는 공간에 projection



PCR

Ridge vs. PCR : comparison

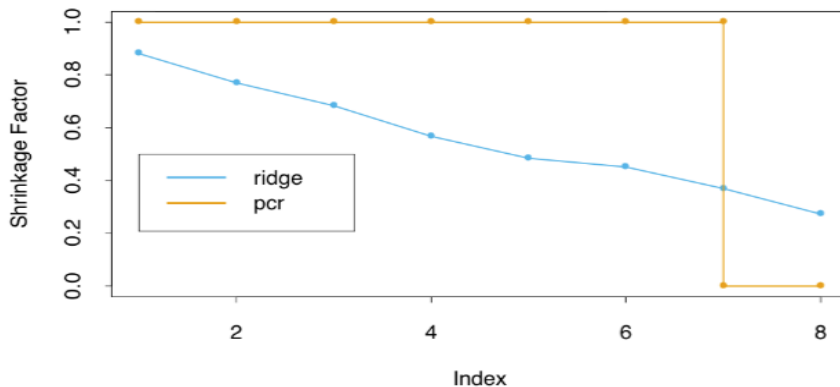


FIGURE 3.17. Ridge regression shrinks the regression coefficients of the principal components, using shrinkage factors $d_j^2/(d_j^2 + \lambda)$ as in (3.47). Principal component regression truncates them. Shown are the shrinkage and truncation patterns corresponding to Figure 3.7, as a function of the principal component index.



PCA in a nutshell

How PCA works (Algorithm)

Step 1. Recenter Data

$$\tilde{X} = X - \bar{X}$$

Step 2. Find the 1st PC as the direction which maximizes variance

$$v_1 = \operatorname{argmax} \sum_{i=1}^n z_i^2 = \operatorname{argmax} \sum_{i=1}^n (v^T \tilde{x}_i)^2 \quad \text{s.t. } \|v\| = 1$$

Step 3. Find the kth PC

3-1) Subtract k-1 PCs from centered data

$$\widetilde{\tilde{x}_{i(k)}} = \tilde{x}_i - \sum_{j=1}^{k-1} (v_j^T \tilde{x}_i) v_j$$

3-2) Find the direction which maximizes the variance of 3-1

$$v_k = \operatorname{argmax} \sum_{i=1}^n z_i^2 = \operatorname{argmax} \sum_{i=1}^n (v_k^T \widetilde{\tilde{x}_{i(k)}})^2 \quad \text{s.t. } \|v\| = 1$$



PCA in a nutshell

How PCA works (Linear Algebra)

Step 1. Recenter Data

$$\tilde{X} = X - \bar{X}$$

Step 2. Compute covariance matrix

$$\Sigma = \frac{1}{n} X^T X$$

Step 3. Find basis : compute eigenvectors of covariance matrix

Select q eigenvectors u_1, \dots, u_q with largest eigenvalues

Step 4. Project data onto principal directions

$$z_1 = u_1^T \tilde{x} = u_{1,1} \widetilde{x}_{,1} + \dots + u_{1,d} \widetilde{x}_{,d}$$

...

<= n x 1 vectors

$$z_q = u_q^T \tilde{x} = u_{q,1} \widetilde{x}_{,1} + \dots + u_{q,d} \widetilde{x}_{,d}$$

$$X_{n \times d}, U_{d \times q} \Rightarrow Z_{n \times q} = XU$$

Identify which "directions" are most "important"



transform data to align in important direction

+

compress/project our data into a smaller space
by dropping the "directions" that are the "least important"





PCA in a nutshell



How PCA works (SVD)

Step 1. Recenter Data

$$\tilde{X} = X - \bar{X}$$

Step 2. Compute SVD of Data

$X = UDV^T$ where diagonal entries of D are eigenvalues of XX^T

Step 3. Goal of PCA is to find an orthogonal matrix V_q

that determines a change of variable such that $Z = XV_q = U_q D_q$

- columns of Z are ordered according to increasing variability
- columns of Z are uncorrelated
- Each column of Z is a linear combination of the columns of X
- Assumes more variability in a direction better explains the response.

📁 Why do we use SVD?

→ PCA = finding eigenvectors of $d \times d$ covariance matrix.

근데 $n < d$ 이면 $n \times n$ 인 XX^T 를 가지고 하는 operation이 더 빠르다!

Covariance matrix의 $X^T X$ 대신
 XX^T 를 써도 되는 이유?

-> $X^T X$ 와 XX^T 의 고유값들은 같다!



PCA in a nutshell

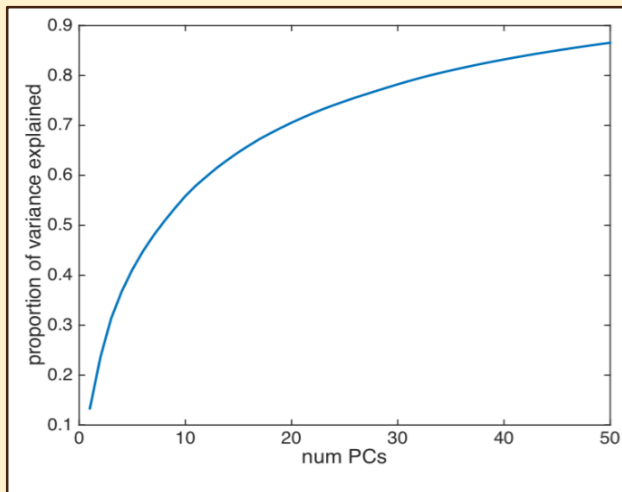


miscellaneous

☞ 그렇다면 내가 SVD/PCA를 통해서 얼마만큼의 정보를 capture했는지는 뭘로 할 수 있을까?

→ trace!

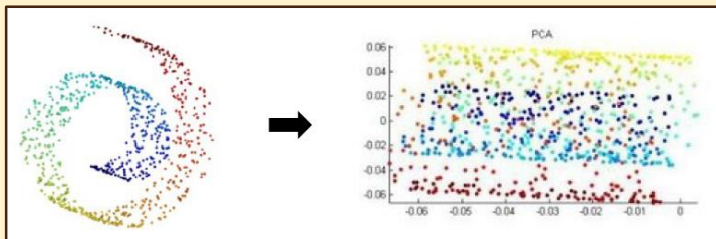
→ criterion for choosing how many PCs to keep = proportion of variance explained
(usually rule of thumb is elbow point! 🤖)





When should / should not I use PCA?

- ✓ 변수를 줄이고는 싶은데, predictor들의 구조도 모르고 뭘 drop해야 할지도 잘 모르겠을 때..
- ✓ You want to ensure your variables are independent of one another.
- ✗ You are not comfortable making your independent variables less interpretable.
(PCA 쓰려면 설명력은 포기해야...)
- ✗ PCA assumes there is a lower dimensional linear subspace that represents the data well.
(Doesn't work well with non-linear manifold) → solutions include t-SNE, Isomap, etc.
(PC들은 기존 변수들의 선형결합이기 때문에 비선형 data에서는 잘 작동하지 않을수도 있음.)



*Home Work
Let's go ~*

Principal Component Analysis






Home Work



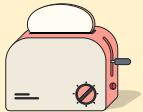
역시나 숙제는
수식 관련 한 문제,
코딩 관련 한 문제입니다!

 $\text{Var}(Y_j) = \lambda_j$

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



Lab Code의 @TODO 부분 채워서 iris data에 dimensionality reduction 해보세요!



Reference

📖 ESL 3.5 (PCR / PLS)

📖 ISL 6.3 (PCR / PLS)

📖 AMSA 11.1-11.3 (PCA and SVD)

📖 Probabilistic Machine Learning (Kevin P. Murphy) 20.1 (Reconstruction Error)

📖 ESL 14.5.1 (Reconstruction Error)

🔗 <https://yuhuini.github.io/2018-12-01-relationship-between-ridge-regression-and-pca>

🔗 https://miro.medium.com/max/2000/1*ep6hbK8rpOsMCtAbIHLCJQ.gif

🔗 <https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c>

(요 글은 꼭 읽어보세요 📖🔗)

Principal Component Analysis

