Machine Learning 2

PCA & SVD

Dept. SW and Communication Engineering

Prof. Giseop Noh (kafa46@hongik.ac.kr)

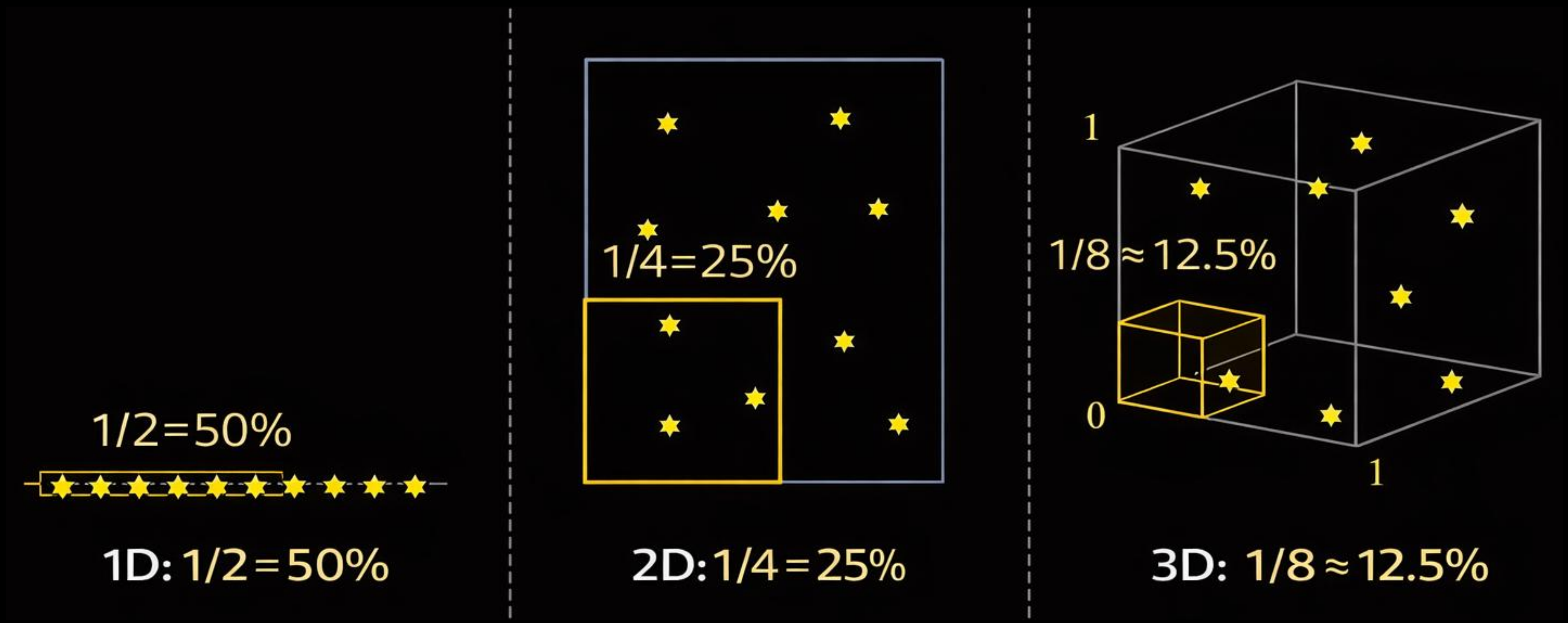
Lecture Goals

- **High-Dimensional Data Problem**
- PCA (Principal Component Analysis)
- SVD (Singular Value Decomposition)
- Analysis on PCs

High-Dimensional Data Problem

Curse of Dimensionality

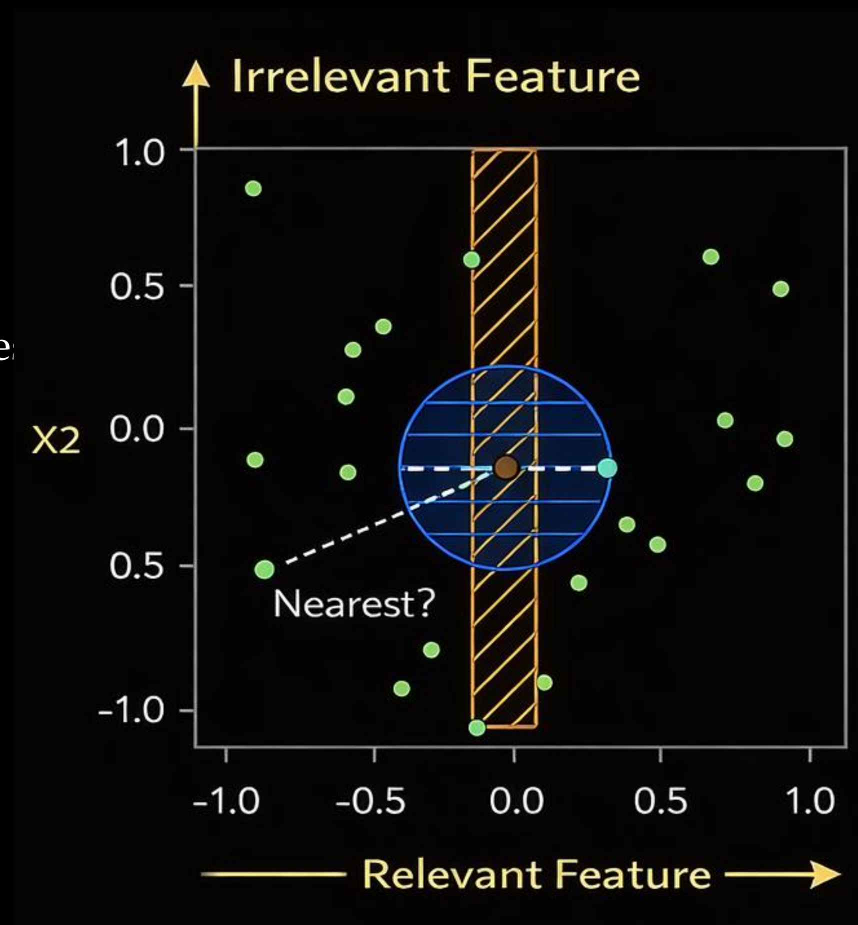
- The number of samples is limited.
- As dimensionality increases, the same local neighborhood occupies a much smaller proportion of the total volume.
- Required data grows exponentially with dimension.



Correlated Features & Distance

- **Consider a model that predicts using nearby samples (k-Nearest Neighbor).**
  - Adding irrelevant features significantly degrades performance.
  - High correlation among independent features distorts distance metrics.

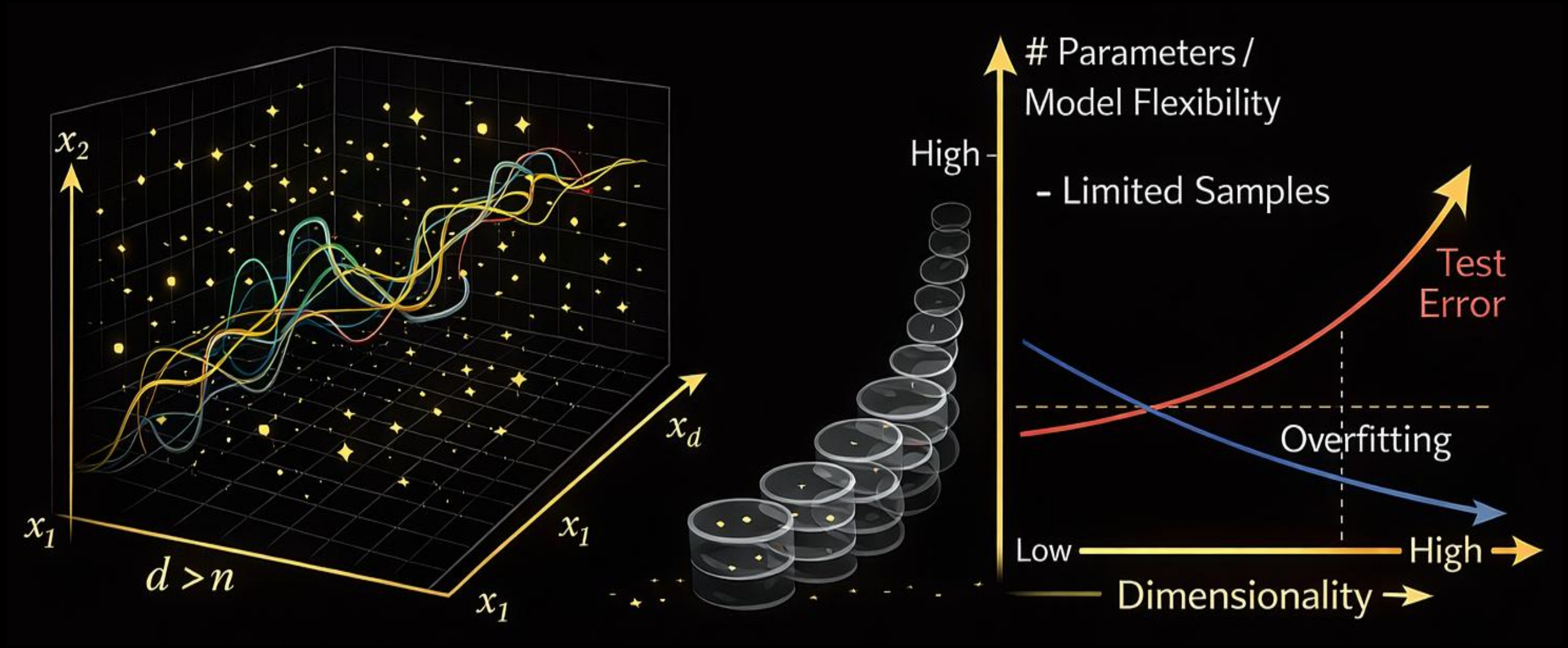
$$d(x_i, x_i) = \sqrt{\sum_{k=1}^{d} \left(x_{ik}, x_{jk}\right)^2}$$



- **Features unrelated to the target increase noise in distance computation.**
- **In high dimensions, the notion of “nearest” becomes unreliable.**

Overfitting Risk

- **Limited training samples cannot sufficiently cover the expanded space.**
- **The model starts fitting noise instead of true underlying patterns.**
- **Training error decreases, but generalization error increases.**

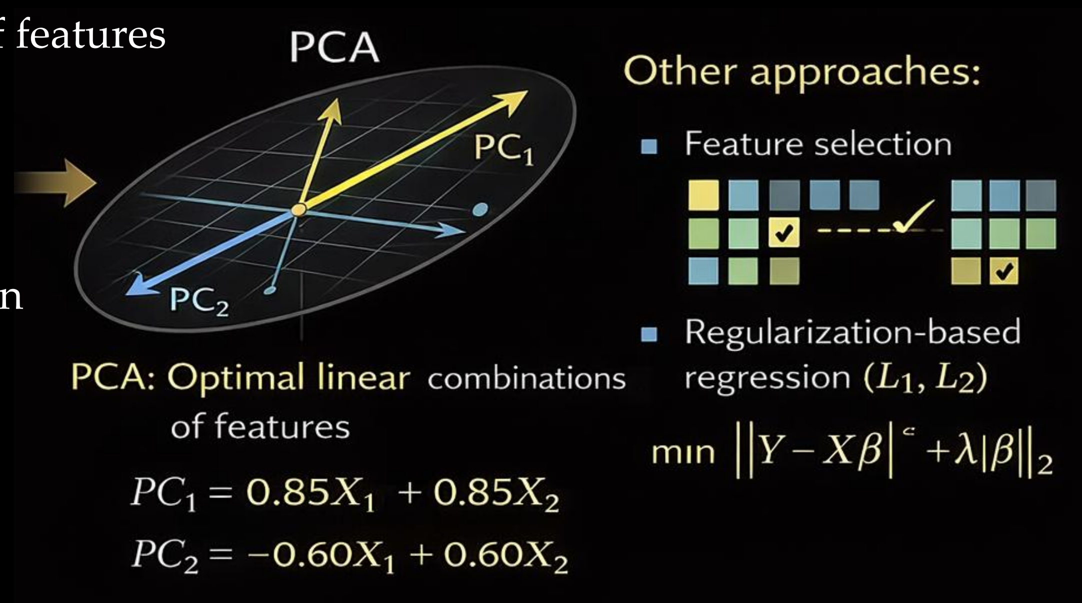


Dimensionality Reduction

- **Removing highly correlated variables may lead to information loss.**
  - If correlation = 0.8,

    the remaining 0.2 variation is discarded.
- **Need to reduces dimensionality while preserving maximum information.**
  - Principal Component Analysis (PCA)
    - Optimal linear combinations of features
  - Other approaches:
    - Feature selection
    - Regularization-based regression

      (L1, L2)
    - Drop out & Bagging

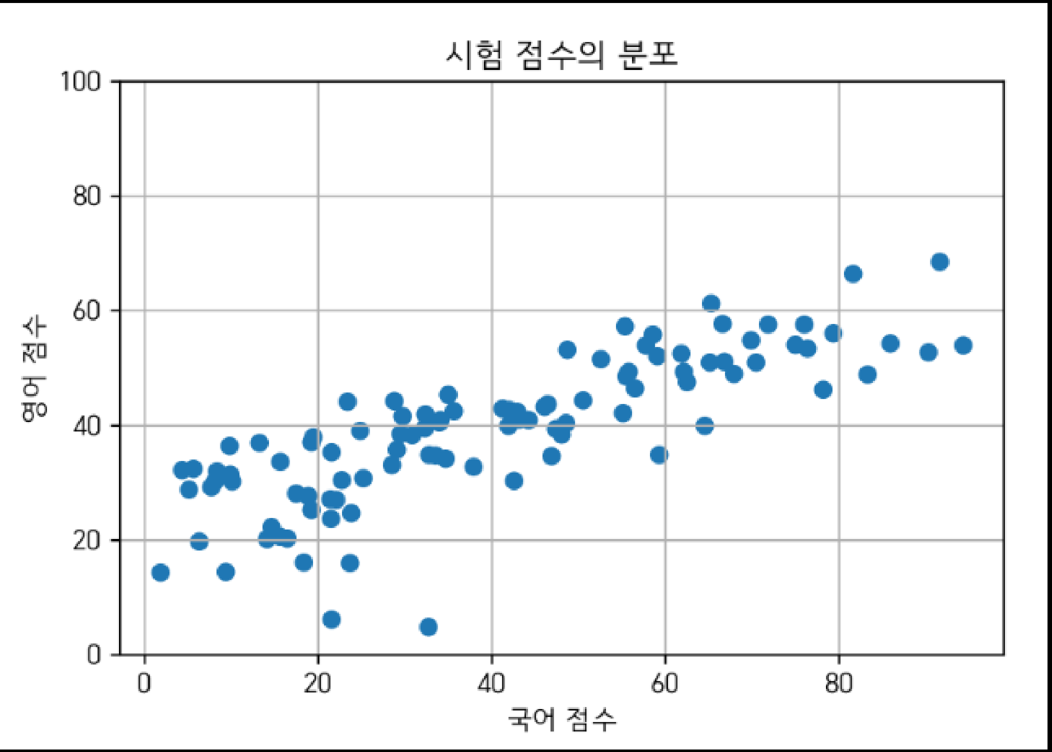


PCA
(Principal Component Analysis)

Data Representation

국어 점수	영어 점수
100	83
70	50
30	25
45	30
⋮	⋮
80	60

데이터가 너무 많다!
국어/영어 점수를 잘 표현하는 값으로 나타내려면?



데이터의 대푯값을 찾는 방법

- **먼저 평균을 취하는 방법이 쉽게 떠오른다.**
  - 그런데 국어는 어렵고, 영어는 조금 쉬운 시험이었다면?
  - 국어 성적의 가중치 60%, 영어 성적의 가중치를 40%로 준다면?

	국어점수	가중치	환산점수		영어점수	가중치	환산점수	대표점수
	100	60%	60		83	40%	33.2	93.2
	70	60%	42		50	40%	20	62
	30	60%	18		25	40%	10	28
	45	60%	27		30	40%	12	39
	:	:	:		:	:	:	:
	80	60%	48		60	40%	24	72
평균	65		39	평균	49.6		19.84	58.8

대푯값: 국어 가중치 평균 + 영어 가중치 평균 = 39 + 19.9 = 58.9

위 과정을 벡터로 표현하면?

➔ 다음 슬라이드로 넘어 가서 확인!

벡터 변환

국어 점수	영어 점수
100	83
70	50
30	25
45	30
⋮	⋮
80	60

$$S = \begin{bmatrix} \mathbf{100} & \mathbf{83} \\ \mathbf{70} & \mathbf{50} \\ \mathbf{30} & \mathbf{25} \\ \mathbf{45} & \mathbf{30} \\ \mathbf{80} & \mathbf{60} \end{bmatrix} \quad S^T = \begin{bmatrix} \mathbf{100} & \mathbf{70} & \mathbf{30} & \mathbf{45} & \mathbf{80} \\ \mathbf{83} & \mathbf{50} & \mathbf{25} & \mathbf{30} & \mathbf{60} \end{bmatrix}$$

가중치 대푯값: $S \cdot W^T$

$$\begin{bmatrix} \mathbf{100} & \mathbf{83} \\ \mathbf{70} & \mathbf{50} \\ \mathbf{30} & \mathbf{25} \\ \mathbf{45} & \mathbf{30} \\ \mathbf{80} & \mathbf{60} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{0.6} \\ \mathbf{0.4} \end{bmatrix} = [\mathbf{93.2} \quad \mathbf{62} \quad \mathbf{28} \quad \mathbf{39} \quad \mathbf{72}]$$

$$Vector\ Size:\ (5 \times 2) \cdot (2 \times 1) = (5 \times 1)$$

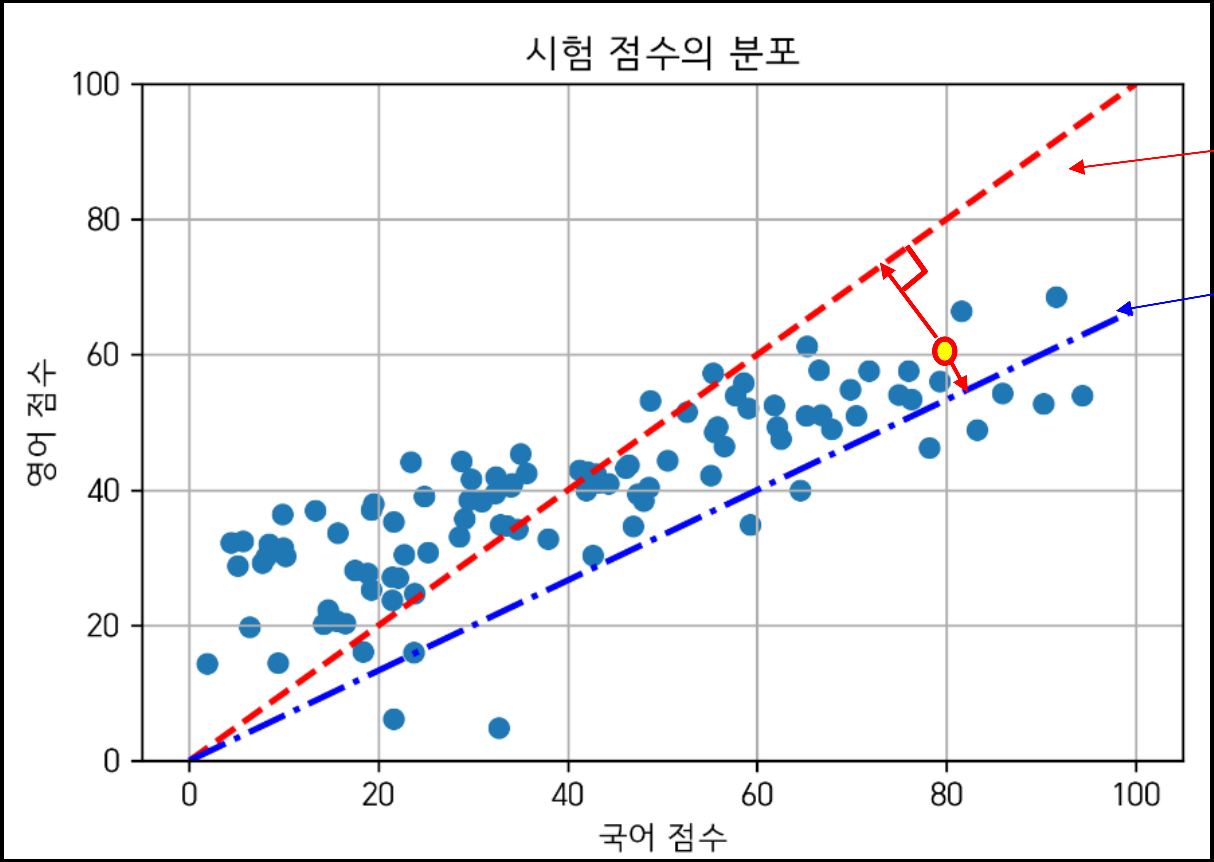
가중치

국어 가중치	영어 가중치
60%	40%

$$\boldsymbol{W} = [\mathbf{0.6} \quad \mathbf{0.4}] \quad \boldsymbol{W}^T = \begin{bmatrix} \mathbf{0.6} \\ \mathbf{0.4} \end{bmatrix}$$

Vector Projection

$$\begin{bmatrix} 100 & 83 \\ 70 & 50 \\ 30 & 25 \\ 45 & 30 \\ \boxed{80} & \boxed{60} \end{bmatrix} \cdot \begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix} = \begin{bmatrix} 93.2 & 62 & 28 & 39 & \boxed{72} \end{bmatrix}$$



벡터 프로젝션

빨간선: 50% : 50%

파란선: 60% : 40%

프로젝션 (Projection)
`정사영` 이라고도 함
특정 축에 직각으로 내린 점

(질문)
어떤 비율이 제일 좋을까???

Covariance Matrix

■ **두 개의 데이터(벡터)가 얼마나 닮았는가?**

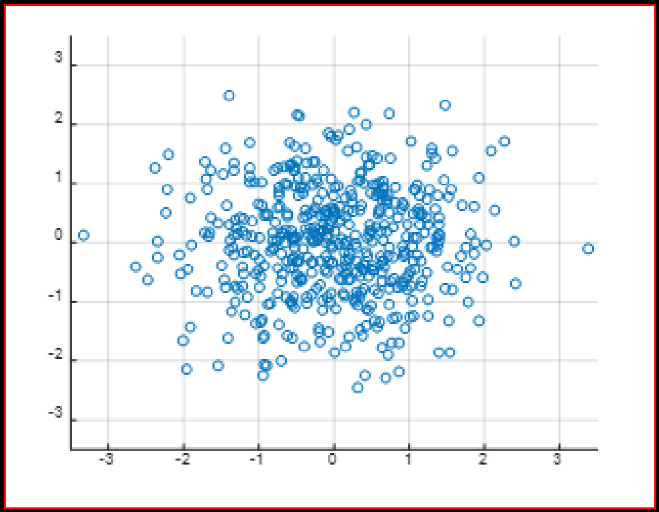
- 특징 쌍 (Feature Pairs) 들이 얼마나 비슷한가?
- 데이터들이 얼마나 같이 변하는가?

수학적 정의

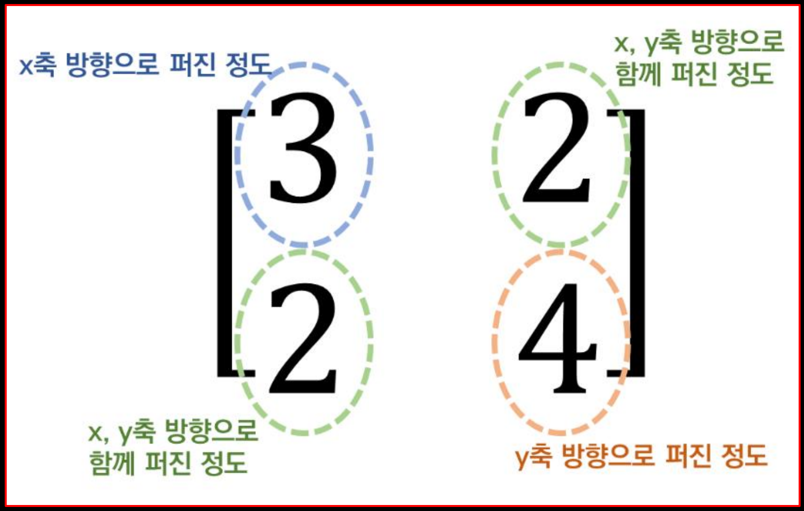
$$Cov(X) = \begin{bmatrix} Var(X_1) & Cov(X_1, X_2) \\ Cov(X_2, X_1) & Var(X_2) \end{bmatrix}$$

■ **공분산 행렬의 기하학적 해석**

- 행렬의 다른 표현 → 선형 변환



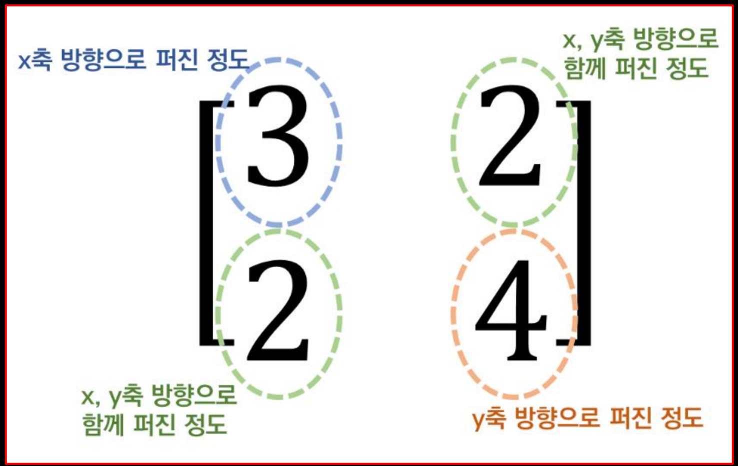
X

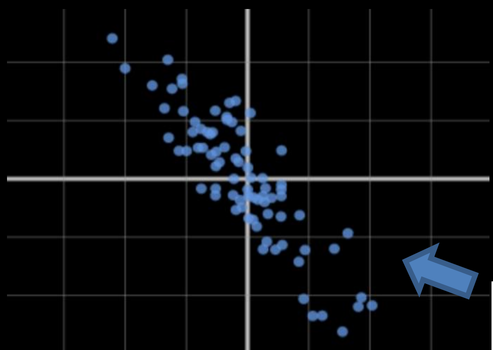


= ?

Covariance Matrix (cont.)

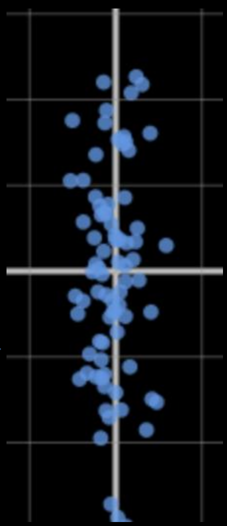
$$Cov(X) = \begin{bmatrix} Var(X_1) & Cov(X_1, X_2) \\ Cov(X_2, X_1) & Var(X_2) \end{bmatrix}$$

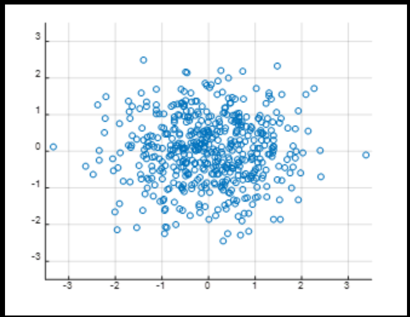


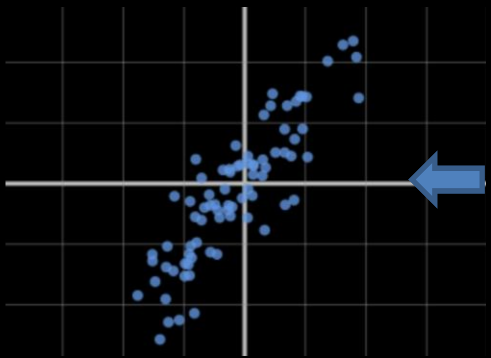


Matrix 2: $\begin{bmatrix} 3 & -2 \\ -2 & 4 \end{bmatrix}$ ×

× Matrix 4: $\begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix}$

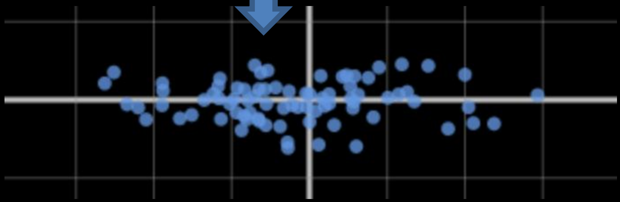






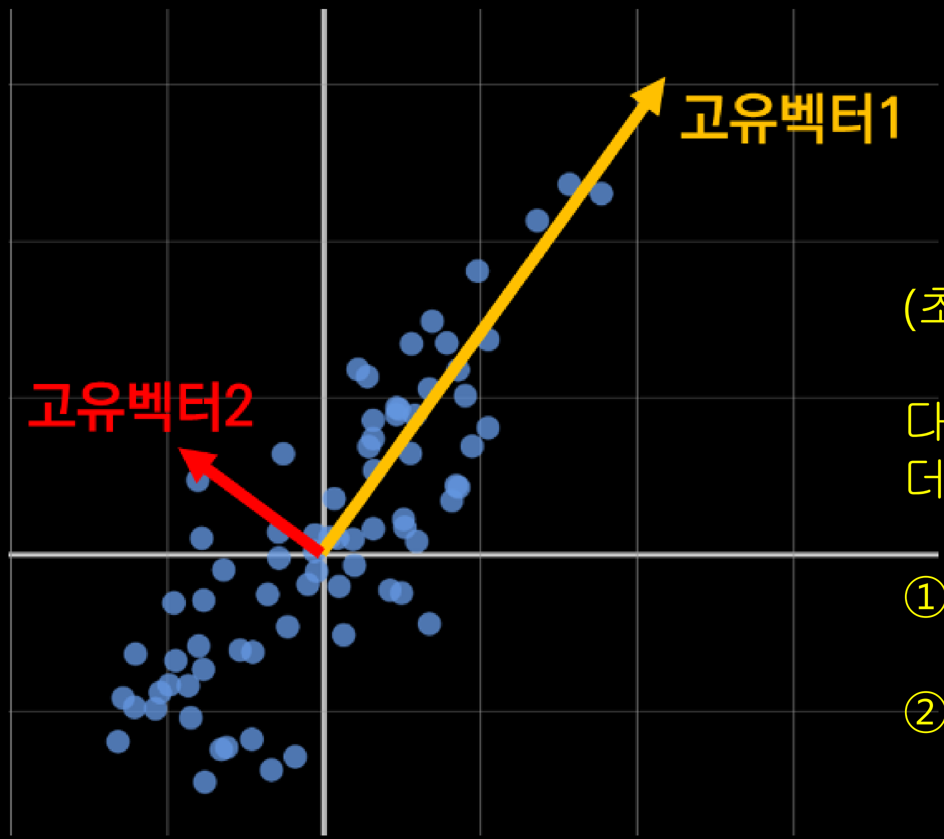
Matrix 1: $\begin{bmatrix} 3 & 2 \\ 2 & 4 \end{bmatrix}$ ×

× Matrix 3: $\begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix}$



Eigen Vector & Value

- **고유 벡터 (Eigen Vector): 행렬이 벡터에 작용하는 축의 방향**
- **고유 값 (Eigne Value): Eigen vector 방향으로 얼마만큼 크기로 늘어나는 정도**



(초딩 문제 .^^.)

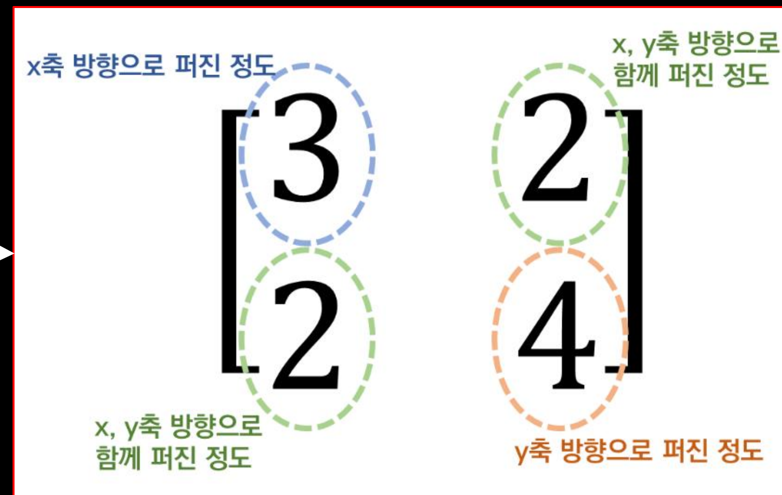
다음 중 어떤 벡터가 데이터를 더 잘 표현하나요?

① 고유벡터1

② 고유벡터2

Fundamental of PC (Principal Component)

- **Eigen Vector는 행렬이 벡터에 작용하는 주축 (Principal Axis)의 방향을 나타낸다.**



공분산 행렬도
역시 행렬!

- **그렇다면 공분산 행렬의 Eigen Vector의 의미는?**
  - 공분산 행렬을 생성한 데이터가 어떤 방향으로 분산되었는지 알 수 있다!
  - Eigen Value는 얼마나 벡터가 늘어나는지를 알려준다.
  - 그러므로 Eigen Value가 큰 순서대로 Eigen Vector를 정렬하고
  - 가장 큰 Eigen value 값을 갖는 Eigne Vector를 순서대로 선택하면 된다.
  - 이렇게 선택되는 벡터들이 주성분(Principal Component)이다.

그렇다면 공분산을 구해야 되나요?

- **교수님~~**
- **그렇다면 공분산을 구해야 되나요?**
  - 네, 맞습니다.

- **교수님~ 저는 수학이 꽝입니다. ㅠㅠ**

Don’t worry, be happy!

- 걱정 마세요 ^^.

- 여러분에게는 교수님과 컴퓨터가 있습니다.

교수님은 이론을 가르쳐 줄 것입니다 ^^.

https://www.hani.co.kr/arti/society/society_general/761295.html

컴퓨터는 실제 계산을 해 줄 것입니다 ^^.

https://www.ajunews.com/view/20210526232711295

Dataset Representation

■ 두 개 이상의 변량 값들 간의 변화량 ➔ 공분산 행렬

- N 개의 데이터로 부터 k 개의 특징(feature)를 모았다고 가정합니다.

$$D = \{a,\ b,\ c\}$$

$$D = \{\ d_i\ \}_{i=1}^{N}$$

$$d_i = (x_i\ ,\ y_i)$$

$$D = \{\ (x_i,\ y_i)\ \}_{i=1}^{N}$$

y (연봉)	x_1 (나이)	x_2 (직종)	x_3 (성별)	x_4 (학력)	…	x_k
y_1=5천만	30	2	1	3		
y_2=7천만	25	1	0	4		
⋮	⋮	⋮	⋮	⋮		
y_N=8천만	27	3	1	4		

$$D = \begin{pmatrix} d_1 & d_2 & d_3 & \cdots & d_k \end{pmatrix} \in \mathbb{R}^{N \times k}$$

Data Centering

- **데이터 벡터가 평균 값 0을 갖도록 조정**
  - 데이터 값에서 평균을 빼 주면 됨

$$D = \begin{pmatrix} d_1 & d_2 & d_3 & \cdots & d_k \end{pmatrix} \Rightarrow X = D - mean(D)$$

예제: 5명으로부터 2개 특징(feature)을 모았다고 가정 ➔ 5명의 키, 몸무게

$$D = \begin{array}{cc} d_1 & d_2 \\ \end{array}$$

$$D = \begin{bmatrix} 170 & 70 \\ 150 & 45 \\ 160 & 55 \\ 180 & 60 \\ 170 & 80 \end{bmatrix} \Rightarrow$$

$$X = D - mean(D)$$

$$\begin{bmatrix} 170 & 70 \\ 150 & 45 \\ 160 & 55 \\ 180 & 60 \\ 172 & 80 \end{bmatrix} - \begin{bmatrix} 166 & 62 \\ 166 & 62 \\ 166 & 62 \\ 166 & 62 \\ 166 & 62 \end{bmatrix} = \begin{bmatrix} 4 & 8 \\ -16 & -17 \\ -6 & -7 \\ 14 & -2 \\ 6 & 18 \end{bmatrix}$$

X를 이용한 공분산 행렬 구하기 1

$$X^T X = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ \vdots \\ X_k \end{pmatrix} \begin{pmatrix} X_1 & X_2 & X_3 & \cdots & X_k \end{pmatrix}$$

X^T: 'X 트랜스포즈' 라고 읽습니다. X의 전치행렬 이라고 부르기도 합니다.

X의 모든 구성 원소 $x_{i,j}$에서 인덱스 값을 서로 바꿔준 행렬입니다. 즉, $X^T = x_{j,i}$

위 식을 전개하면 어떻게 될까요?

다음 슬라이드에서 확인해 보겠습니다 ^^.

X를 이용한 공분산 행렬 구하기 2

$$X^T X = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ \vdots \\ X_k \end{pmatrix} \begin{pmatrix} X_1 & X_2 & X_3 & \cdots & X_k \end{pmatrix}$$

$$= \begin{bmatrix} dot(X_1, X_1) & dot(X_1, X_2) & \cdots & dot(X_1, X_k) \\ dot(X_2, X_1) & dot(X_2, X_2) & \cdots & dot(X_2, X_k) \\ \vdots & \vdots & \ddots & \vdots \\ dot(X_N, X_1) & dot(X_N, X_2) & \cdots & dot(X_N, X_k) \end{bmatrix}$$

$X^T X$ 행렬의 i 행, j 열 성분 $(X^T X)_{i,j}$ 의미는 무엇일까요?

➔ k 개 feature 중에서 i 번째와 j 번째 feature가 얼마나 닮았는지 표현한 수치 입니다.

예제를 통해 살펴보기

앞에서 예제로 사용한 5명의 2개 features (키, 몸무게)를 사용해 보겠습니다 ^^.

$$D = \begin{array}{cc} d_1 & d_2 \end{array}$$

$$D = \begin{bmatrix} 170 & 70 \\ 150 & 45 \\ 160 & 55 \\ 180 & 60 \\ 170 & 80 \end{bmatrix}$$

$$X = \begin{bmatrix} 170 & 70 \\ 150 & 45 \\ 160 & 55 \\ 180 & 60 \\ 172 & 80 \end{bmatrix} - \begin{bmatrix} 166 & 62 \\ 166 & 62 \\ 166 & 62 \\ 166 & 62 \\ 166 & 62 \end{bmatrix} = \begin{bmatrix} 4 & 8 \\ -16 & -17 \\ -6 & -7 \\ 14 & -2 \\ 6 & 18 \end{bmatrix}$$

$$X^T X = \begin{bmatrix} 4 & -16 & -6 & 14 & 6 \\ 8 & -17 & -7 & -2 & 18 \end{bmatrix} \begin{bmatrix} 4 & 8 \\ -16 & -17 \\ -6 & -7 \\ 14 & -2 \\ 6 & 18 \end{bmatrix} = \boxed{\begin{bmatrix} 540 & 426 \\ 426 & 730 \end{bmatrix}}$$

공분산 행렬

그런데... 데이터를 모을수록(많아지면) 값이 점점점…점점점 커집니다.
이런 현상을 방지하기 위해 데이터의 개수 n 으로 나누어 줍니다.

공분산 행렬 - 최종판

$$Cov\ Mat(\Sigma) = \frac{X^T X}{n} = \frac{1}{n}\begin{bmatrix} dot(X_1, X_1) & dot(X_1, X_2) & \cdots & dot(X_1, X_k) \\ dot(X_2, X_1) & dot(X_2, X_2) & \cdots & dot(X_2, X_k) \\ \vdots & \vdots & \ddots & \vdots \\ dot(X_N, X_1) & dot(X_N, X_2) & \cdots & dot(X_N, X_k) \end{bmatrix}$$

우리의 5명의 키, 몸무게 예를 들면...

$$\Sigma = \frac{1}{5} X^T X = \frac{1}{5}\begin{bmatrix} 540 & 426 \\ 426 & 730 \end{bmatrix} = \begin{bmatrix} 108 & 85.2 \\ 85.2 & 146 \end{bmatrix}$$

공분산 행렬까지 구했습니다.
여러분들이 손으로 계산할 일은 없습니다.
코딩 한 줄로 처리할 수 있는 다양한
라이브러리들이 있습니다.

예) Numpy
import numpy as np
np.cov( )

예) Pandas DataFrame
import pandas as pd
pd.DataFrame.cov( )

Next Step?

- 교수님~~
- 공분산 행렬 라이브러리를 이용해 쉽게 계산했어요...
- 다음에는 고유 벡터 (Eigen Vector), 고유 값(Eigen Value) 구해야 되나요?
  - 네, 맞습니다 .^^.

- 교수님~ 저는 수학이 꽝이라니까요!!

걱정 마세요...

■ **걱정 마세요 ^^.**

■ **여러분에게는 교수님과 컴퓨터가 있습니다.**

교수님은 이론을 가르쳐 줄 것입니다 ^^.

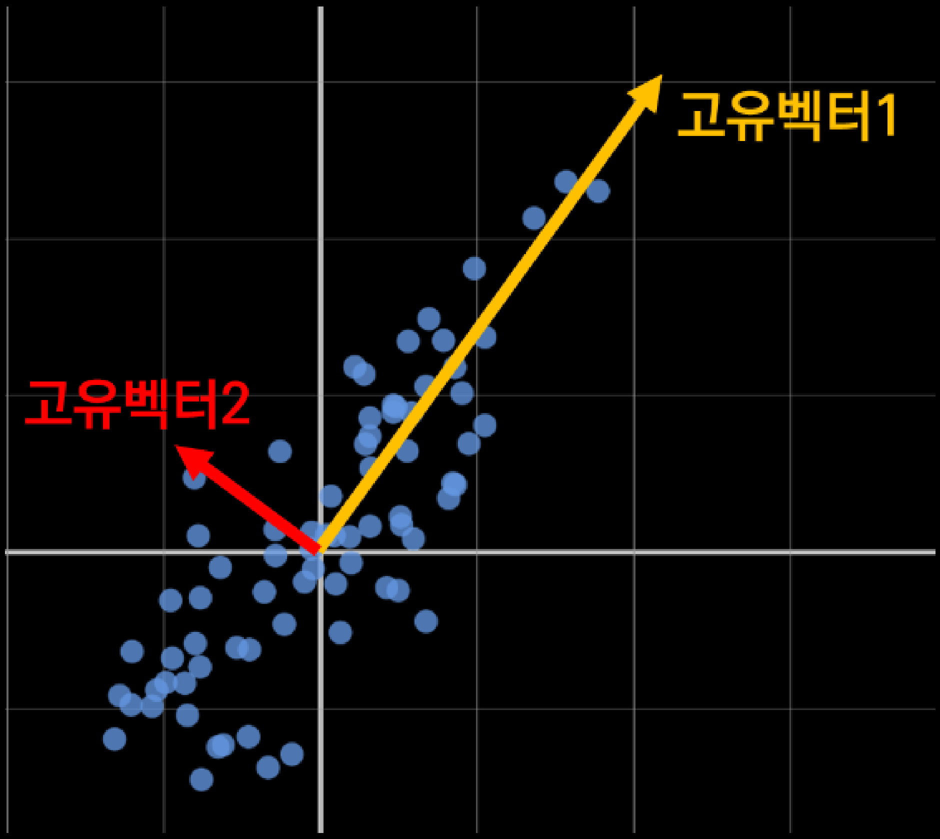
https://www.hani.co.kr/arti/society/society_general/761295.html

컴퓨터는 실제 계산을 해 줄 것입니다 ^^.

https://www.ajunews.com/view/20210526232711295

데이터를 잘 설명하는 어떤 값을 찾고 싶은 것!

- **고유 벡터 (Eigen Vector): 행렬이 벡터에 작용하는 축의 방향**
- **고유 값 (Eigne Value): Eigen vector 방향으로 얼마만큼 크기로 늘어나는 정도(값)**



차원을 줄이면서 정보 손실을 최소화 하는 것

더 적은 데이터로 많은 데이터를 잘 설명할 수 있는 새로운 축을 찾는 것

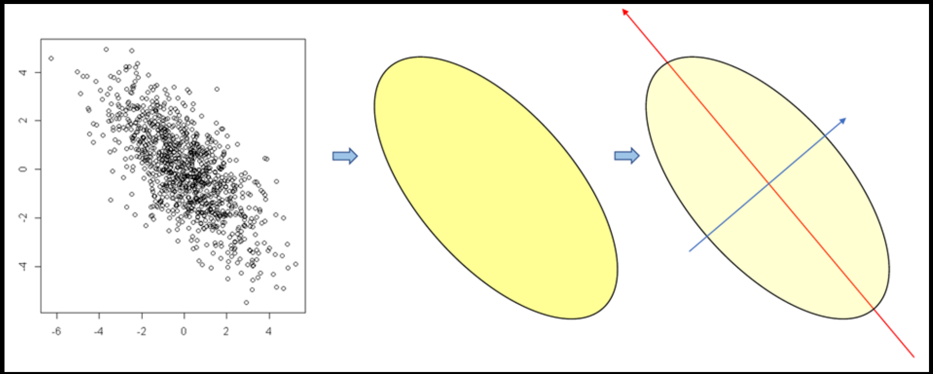
이렇게 찾아낸 축을 주성분(Principal Componet)라고 부릅니다.

주성분(Principal Component, PC)

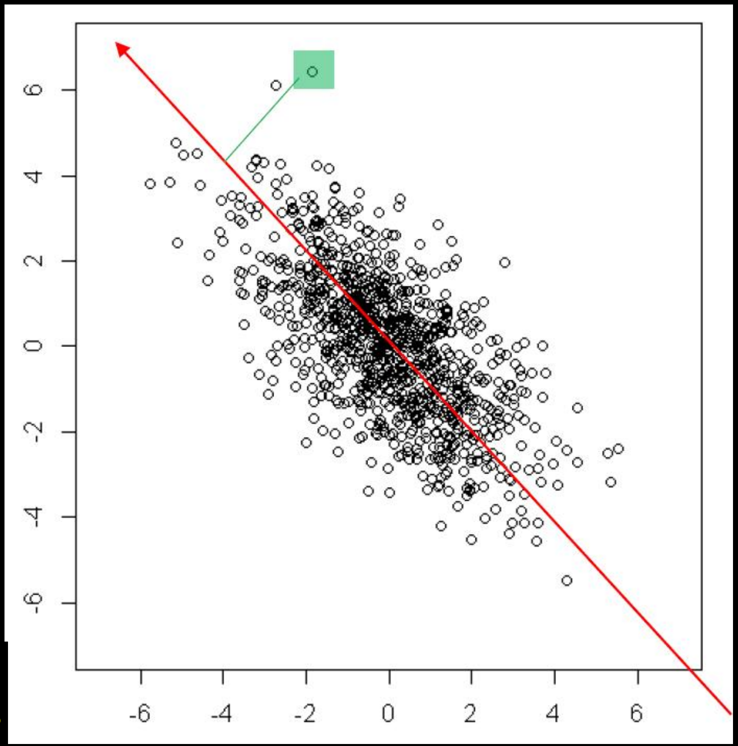
■ 주성분(Principal Component, PC)

예시) 2차원 데이터 (3차원 이상에서는 아주 복잡하겠죠?)

- 공분산이 데이터의 형태를 변형시키는 방향축, 그리고 그 축에 직교하는 축을 찾는 과정
- 공분산이 나타내는 타원의 장축과 단축을 찾는 것과 같은 말입니다.



■ Principal component, PC, PC score



➔ 모두 같은 말입니다.

- 물리적 의미: 새로 찾아낸 축에서의 좌표값
- 교수님~ 그래서 어떻다는 거죠?
  공분산 행렬을 이용해서 PC를 찾는다는 말인가요?

Determinant (행렬식)

Determinant (행렬식)

■ 교수님~ Determinant가 뭔가요?

- Determinant는 `행렬식` 이라고 부르기도 합니다. 사전적 의미는 `결정 요인`이라는 뜻
- 일반적으로 영어 `디터미넌트`라고 더 자주 부릅니다.
- 어떤 행렬이 있을 때 주어진 계산식에 대입하여 구한 숫자를 말합니다.
- 수식으로 어떤 행렬 A에 대하여 $|A|$ 또는 det(A) 라고 표기합니다.

존재를 결정하는 값으로 활용하기도 합니다.
나중에 나옵니다.

■ Determinant 구하는 공식이 어떻게 되나요?

2x2 행렬인 경우

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \Rightarrow |A| = a_{11}a_{22} - a_{12}a_{21}$$

3x3 행렬인 경우

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$\Rightarrow |A| = +a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12}\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13}\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

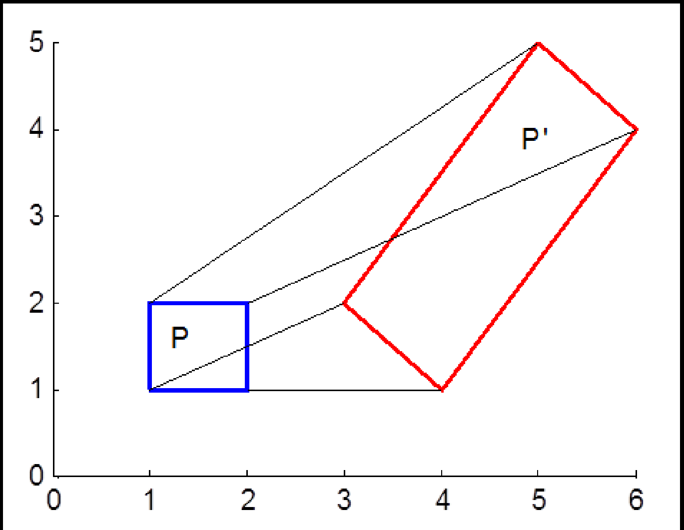
$$= a_{11}(a_{22}a_{33} - a_{23}a_{32}) - a_{12}(a_{21}a_{33} - a_{23}a_{31}) + a_{13}(a_{21}a_{32} - a_{22}a_{31})$$

행렬이 더 클경우?
➔ 일반화 시키면 됨

Determinant의 기하학적 의미

■ 그래서 Determinant가 어쨌다는 건가요?

- 기하학적 의미를 알아야 합니다.
- $|A|$ 는 행렬 A가 선형변환 할 때 스케일(비율) 성분을 뜻합니다.
- 어떤 행렬 $P = \begin{bmatrix} 1 & 1 & 2 & 2 \\ 1 & 2 & 1 & 2 \end{bmatrix}$ 가 있다고 가정해 봅니다.
- 행렬 P에 행렬 $A = \begin{bmatrix} 1 & 2 \\ -1 & 3 \end{bmatrix}$ 을 곱해서 선형변환을 한다면... 즉, $P' = AP$ 인 경우
  - $P` = \begin{bmatrix} 1 & 2 \\ -1 & 3 \end{bmatrix} \begin{bmatrix} 1 & 1 & 2 & 2 \\ 1 & 2 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 3 & 5 & 4 & 6 \\ 2 & 5 & 1 & 4 \end{bmatrix}$
  - $\det(A) = |A| = 1 \times 3 - (-1) \times 2 = 3 + 2 = 5$
  - $\text{area}(P') = |A|\ \text{area}(P) = 5 \times 1 = 5$



교수님~ det(A) 값이 0 이면 어찌 되나요?
면적이 0이 되므로, P' 은 면적이 없는 선분이 됩니다 ^^

역행렬(Inverse)

- **교수님~ 역행렬 (Inverse)가 뭔가요?**
  - 어떤 행렬 A와 곱했을때 단위행렬 E가 나오는 행렬을 A의 역행렬이라고 합니다.
  - 영어로는 `A inverse (에이 인버스)` 라고 읽습니다.
  - 수식으로 어떤 행렬 A의 역행렬은 A^{-1} 이라고 씁니다.
- **우쒸! 교슈님, 단위행렬 E는 또 뭔가요?**
  - n × n 크기를 갖는 행렬 A가 있을 때 대각(diagonal) 원소는 모두 1이고
  - 나머지 모든 원소 값이 0 인 행렬입니다. ^^
  - 어떤 행렬에 단위 행렬을 곱하면 자기 자신이 됩니다. $A^{-1}A = AA^{-1} = A^{-1}A$
  - 수학적으로는 $1_{n \times n}$ 이라고 씁니다.

$$1_{n\times n} = \mathrm{diag}(\underbrace{1,1,1,\ldots,1}_{n}) = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}_{n\times n}$$

역행렬과 Determinant

- **먼저, 역행렬이 존재할 경우 다음과 같은 아름다운 계산이 가능합니다.**

  $C = AB$

  $A^{-1}C = A^{-1}AB$ ← 양변에 A^{-1} 행렬을 곱해주었습니다.

  따라서 우리는 행렬 B를 구할 수 있습니다. ➔ $A^{-1}C = B$

- **역행렬과 Determinant의 관계**
  - 어떤 행렬식 A 의 determinant, $|A| = 0$ 인 경우, 역행렬이 존재하지 않습니다.

- **이 성질을 이용하면,**
  - 두 행렬 A, B를 곱했을때 결과가 0인 경우, 즉 $AB = 0$
  - 어떤 행렬이 0 인지 찾아낼 수 있습니다.
  - 만약 A의 역행렬 A^{-1} 이 존재한다면, 행렬 $B = 0$ 이 됩니다.

  $AB = 0$

  $A^{-1}AB = A^{-1}0$

  $B = 0$

존재의 의미를 찾자! – Determinant 활용

AB = 0

- 위 식에서 A 아니면 B는 0이 될 겁니다.

- 행렬 A의 역행렬이 존재하면, 행렬 B = 0 입니다.
  - 행렬 A 역행렬이 존재하려면 $|A| \neq 0$

- 행렬 B의 역행렬이 존재하면, 행렬 A = 0 입니다.
  - 행렬 B 역행렬이 존재하려면 $|B| \neq 0$

Determinant (행렬식)는 존재를 결정하는 값으로 활용합니다.

■ 반대로 표현할 수도 있겠죠?

일단 여기까지 알아두세요^^
비밀의 문이 서서히 열리게 됩니다.

AB = 0

만약 $|A| = 0$ 이면, A = 0

왜냐하면, $|A| = 0$, ➔ A의 역행렬 존재 X ➔ B 역행렬 존재 ➔ 행렬 B 소거됨 ➔ A = 0

만약 $|B| = 0$ 이면, B = 0 (위와 마찬가지 이유)

두번째 언덕: Eigen vector, eigen value

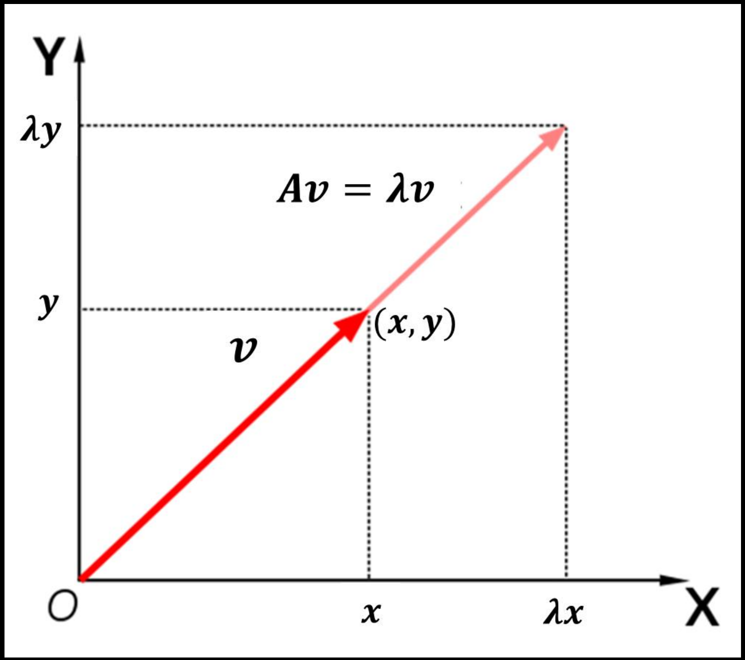
Eigen vector, eigen value

- **어떤 정방행렬(정사각행렬) A가 있습니다.**
- **그리고 어떤 벡터 $\boldsymbol{v}$ 가 있습니다.**

  아래 조건을 만족할 때,

  $$\boldsymbol{Av = \lambda v}$$

  - $\boldsymbol{v}$ 를 eigen vector (고유 벡터),
  - 상수 $\boldsymbol{\lambda}$ 를 eigen value (고유 값) 이라고 합니다.



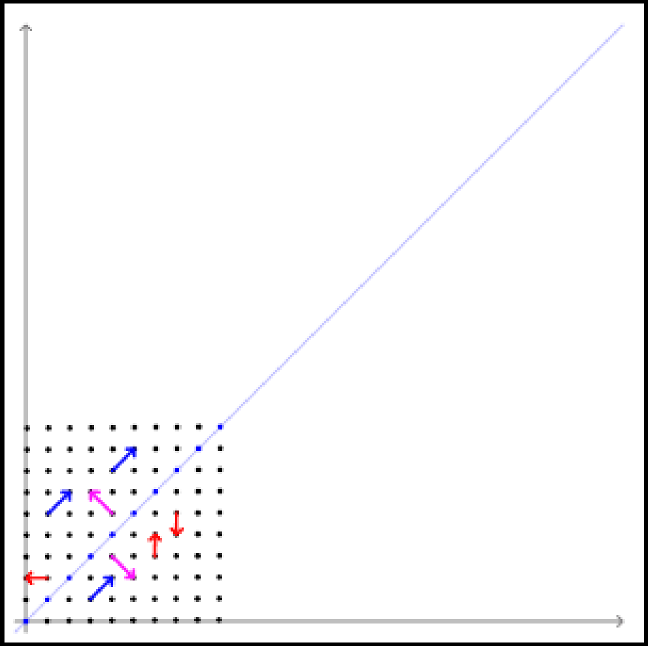
- **물리적 의미**
  - 행렬 A는 다음과 같이
  - 벡터 $\boldsymbol{v}$ 를 선형 변환합니다.

물리적 의미

- **Eigen vector, eigen value**

$$Av = \lambda v$$

- **어떤 점에 대해 행렬 A로 선형변환 할 때**
  - 방향은 바뀌지 않고, 크기만 바뀌는 벡터를 eigen vector (고유 벡터) $\boldsymbol{v}$ 라 합니다.
  - 방향만 바뀔 때 변화되는 정도를 eigen value (고유 값) λ 이라고 합니다.



이미지 출처:
https://upload.wikimedia.org/wikipedia/commons/0/06/Eigenvectors.gif

Eigen vector 구하기 1

$$Av = \lambda v$$

$$A[\ \mathbf{v}_1\ \mathbf{v}_2 \cdots \mathbf{v}_n\ ] = [\ \lambda_1\mathbf{v}_1\ \lambda_2\mathbf{v}_2 \cdots \lambda_n\mathbf{v}_n\ ] = [\ \mathbf{v}_1\ \mathbf{v}_2 \cdots \mathbf{v}_n\ ]\begin{bmatrix}\lambda_1 & & & 0\\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n\end{bmatrix}$$

$Av - \lambda v = 0$ ← 양변을 λv 뺐습니다.

$(A - \lambda I)v = 0$ ← v 로 묶어주고, 숫자 값 λ에 행렬계산을 위해 항등 행렬 I 을 곱해 줌

2×2 행렬 $A = \begin{bmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\end{bmatrix}$, $I = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$, $v = \begin{bmatrix}x_1\\ x_2\end{bmatrix}$, 예로 들어 위 식을 전개

$$\begin{bmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} - \lambda\begin{bmatrix}x_1\\ x_2\end{bmatrix} = 0$$

$$\begin{bmatrix}a_{11}x_1 + a_{12}x_2\\ a_{21}x_1 + a_{22}x_2\end{bmatrix} - \begin{bmatrix}\lambda x_1\\ \lambda x_2\end{bmatrix} = 0$$

$$\begin{bmatrix}a_{11}x_1 + a_{12}x_2 - \lambda x_1\\ a_{21}x_1 + a_{22}x_2 - \lambda x_2\end{bmatrix} = 0$$

$$\begin{bmatrix}(a_{11} - \lambda)x_1 + a_{12}x_2\\ a_{21}x_1 + (a_{22} - \lambda)x_2\end{bmatrix} = 0$$

우리의 목표는 $v = \begin{bmatrix}x_1\\ x_2\end{bmatrix}$ 를 찾는 것!

v 가 존재하려면 $(A - \lambda I)$의 역행렬이 존재하지 않아야 합니다.

역행렬이 존재하지 않으려면 $\det(A - \lambda I) = 0$

Eigen vector 구하기 2

$$\boldsymbol{Av = \lambda v}$$

$$\boldsymbol{(A - \lambda I)v = 0}$$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a_{11}} & \boldsymbol{a_{12}} \\ \boldsymbol{a_{21}} & \boldsymbol{a_{22}} \end{bmatrix}, \ \boldsymbol{I} = \begin{bmatrix} \boldsymbol{1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix}, \ \boldsymbol{v} = \begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \lambda \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{bmatrix} - \begin{bmatrix} \lambda x_1 \\ \lambda x_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} a_{11}x_1 + a_{12}x_2 - \lambda x_1 \\ a_{21}x_1 + a_{22}x_2 - \lambda x_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} (a_{11} - \lambda)x_1 + a_{12}x_2 \\ a_{21}x_1 + (a_{22} - \lambda)x_2 \end{bmatrix} = 0$$

$$\boxed{\begin{bmatrix} (a_{11} - \lambda) & a_{12} \\ a_{21} & (a_{11} - \lambda) \end{bmatrix}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

우리의 목표는 $\boldsymbol{v} = \begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \end{bmatrix}$ 를 찾는 것!

$\boldsymbol{v}$ 가 존재하려면 $\boldsymbol{(A - \lambda I)}$의 역행렬이 존재하지 않아야 합니다.

역행렬이 존재하지 않으려면

$$\det\left(\begin{bmatrix} (a_{11} - \lambda) & a_{12} \\ a_{21} & (a_{22} - \lambda) \end{bmatrix}\right) = 0$$

따라서,

$$(a_{11} - \lambda)(a_{22} - \lambda) - a_{12}\, a_{21} = 0$$

위 식을 만족하는 λ를 찾으면 됩니다.
(고등학교때 배운 2차 방적식 푸는 문제와 동일해 집니다.)

Eigen vector 구하기 – 실습 예제

정방행렬(정사각행렬) A의 eigen vector를 구해 볼까요?

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{4} & \mathbf{2} \\ \mathbf{3} & \mathbf{5} \end{bmatrix}$$

$\det\left(\begin{bmatrix} (a_{11}-\lambda) & a_{12} \\ a_{21} & (a_{22}-\lambda) \end{bmatrix}\right) = 0$, 따라서, $(a_{11}-\lambda)(a_{22}-\lambda) - a_{12}\ a_{21} = 0$

위 수식에 그대로 대입해 보겠습니다.

$$(a_{11}-\lambda)(a_{22}-\lambda) - a_{12}\ a_{21} = 0$$

$$(4-\lambda)(5-\lambda) - 2 \times 3 = 0$$

$$20 - 4\lambda - 5\lambda + \lambda^2 - 6 = 0$$

$$\lambda^2 - 9\lambda + 14 = 0$$

$$(\lambda - 7)(\lambda - 2) = 0$$

이차방정식의 답: $\lambda = 7$ 또는 $\lambda = 2$

$$\boldsymbol{Av} = \boldsymbol{\lambda v}$$

$\boxed{(\boldsymbol{A} - \boldsymbol{\lambda I})\boldsymbol{v} = \mathbf{0}}$ 이므로 그대로 대입하면

$$\begin{bmatrix} \mathbf{4-7} & \mathbf{2} \\ \mathbf{3} & \mathbf{5-7} \end{bmatrix} \begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \end{bmatrix} = \mathbf{0} \Rightarrow \boldsymbol{v} = \begin{bmatrix} \mathbf{2} \\ \mathbf{3} \end{bmatrix}$$

또는

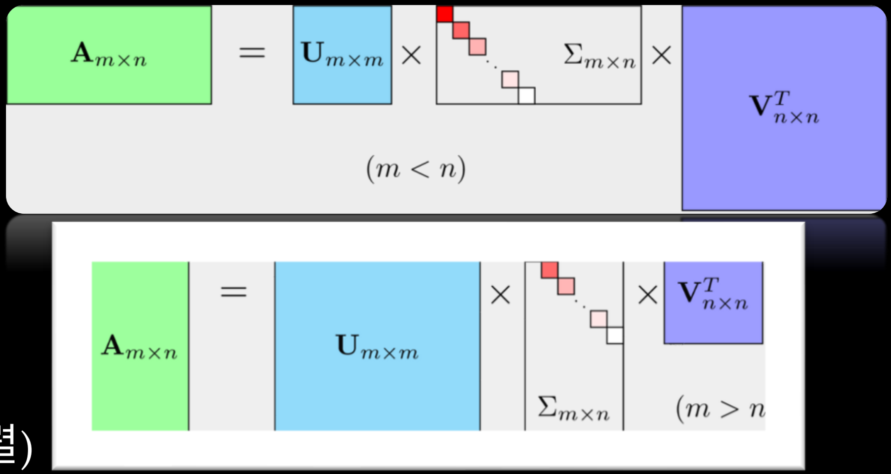
$$\begin{bmatrix} \mathbf{4-7} & \mathbf{2} \\ \mathbf{3} & \mathbf{5-7} \end{bmatrix} \begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \end{bmatrix} = \mathbf{0} \Rightarrow \boldsymbol{v} = \begin{bmatrix} \mathbf{-1} \\ \mathbf{1} \end{bmatrix}$$

세번째 언덕: SVD
(Singular Value Decomposition)

SVD: Singular Value Decomposition

- 하나의 행렬을 여러 개로 분해하는 방법 (정확히 3개의 행렬로 분해합니다.)
- 어떤 행렬 A 가 $m \times n$ 의 크기를 가진 직사각형 행렬인 경우, 다음과 같이 분해합니다.

$$A = U\Sigma V^T$$



- 여기서 각 행렬의 크기는 다음과 같습니다.
  - A: $m \times n$ U: $m \times m$ Σ: $m \times n$
- 여기서 각 행렬의 특징은 다음과 같습니다.
  - 행렬 U 와 V: Orthogonal matrix (직교 행렬)
  - 행렬 Σ: Diagonal Matrix (대각 행렬)

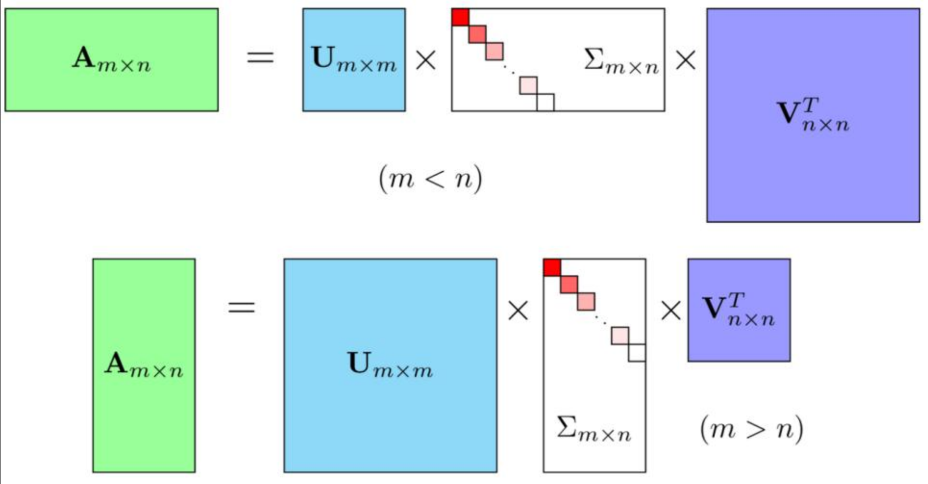
교수님~ Orthogonal matrix? 직교 행렬이 도데체 뭔가요?
Diagonal Matrix (대각 행렬)도 모르겠어요 ㅠㅠ
➔ 다음 슬라이드에서 설명하겠습니다.

SVD: Singular Value Decomposition

SVD: Singular Value Decomposition

- 하나의 행렬을 여러 개로 분해하는 방법 (정확히 3개의 행렬로 분해합니다.)
- 어떤 행렬 A 가 $m \times n$ 의 크기를 가진 직사각형 행렬인 경우, 다음과 같이 분해합니다.

$$A = U\Sigma V^T$$



- 여기서 각 행렬의 크기는 다음과 같습니다.
  - A: $m \times n$ U: $m \times m$ Σ: $m \times n$ V: $n \times n$
- 여기서 각 행렬의 특징은 다음과 같습니다.
  - $V^T V = I_p$
  - $U^T U = I_p$
  - Σ: Diagonal Matrix
  - V 의 열벡터(column vectors) ➔ $X^T X$ 의 eigen vectors
  - Σ의 대각 원소(diagonal entries) ➔ $X^T X$ 의 eigen values

Orthogonal Matrix (직교 행렬)

■ **Orthogonal Matrix (직교 행렬)**

- 모든 column vector가 자기 자신을 제외한 나머지 모든 column vector들과 Orthogonal (직교)하는 unit vector (단위 벡터, 길이가 1인 벡터)

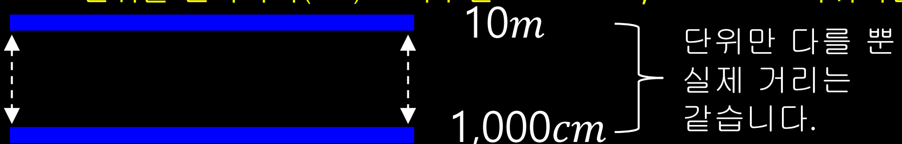
헐… 교수님~
Unit vector (단위 벡터)는 또 뭐래요? ㅠㅠ

교수님~
Orthogonal (직교) 라구요? 그거 먹는 건가요? ㅠㅠㅁ

Unit vector (단위 벡터)

■ Unit vector (단위 벡터)

- 벡터의 단위를 표현합니다.
- 우리가 자주 사용하는 거리 단위는 뭐가 있나요? 대표적인 것이 미터 (m) 입니다.
  - 10m (십 미터) 의미는? '미터 (m)' 단위가 10개 있다는 뜻이겠죠?
  - 여기서 단위 미터 (m) 는 단위만 표현할 뿐 실제 거리에 영향은 없는거죠
    - 단위를 센치미터(cm)로 바꾸면 10m ➔ 1,000cm 로 바뀌지만 실제 거리는 같습니다^^.



(질문) 크기와 방향이 있는 벡터에도 단위가 있으면 좋지 않을까요? 우리가 일상생활에서 왜 단위를 사용하죠?

- Unit vector 는 길이가 1인 벡터입니다.
  - 길이가 1 이라는 단위로 정해졌기 때문에 방향 성분만 의미가 있습니다.
  - 어떤 벡터 v 를 unit vector $\hat{v}$ 으로 만들기 위해서는 벡터의 크기로 나눠줍니다 (당연하겠죠?)

$$\hat{v} = \frac{v}{\|v\|}$$

$$\boldsymbol{v} = \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}, \qquad \|\boldsymbol{v}\| = \sqrt{2^2+1^2+3^2} = \sqrt{14}$$

$$\hat{\boldsymbol{v}} = \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|} = \begin{bmatrix} \frac{2}{\sqrt{14}} & \frac{1}{\sqrt{14}} & \frac{3}{\sqrt{14}} \end{bmatrix}^T$$

Unit vector (단위 벡터)

Unit vector (단위 벡터)

- 우리가 일상생활에서 왜 단위를 사용하죠?
  - 미터, 리터, 평, 킬로그램 ... ➔ 이런 것들이 왜 필요하죠?
  - 서로 다른 단위를 쓴다면 매번 자기네들 단위로 환산하는 불편함
    - 달러 → 원, 피트 → 미터, 파운드 → 킬로그램 ...

- 벡터도 마찬가지 입니다.
  - 어떤 벡터를 기준(단위)로 삼고 다른 벡터들을 비교하면 편리할 것입니다.
  - 단위 벡터는 길이가 1인 벡터입니다.
  - 1 이라는 숫자(단위)는 비교에 있어서 가장 편리하고 고마운 숫자입니다.

Orthogonal vector (직교 벡터)

Orthogonal ??

- 두 벡터의 내적이 0 인 상황을 의미합니다.
- 기하학적으로 직각을 이룰 때 입니다. 수학 기호는 ⊥ 와 같이 표기합니다.

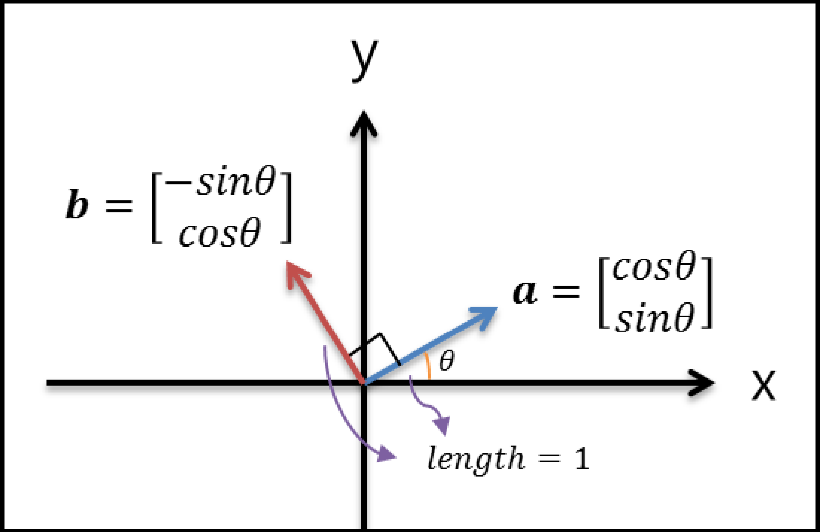
두 벡터 a 와 b 가 orthogonal 하다면,

- 다른 벡터와의 내적값은 0
- 자기 자신과의 내적값은 1

$$q_i^T q_j \begin{cases} 0 & if\ i \neq j \\ 1 & if\ i = j \end{cases}$$

단위 벡터가 직교하는 경우

- 정규직교벡터



이미지 출처: https://twlab.tistory.com/37

Orthogonal matrix (직교 행렬)

■ **행렬(matrix): 벡터들을 여러개 모아 놓은 것**

n 개의 특성

- 특성들
- Features
- feature vector

모두 같은 말

m 개의 데이터

키	몸무게	...	나이
183	65	...	26
:	:	⋱	:
168	72	...	21
a_1	a_2		a_n

$$A = \begin{pmatrix} a_1 & a_2 & \cdots & a_n \end{pmatrix} \xrightarrow[\text{(Transpose)}]{\text{전치행렬 구하기}} A^T = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{pmatrix}$$

Orthogonal matrix (직교 행렬) - 전치 행렬과의 곱셈

$$A^T A = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{pmatrix} \begin{pmatrix} a_1 & a_2 & \cdots & a_n \end{pmatrix} = \begin{pmatrix} 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 1 \end{pmatrix}$$

Orthogonal vector
1. 자기 자신과의 내적은 1
2. 다른 벡터와 곱은 0
   (왜냐하면 직각교차(직교) 하므로)

$X^T X$ 결과는
단위행렬(Identity matrix)

다시한번 SVD 특징을 기억해 봅니다.

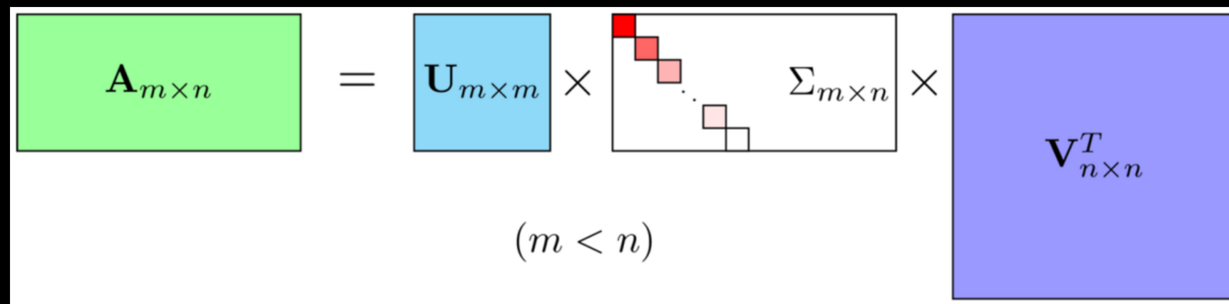
$A = U\Sigma V^T$

- 행렬의 크기
  A: $m \times n$ U: $m \times m$ Σ: $m \times n$
  V: $n \times n$
- 행렬의 특징
  행렬 U 와 V: Orthogonal matrix (직교 행렬)
  행렬 Σ: Diagonal Matrix (대각 행렬)

SVD의 숨겨진 비밀을 찾아서

다시 한번 SVD 특징을 기억해 봅니다. 기억 나시죠 ^^ ?

$A = U\Sigma V^T$



- 행렬의 크기
  - A: $m \times n$
  - U: $m \times m$
  - Σ: $m \times n$
  - V: $n \times n$

- 행렬의 특징
  - 행렬 U 와 V: Orthogonal matrix (직교 행렬)
  - 행렬 Σ : Diagonal Matrix (대각 행렬)

헐!!!
헉!!
완전 대박!

행렬 V 의 열 벡터(column vector)는 A^TA 의 고유 벡터들 입니다.

행렬 Σ 의 대각선 원소는 A^TA 의 고유 값들 입니다.

교수님~
Eigen vector, eigen value라구요? 왜 그렇죠?
➔ 다음 슬라이드에서 설명하겠습니다.

SVD에서 eigen vector, eigen value 뽑아 내기

$A = U\Sigma V^T$

- 행렬의 크기

  $A: m \times n$

  $U: m \times m$

  $\Sigma: m \times n$

  $V: n \times n$

- 행렬의 특징

  U 와 V: Orthogonal matrix

  Σ : Diagonal Matrix

먼저 A^T 를 구해 봅니다.

$A^T = (U\Sigma V^T)^T = V\Sigma^T U^T$ 입니다.

행렬 A^T 와 A 를 곱합니다.

$A^T A = V\Sigma^T U^T U \Sigma V^T$

Orthogonal matrix U^T 와 U 를 곱하면 identity matrix 입니다(소거됨).

$A^T A = V\Sigma^T \Sigma V^T$

Diagonal matrix Σ^T 와 Σ 를 곱하면 자기 자신의 제곱이 됩니다.

$A^T A = V\Sigma^2 V^T$

양변에 V 를 곱합니다.

$A^T A V = V\Sigma^2 V^T V = V\Sigma^2$

$V = [v_1, v_2, \cdots, v_n]$ 을 대입해서 전개

$$A^T A\ [v_1, v_2, \cdots, v_n] = [v_1, v_2, \cdots, v_n]\Sigma^2 = [\sigma_1^2 v_1, \sigma_2^2 v_2, \cdots, \sigma_n^2 v_n]$$

각 원소를 일반화 합니다.

$$\boxed{A^T A}\ \boxed{v_i} = \boxed{\sigma_i^2}\ \boxed{v_i}$$

$$\boxed{\boldsymbol{A}}\ \boxed{\boldsymbol{v}} = \boxed{\boldsymbol{\lambda}}\ \boxed{\boldsymbol{v}}$$

Eigen vector, eigen value의 정의와 정확히 일치합니다!

PCA using SVD 1

- **SVD를 이용한 PCA 수행**
- **먼저 데이터를 수집합니다.**
  - 2개의 Feature를 갖는 1,000개의 데이터셋을 구축했다고 가정하겠습니다.
- **Mean Centering을 수행합니다.**

행렬 Z

	Z1	Z2
0	2.0046	-1.0718
1	0.1823	0.2250
2	-0.5179	0.2620
3	-0.6507	-1.7142
4	0.3236	-3.0723

Mean Centering

각 feature (Z1, Z2)의 평균값을 빼줍니다.

Z1 평균: -0.0289
Z2 평균: 0.0288

행렬 A

	A1	A2
0	2.0334	-1.1006
1	0.2112	0.1962
2	-0.4909	0.2332
3	-0.6219	-1.7430
4	0.3525	-3.1011

PCA using SVD 2

- **SVD를 수행합니다.**
  - 자료가 많아서 3개로 줄여서 표현하였습니다. ➔ 실제 SVD 계산은 컴퓨터가 합니다.

행렬 Σ 의 대각선 원소는 $A^T A$ 의 고유 값들 입니다.

$$A = U\Sigma V^T = \begin{bmatrix} -0.0309 & -0.0216 \\ -0.0001 & -0.0092 \\ 0.0071 & 0.0060 \end{bmatrix} \begin{bmatrix} 71.54 & 0 \\ 0 & 31.15 \end{bmatrix} \begin{bmatrix} -0.70 & 0.71 \\ 0.71 & 0.70 \end{bmatrix}$$

행렬 V 의 열 벡터(column vector)는 $A^T A$ 의 고유 벡터들 입니다.

$$Cov(A) = \frac{A^T A}{n-1}$$

$A^T A v_i = \sigma_i^2 v_i$ 이므로

$$\frac{A^T A}{n-1} v_i = \boxed{\frac{\sigma_i^2}{n-1}} v_i$$

Eigen values

$$\lambda_1 = \frac{71.54^2}{1000-1} = 5.12$$

$$\lambda_2 = \frac{31.15^2}{1000-1} = 0.97$$

행렬 V 는 V^T 의 전치행렬이므로

$$V = \begin{bmatrix} -0.70 & 0.71 \\ 0.71 & 0.70 \end{bmatrix}$$

Eigen vector 는

$$v_1 = \begin{bmatrix} -0.70 \\ 0.71 \end{bmatrix} \quad v_2 = \begin{bmatrix} 0.71 \\ 0.70 \end{bmatrix}$$

PCA using SVD 3

- **PC (Principal Component) score 구하기**

$$AV = U\Sigma = \begin{bmatrix} -0.0309 & -0.0216 \\ -0.0001 & -0.0092 \\ 0.0071 & 0.0060 \end{bmatrix} \begin{bmatrix} 71.54 & 0 \\ 0 & 31.15 \end{bmatrix} = \begin{bmatrix} -2.21 & -0.67 \\ -0.71 & -0.29 \\ 0.51 & 0.19 \end{bmatrix}$$

PC1 PC2

- **구한 PC 값들을 이용하여 다양한 분석 수행**
- **여기까지 따라 왔으면, 이제는 PCA를 끝낸 겁니다.^^**

교수님~
PCA 수행과정 (PC score 구하기)는 알겠는데요…
그래서 PCA가 도대체 머래요?

Analysis on PCs

PC score 가 도대체 어떤 의미인가?

■ PC score 가 도대체 어떤 의미인가?

$$A = U\ \Sigma\ V^T$$

$$\boxed{A}\ \boxed{V} = U\ \Sigma$$

우리가 확보한 데이터셋
(mean centering 수행된 것)

$$A = \begin{pmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_n \\ | & | & & | \end{pmatrix}$$

$$A^T = \begin{pmatrix} \text{—} a_1 \text{—} \\ \text{—} a_2 \text{—} \\ \vdots \\ \text{—} a_k \text{—} \end{pmatrix} = \begin{pmatrix} {a_1}^T \\ {a_2}^T \\ \vdots \\ {a_n}^T \end{pmatrix}$$

$$a_i^T = [a_{i,1}, a_{i,2}, \cdots, a_{i,m}]$$

$$V = [v_1, v_2, \cdots, v_n]$$

$$v_i = \begin{bmatrix} v_{i,1} \\ v_{i,2} \\ \vdots \\ v_{i,m} \end{bmatrix}$$

n 개의 특성 (feature)

m 개의 데이터

키	몸무게	...	나이
183	65	...	26
:	:	⋱	:
168	72	...	21

a_1 a_2 a_n

헐... 교슈님~ ㅠㅠㅠ
내적 하네요? 그게 도데채 무슨 의미래요?

i 번째 관측치의 1번째 PC score = $a_i^T v_1$

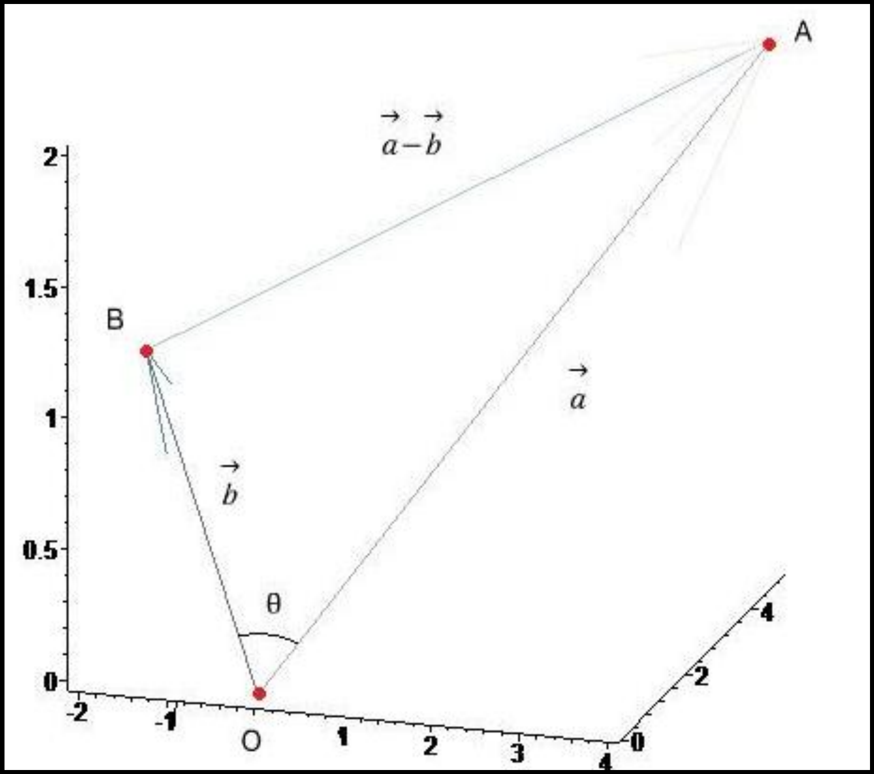
i 번째 관측치의 PC score (vector)
= $a_i^T V = [a_i^T v_1, a_i^T v_2, \cdots, a_i^T v_n]$

벡터 내적 (inner product) 의미

■ **벡터의 내적 (inner product 또는 dot product 라고 부르기도 합니다. ^^)**

$$A = [a_1, a_2, \cdots, a_n] \qquad B = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

$$AB = [a_1 b_1 + a_2 b_2 + \cdots + a_n b_n]$$



삼각형 OAB에 제2 코사인법칙 적용

$$\|a - b\|^2 = \|a\|^2 + \|b\|^2 - 2\|a\|\|b\| \cos\theta$$

한편, 아래 공식도 성립

$$\begin{aligned} \|a - b\|^2 &= (a - b) \cdot (a - b) \\ &= \|a\|^2 - 2a \cdot b + \|b\|^2 \end{aligned}$$

위 두 식을 결합하면

$$\|a\|^2 + \|b\|^2 - \|a\|\|b\| \cos\theta$$
$$= \|a\|^2 - 2a \cdot b + \|b\|^2$$

결국에는

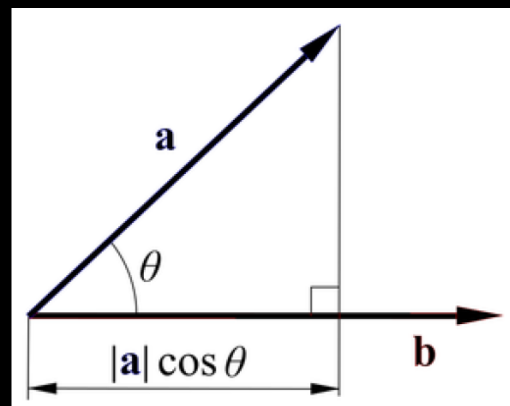
$$\begin{cases} a \cdot b = \|a\|\|b\| \cos\theta \\ \cos\theta = \dfrac{a \cdot b}{\|a\|\|b\|} \end{cases}$$

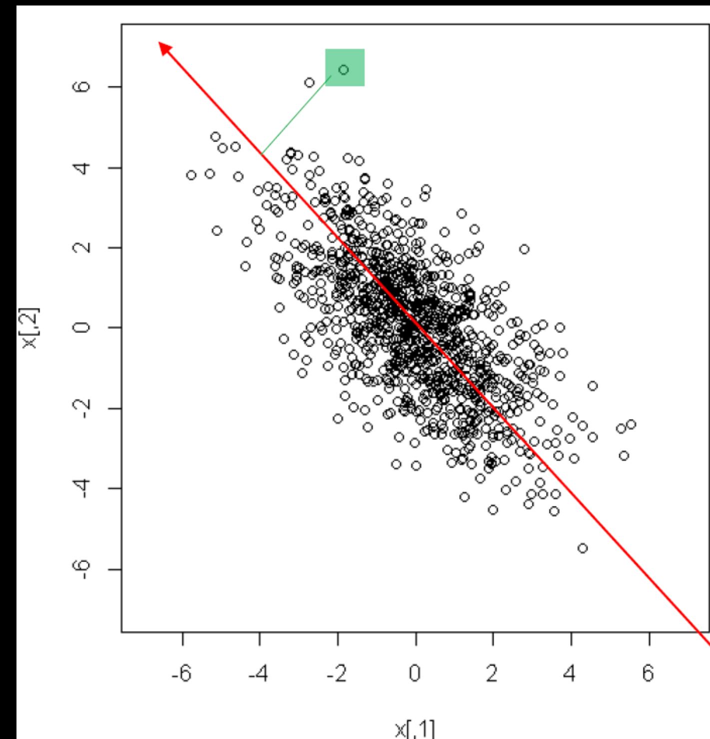
벡터 내적과 정사영(Projection)

■ 두 벡터를 내적한다는 의미

$$a \cdot b = \|a\|\|b\| \cos\theta$$
$$= \boxed{\|a\| \cos\theta} \|b\|$$

벡터 a 를 b 에 정사영 하고,
b 의 크기(단위)만큼 곱한다.





하나의 관측치 벡터 a_i^T를 어떤 벡터 v_i와 내적한다.

하나의 관측치 벡터 a_i^T를 벡터 v_i에 정사영(Projection) 한다.

하나의 관측치 벡터 a_i^T를 벡터 v_i 직선상에 투사한다.

벡터 a_i^T 와 v_i 를 v_i 기준으로 차원 축소한다.

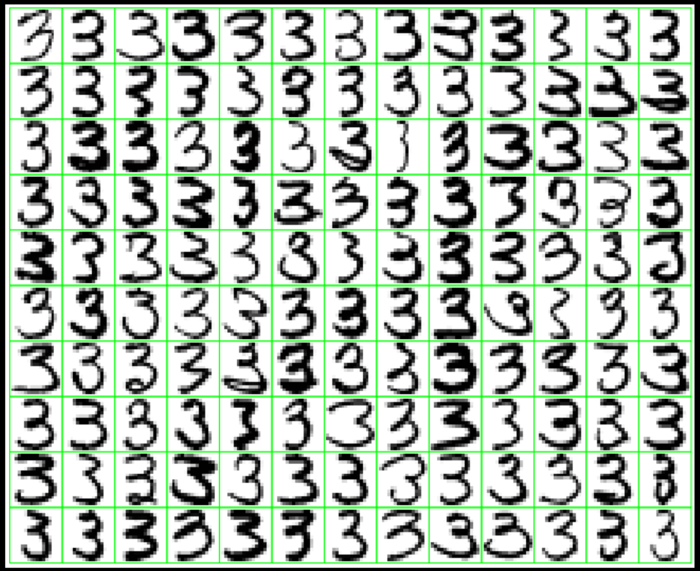
모두 같은 뜻 입니당^^

PCA 실습 – 손글씨 `3` 차원 축소 및 해석 1

■ 손글씨 데이터셋

- 손글씨로 숫자 `3`을 기록 → 130 장 수집
- 한 장의 이미지는 16×16 픽셀로 구성
  - 이미지를 1차원 벡터로 만들면 $16 \times 16 = 256$

$$X \xrightarrow{centering} A \in \mathbb{R}^{130\times 256}$$

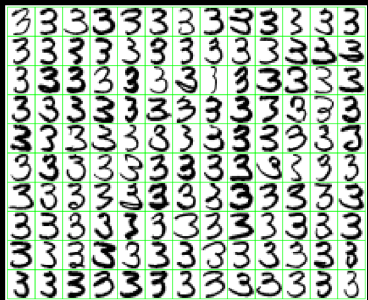


256개의 특성 (feature)

	px1	px2	px3	⋯	px256
1번째 이미지	42	255	120	⋯	37
2번째 이미지	41	243	115	⋯	41
⋮	⋮	⋱	⋱	⋱	⋮
⋮	⋮	⋱	⋱	⋱	⋮
⋮	⋮	⋱	⋱	⋱	⋮
⋮	⋮	⋱	⋱	⋱	⋮
130번째 이미지	37	215	130	⋯	45

이미지 출처:
1. The Elements of Statistical Learning, Tibshirani and Friedman
2. https://heung-bae-lee.github.io/2020/04/03/machine_learning_07/

PCA 실습 – 손글씨 `3` 차원 축소 및 해석 2



px1	px2	px3	⋯	px256
42	255	120	⋯	37
41	243	115	⋯	41
⋮	⋱	⋱	⋱	⋮
⋮	⋱	⋱	⋱	⋮
⋮	⋱	⋱	⋱	⋮
⋮	⋱	⋱	⋱	⋮
37	215	130	⋯	45

$$X \xrightarrow{centering} A \in \mathbb{R}^{130\times256}$$

$A : m\times n$ 직사각형 행렬

$U : m\times m$ 직교 행렬

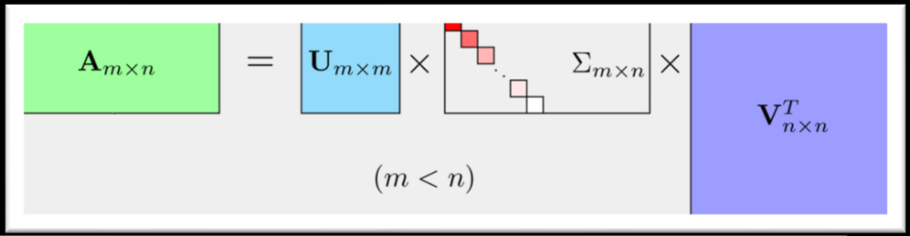
Σ : m×n 대각 행렬

V : n×n 직교 행렬

PC score 구하기

$$A = U\ \Sigma\ V^T$$

- Full SVD
모든eigen vector 계산



- Truncated SVD:
Eigne value (Σ 의 대각원소)를 내림차순 정렬하고 값이 큰 k 개에 대응하는 eigen vector만 계산

eigen value가 크다는 의미: 데이터를 가장 잘 설명할 수 있다.

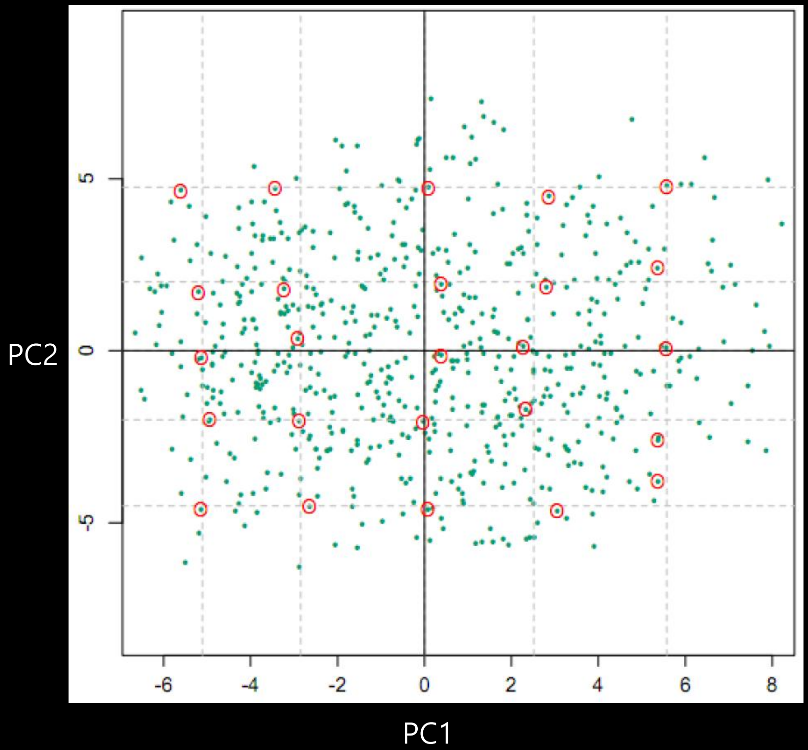
$AV = U\Sigma$ 🡐 $m\times n$ 직사각형 행렬, 차원 유지

$AV = U\Sigma$ 🡐 $m\times\ k$ 직사각형 행렬, 차원 축소
만약 $k = 2$ 일 경우 $(130\times256) \rightarrow (130\times2)$
Feature 차원을 256 → 2차원으로 감소

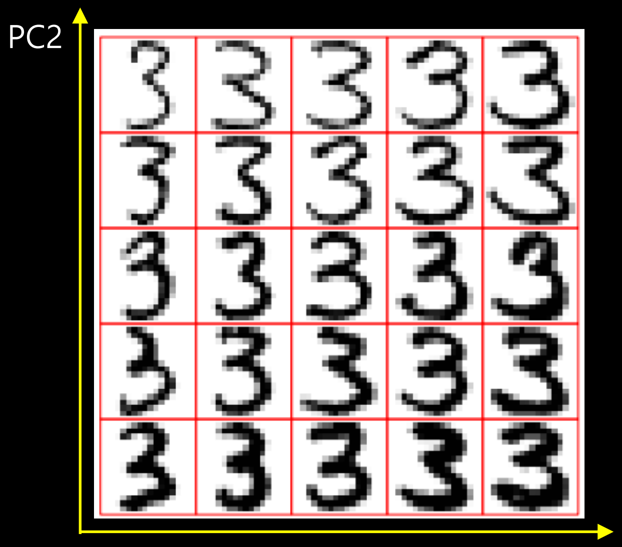
PCA 실습 – 손글씨 `3` 차원 축소 및 해석 3

■ 개별 PC 의미 해석

- 손글씨 `3` 이미지를 2차원으로 축소 → 2개의 PC score vector
- PC1 과 PC2 가 각각 새로운 축이 됨 (256 차원 → 2 차원 평면)



- 데이터 개수를 기준으로 일정 quantile로 격자 생성
- 각 분위(quantile) 별로 가까운 점을 선택
- 선택된 점(관측 데이터)의 원래 데이터를 표시



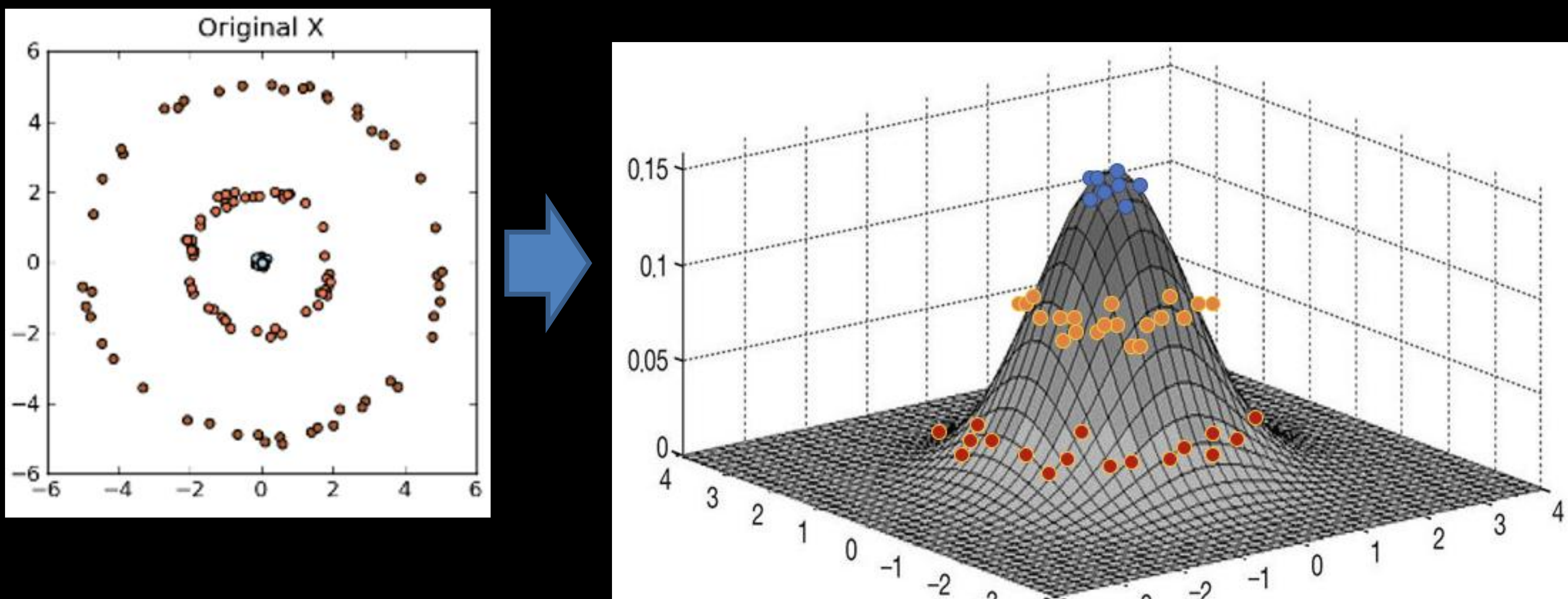
PC1: 숫자 `3` 의 아래쪽 고리 길이 설명
PC2: 손글씨 두께를 설명

참고: PC1의 분산(범위)가 PC2 보다 큽니다.
왜냐하면 PC1의 eigen value가 큰 것을 선택했기 때문입니다.

Kernal PCA

- **엥! PCA도 어려운데 Kernal 이라구요??**
- **그렇다면 다음과 같은 데이터는 어떻게 해석하실 건가요?**
  - x_1 과 x_2 에서 존재하는 데이터는 어떤 관계는 있지만, 선형적이지 않습니다. ㅠㅠ

차원을 축소하는 대신, 고차원 공간으로 맵핑!



Thank you!