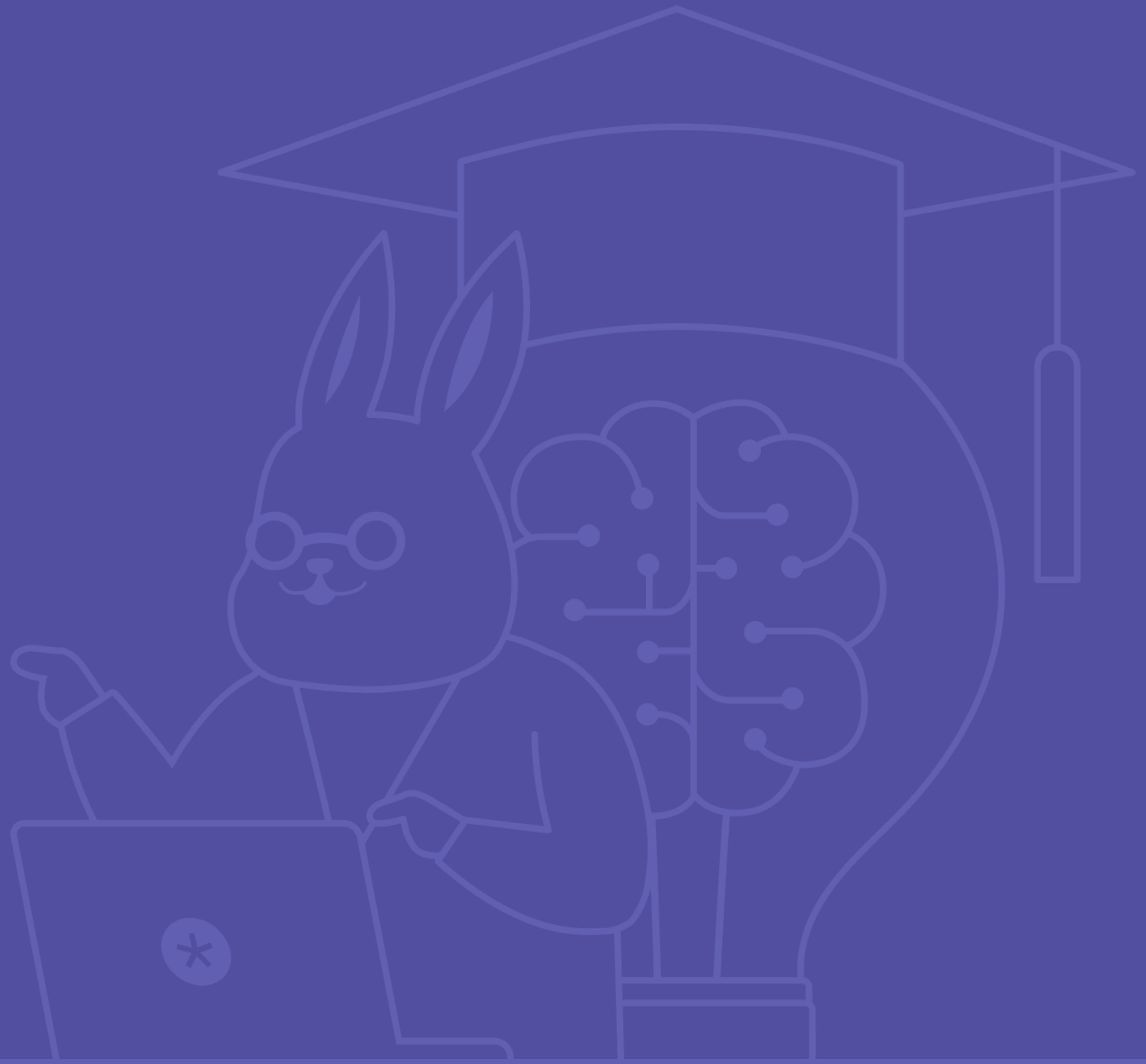
2020 AI College

8장 비지도 학습 – 차원 축소

정민수 강사



Confidential all right reserved

/* elice */

Contents

Target

차원 축소의 필요성을 인지한다.

차원의 저주가 왜 발생하고 차원 축소의 필요성을 인지한다.

PCA 기법에 대하여 이해한다.

차원 축소를 위한 대표 기법인 PCA 기법에 대하여 이해한다.

PCA 외 다른 차원 축소 기법에 대하여 살펴본다.

차원 축소의 다른 기법인 CUR 분해와 시각화 방법론 tSNE에 대하여 살펴본다.

01

차원의 저주와 차원 축소

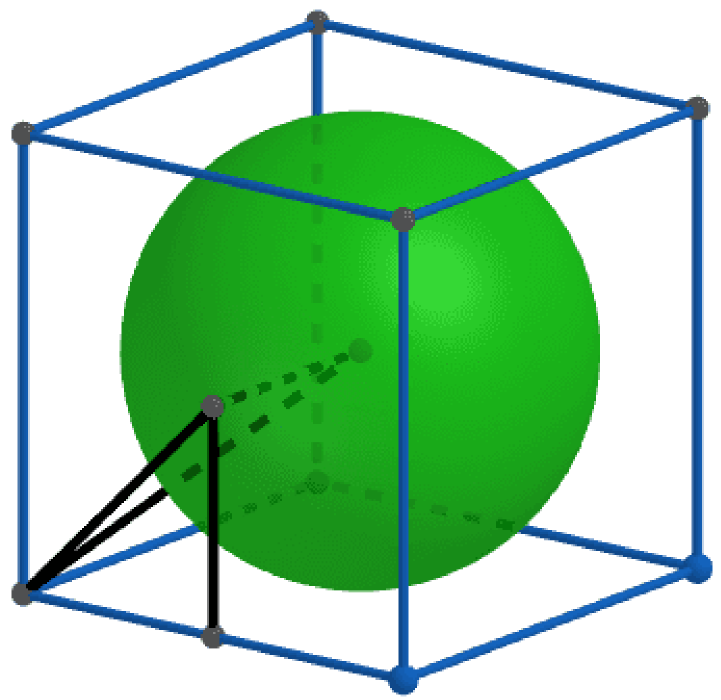
Confidential all right reserved

차원의 저주

- 저차원 vs. 고차원: 저차원에서의 직관이 성립하지 않음
- 2차원의 단위면적을 가진 정사각형 안에 있는 점을 무작위로 선택할 때 가장자리에 있는 점을 선택할 가능성은 매우 낮음
- 10,000차원의 단위면적을 가진 초입방체(hyper cube)에서는 이 가능성이 99.99 이상

- 2차원의 단위 정사각형에서 임의의 두 점을 선택하면 두 점 사이의 평균거리는 0.52
- 1,000,000차원의 단위 초입방체에서 임의의 두 점을 선택하면 두 점 사이의 평균거리는 428.25

차원의 저주



- 한변의 길이가 2r인 초입방체의 부피는 $V_{n-cube} = (2r)^n$
- 한변의 길이가 r인 초구(hyper sphere)의 부피는

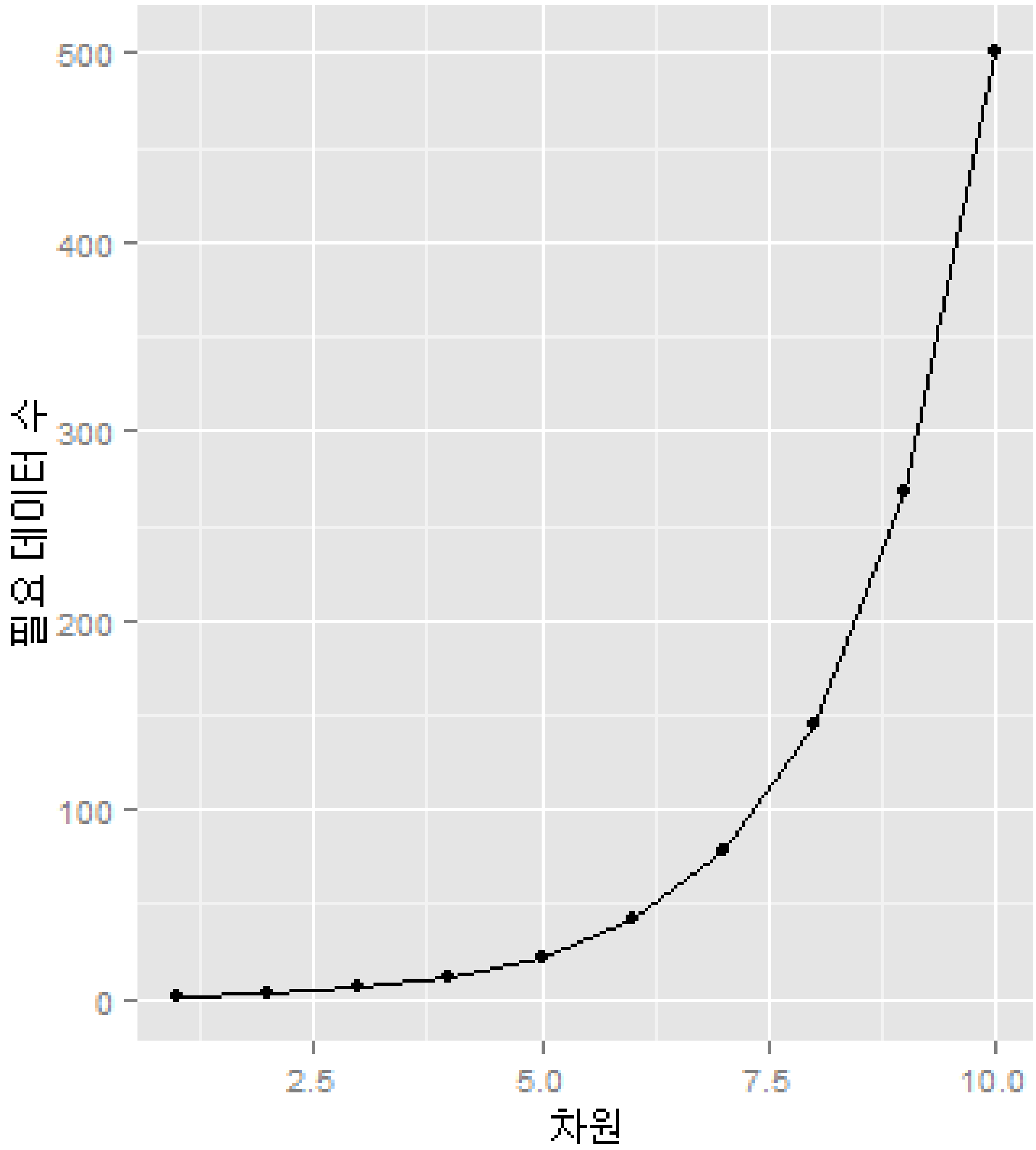
$$V_{n-sphere} = \frac{2r^n}{n\,\Gamma(\frac{n}{2})}\pi^{\frac{n}{2}}$$

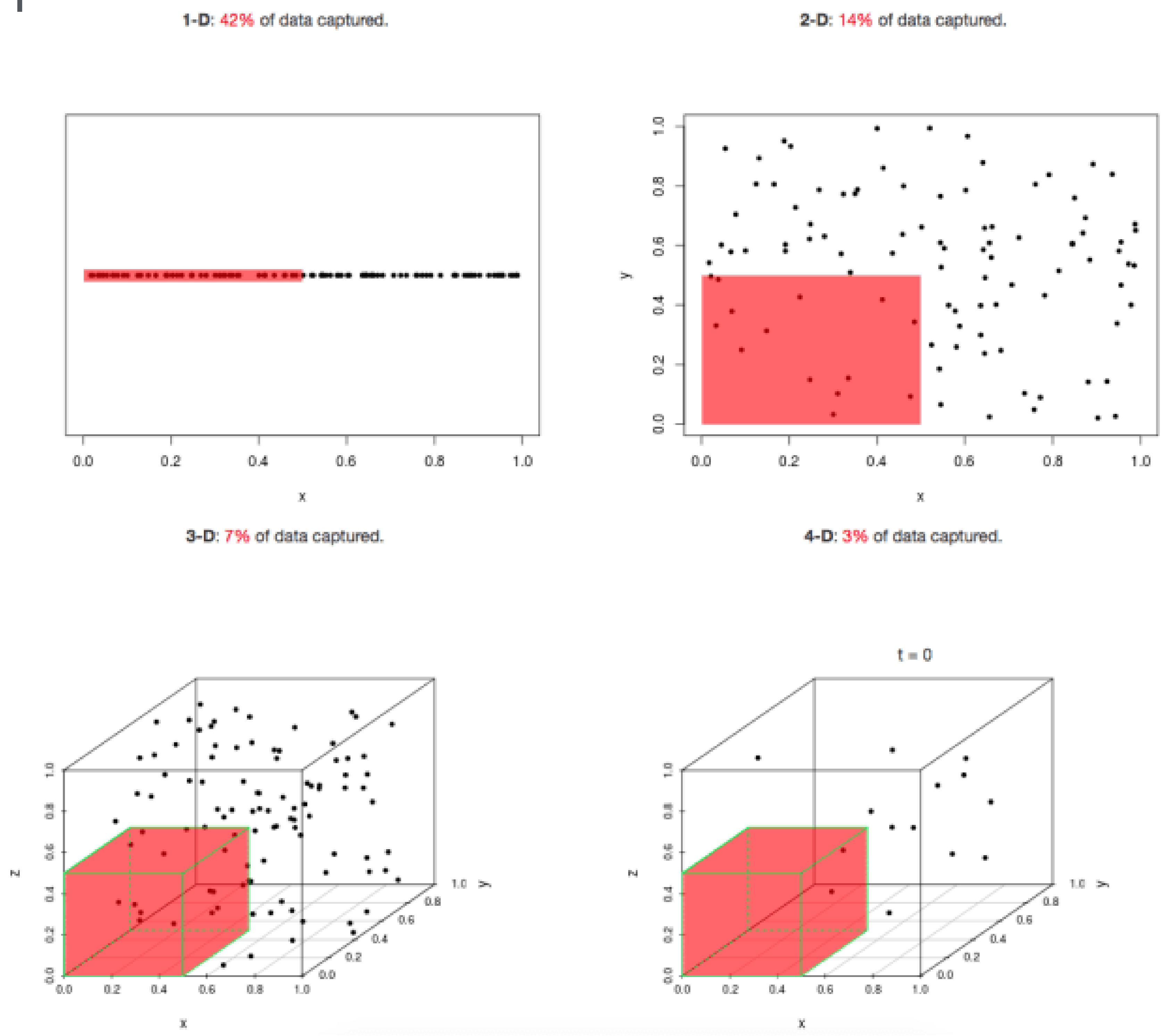
- 차원이 커지면 초입방체에 내접한 초구의 부피의 비율은 0으로 수렴

$$\frac{V_{n-sphere}}{V_{n-cube}} = \frac{2}{n\,\Gamma\left(\frac{n}{2}\right)} \times \left(\frac{\sqrt{\pi}}{2}\right)^n \rightarrow 0$$

차원의 저주

- 고차원일수록 전체에서 데이터가 차지하는 공간이 매우 적어진다
- 필요한 데이터 양이 기하급수적으로 증가

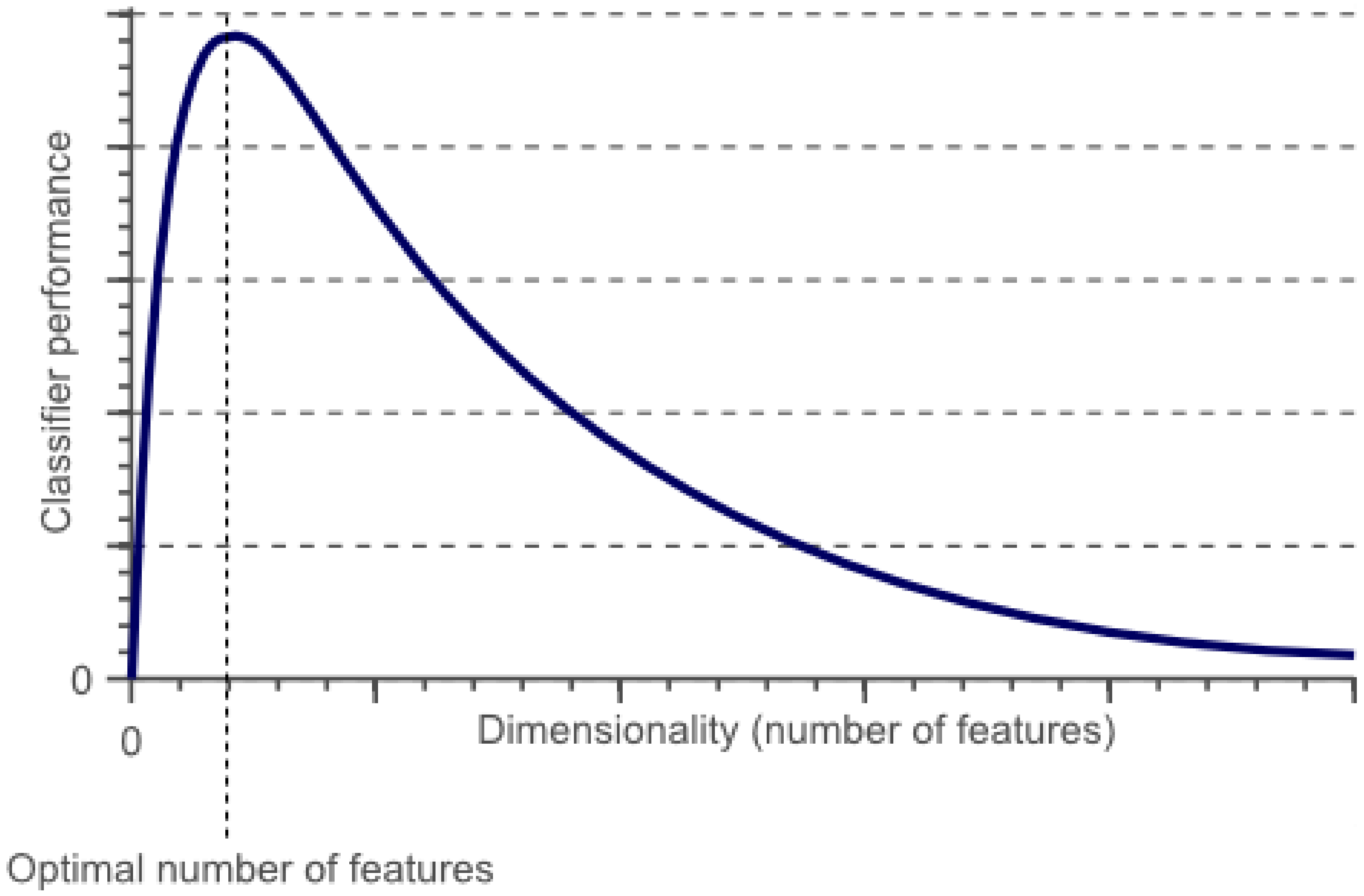




/* elice */

차원의 저주

- 훈련샘플 각각이 엄청나게 많은(Ex. 수백만 개) 특성을 가지고 있을 때 훈련이 느려질 뿐만 아니라, 최적의 솔루션을 찾기 어려워지는 현상
- Ex) 일정 차원을 넘으면 분류기의 성능은 점점 떨어져 0으로 수렴



차원 축소

- 고양이들에게는 비슷한 점들이 많음
- 굳이 모든 픽셀을 다 보지 않고도 중요한 특징을 잡아낼 수 있음

차원 축소

- 관찰 대상들을 잘 설명할 수 있는 잠재 공간(latent space)은 실제 관찰 공간(observation space)보다 작을 수 있음

- 차원 축소란?
  관찰 공간 위의 샘플들에 기반으로 잠재 공간을 파악하는 것

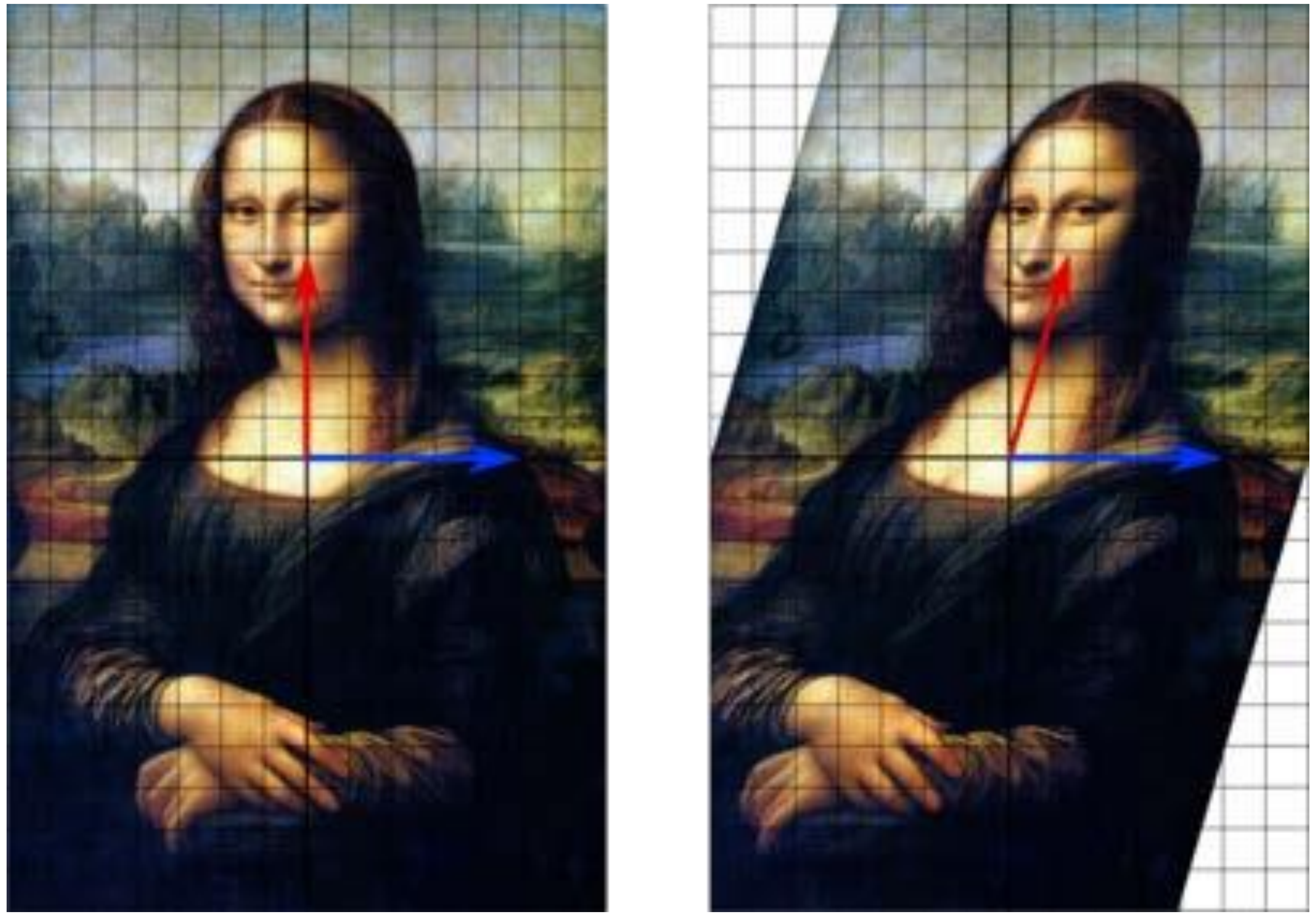
02

Principal Component Analysis (PCA)

Confidential all right reserved

선형 변환

- 아래와 같은 선형변환에 의해 파란색 벡터는 방향이 변하지 않음



고유값(Eigen value)과 고유벡터(Eigen vector)

- 정사각행렬 A 에 대해 영벡터가 아닌 벡터 x에 대해 $Ax = \lambda x$일 때,
- λ를 고유값(Eigen value), x를 고유벡터(Eigen vector)라 함

특성방정식(Characteristic Equation)

- $(\lambda I - A)x = 0$의 영공간(Null space)이 영벡터가 아닌 벡터를 포함하려면 $\det(\lambda I - A) = 0$이 성립해야 한다.
- 이를 A의 특성방정식이라 한다.
- 특성방정식은 최고차항 계수가 1인 n차 방정식

대각화(Diagonalization)

- 고유값들을 대각성분으로 갖는 행렬을 D, 고유값들에 대응하는 고유벡터들을 열벡터로 갖는 행렬을 Q라 하면 $A = QDQ^{-1}$과 같이 표현 가능하고, 이를 대각화라 한다.

- 대칭(real Symmetric)행렬은 항상 직교(orthogonal) 행렬로 대각화 할 수 있다.

/* elice */

고유값과 고유벡터 계산 예제

Example 6.1.1 Find the eigenvalues and eigenvectors of

$$A = \begin{bmatrix} 2 & \sqrt{2} \\ \sqrt{2} & 1 \end{bmatrix}.$$

Solution: The characteristic polynomial is

$$\det(\lambda I - A) = \det \begin{bmatrix} \lambda - 2 & -\sqrt{2} \\ -\sqrt{2} & \lambda - 1 \end{bmatrix} = \lambda^2 - 3\lambda = \lambda(\lambda - 3).$$

Thus the eigenvalues are $\lambda_1 = 0$ and $\lambda_2 = 3$. To determine the eigenvectors belonging to λ_i's, we should solve the homogeneous system of equations $(\lambda_i I - A)\mathbf{x} = 0$ for each λ_i's.

고유값과 고유벡터 계산 예제

For $\lambda_1 = 0$, the system of equations $(\lambda_1 I - A)\mathbf{x} = \mathbf{0}$ becomes

$$\begin{cases} -2\,x_1 & - & \sqrt{2}\,x_2 & = & 0, \\ -\sqrt{2}\,x_1 & - & x_2 & = & 0, \end{cases} \quad \text{or} \quad x_2 = -\sqrt{2}\,x_1.$$

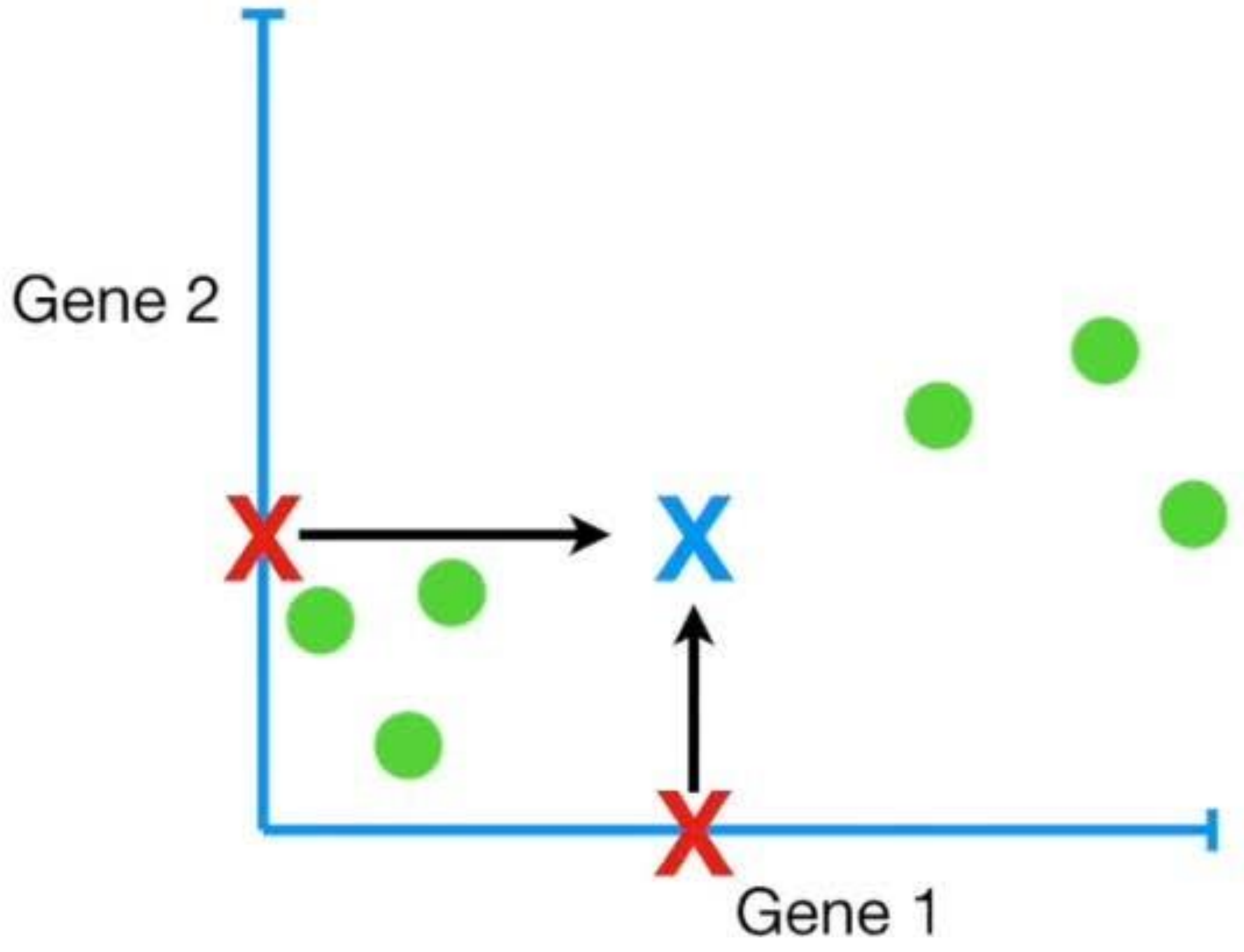
Hence, $\mathbf{x}_1 = (x^1,\ x^2) = (-1,\ \sqrt{2})$ is an eigenvector belonging to $\lambda_1 = 0$, and $E_0 = \{t\mathbf{x}_1\ :\ t \in \mathbb{R}\}$.

For $\lambda_2 = 3$, an eigenvector belonging to $\lambda_2 = 3$, as the solutions of the system of equations $(\lambda_2 I - A)\mathbf{x} = \mathbf{0}$, is $\mathbf{x}_2 = (\sqrt{2},\ 1)$, and so $E_3 = \{t\mathbf{x}^2\ :\ t \in \mathbb{R}\}$. Note that the eigenvectors $\mathbf{x}_1$ and $\mathbf{x}_2$ belonging to the eigenvalues λ_1 and λ_2 respectively are linearly independent. $\square$

PCA란?

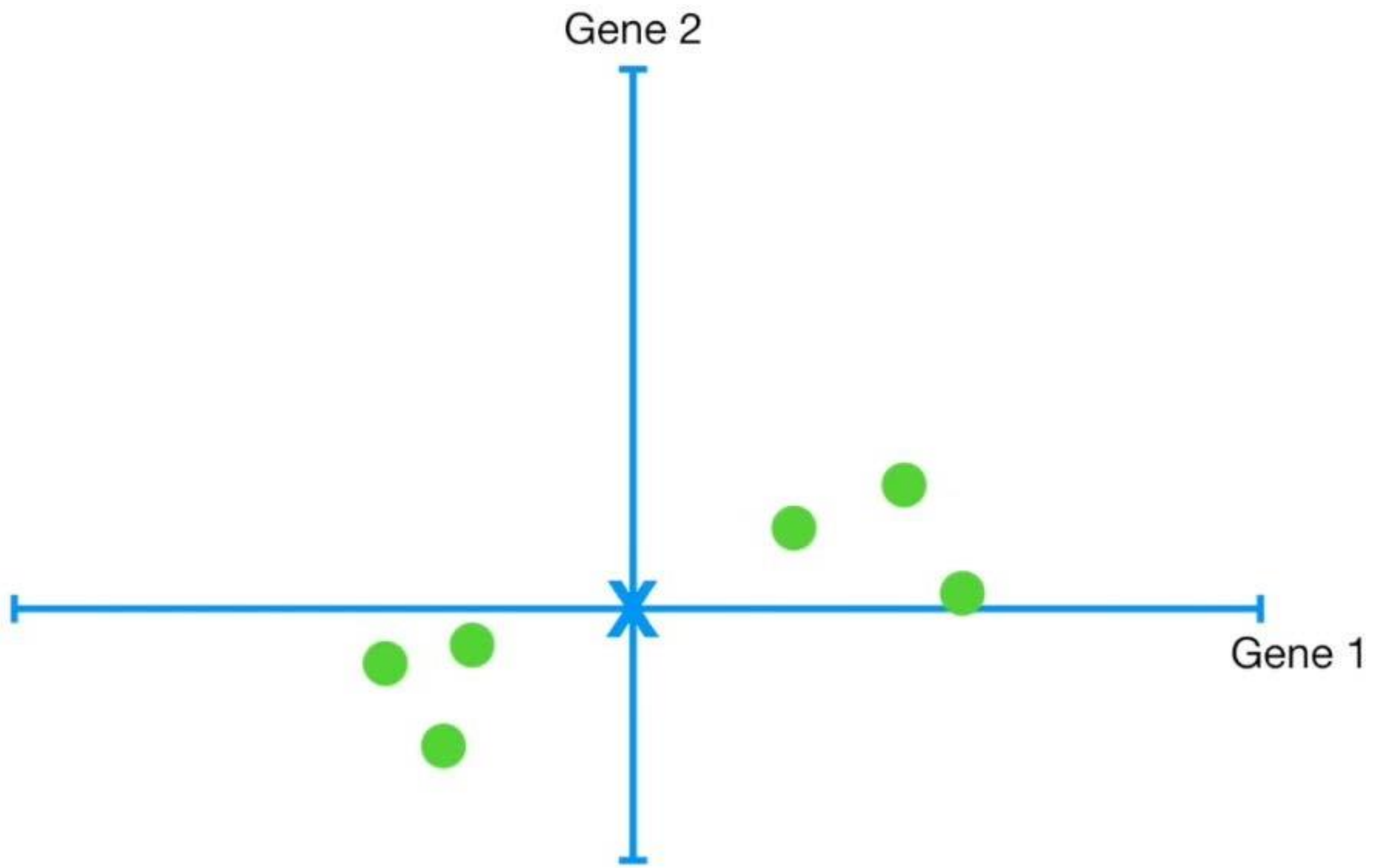
- 피쳐가 2개, 샘플의 개수가 6개인 데이터를 시각화하고 평균을 표시

	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1



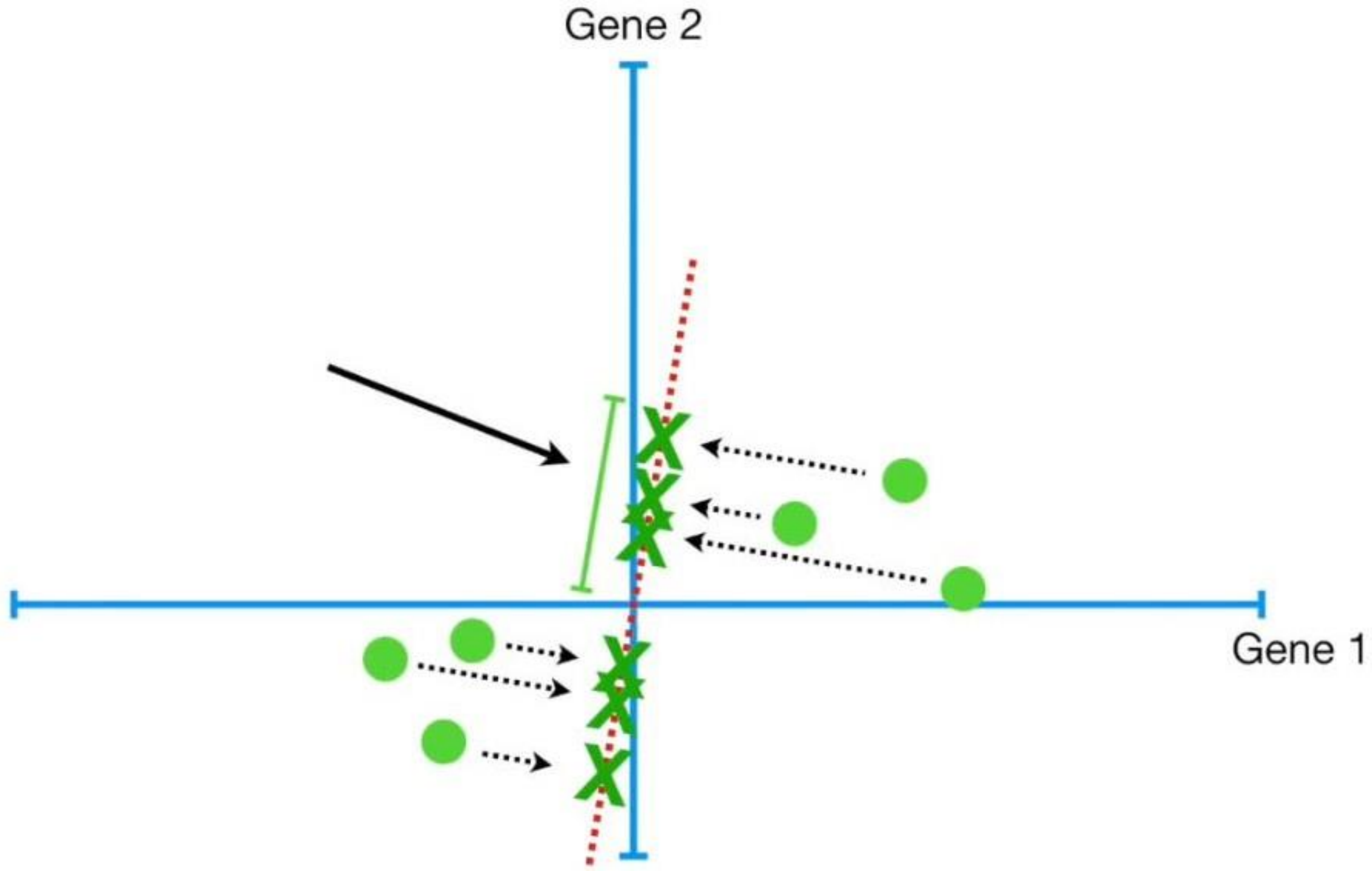
PCA란?

- 모든 데이터에서 각 행의 평균을 빼서 모든 행의 평균이 0이 되게 함 (centering 작업)



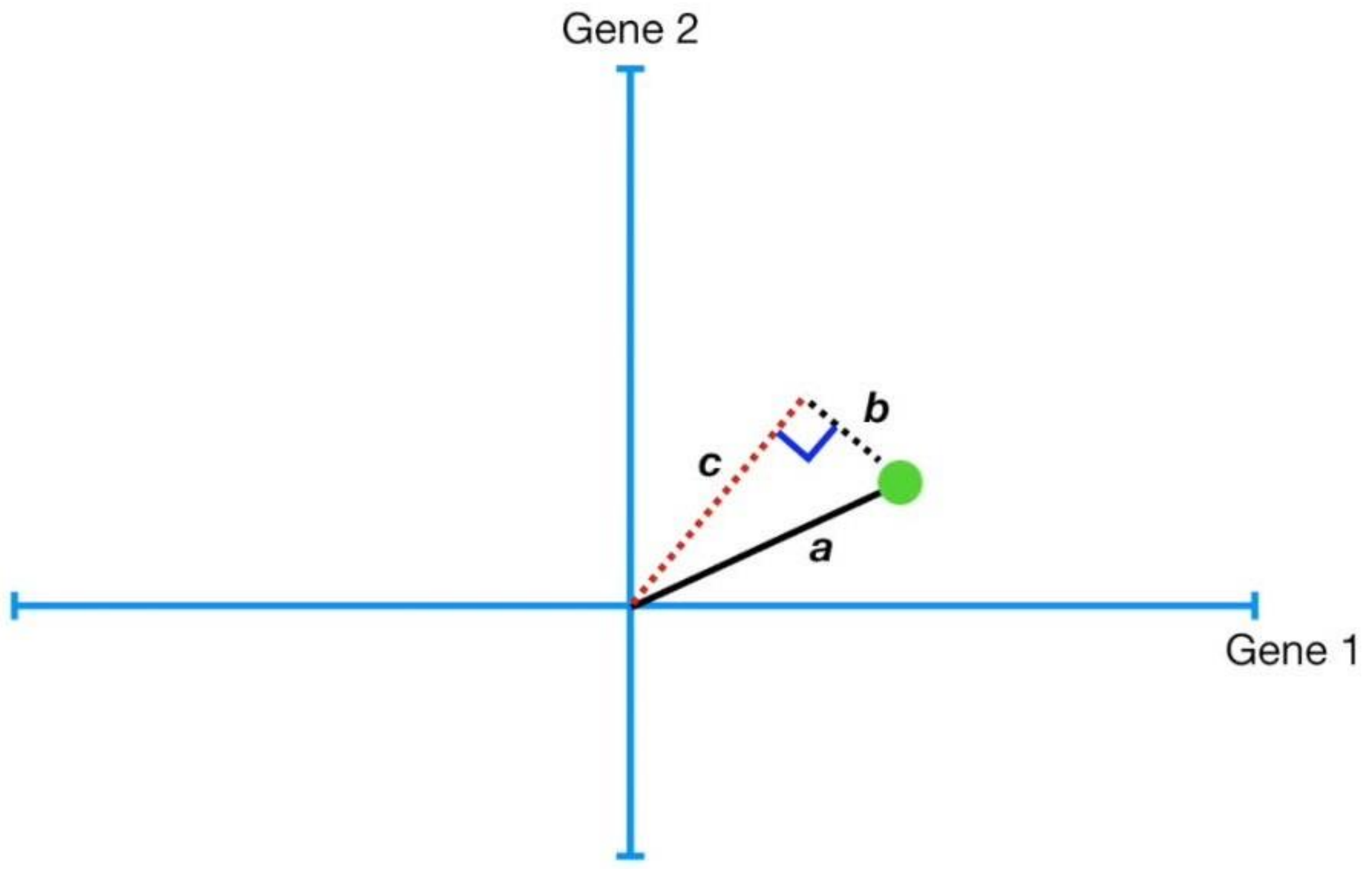
PCA란?

- 원점을 지나는 직선 중에서 데이터들을 정사영 했을 때의 분산을 최대로 하는 직선을 찾고자 함



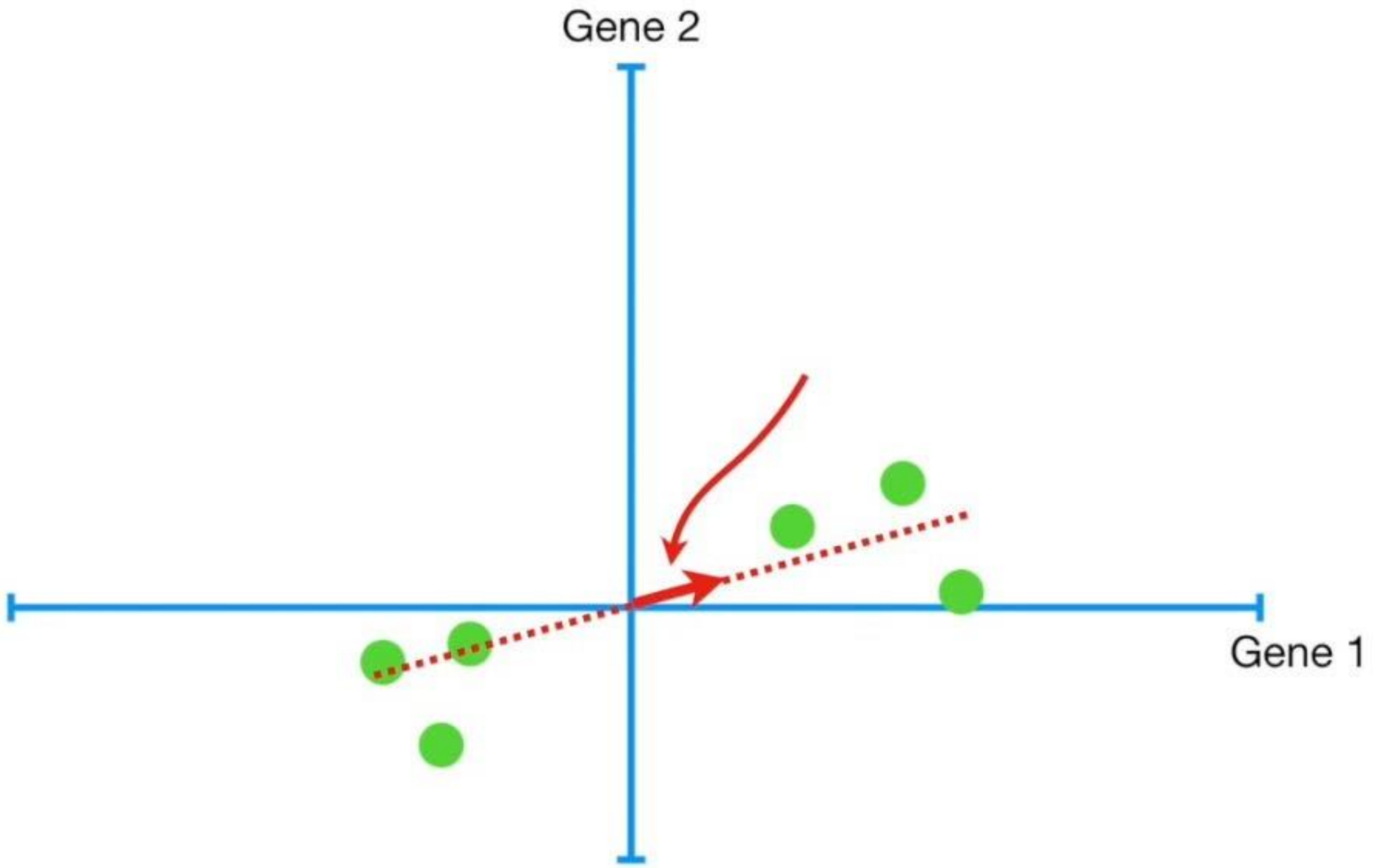
PCA란?

- 피타고라스 정리에 의해 $a^2 = b^2 + c^2$이므로 결국 정사영 했을 때의 분산을 최대화하는 것은 각 점에서 직선까지의 거리 제곱의 합을 최소화하는 것과 같음



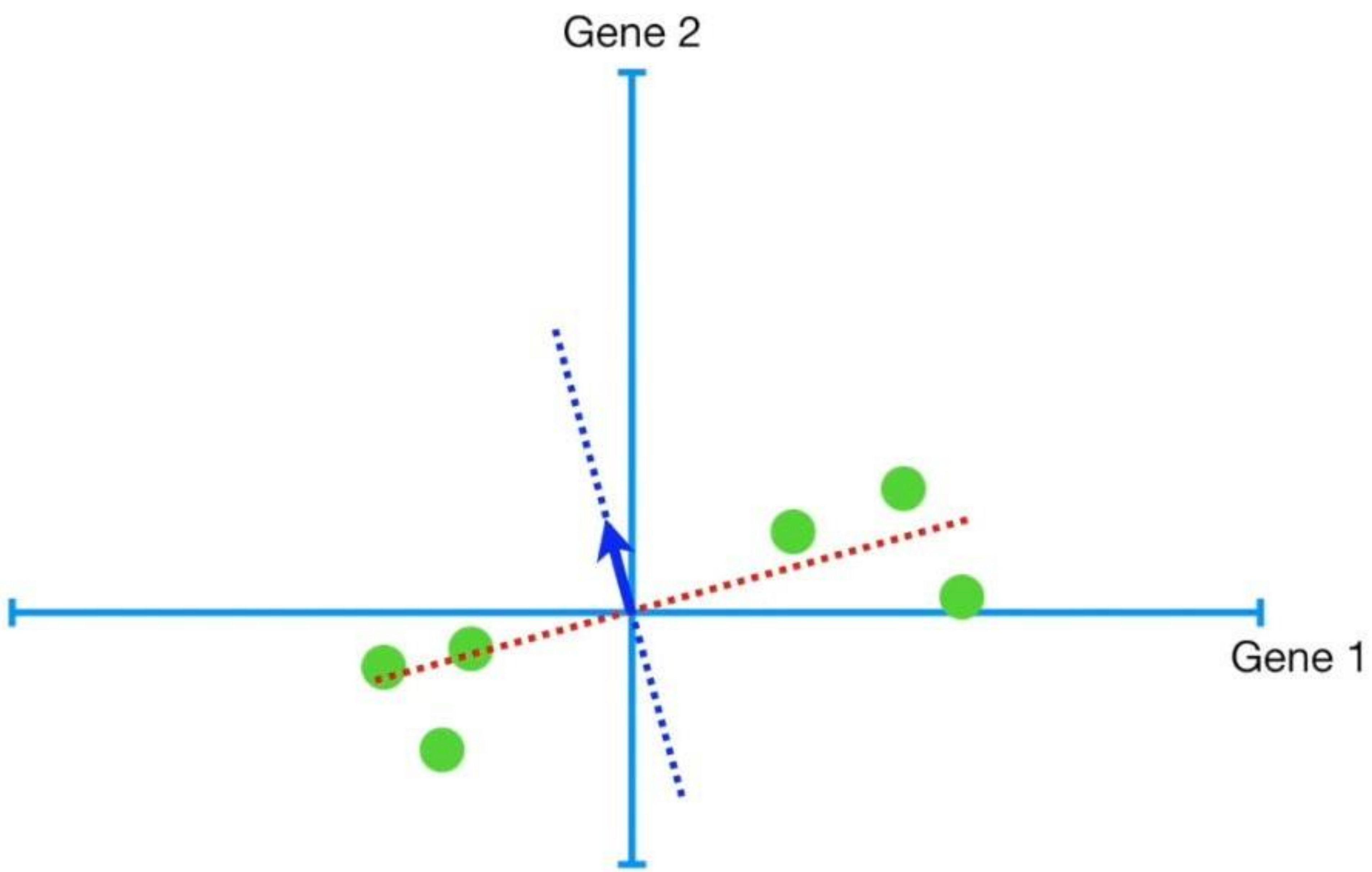
PCA란?

- 그렇게 찾은 직선을 첫번째 주성분(PC_1), 빨간색 벡터를 PC_1의 싱귤러(singular) 벡터라고 함
- PC_1 의 싱귤러 벡터는 공분산 행렬의 가장 큰 고유값(λ_1)에 대한 고유벡터가 됨



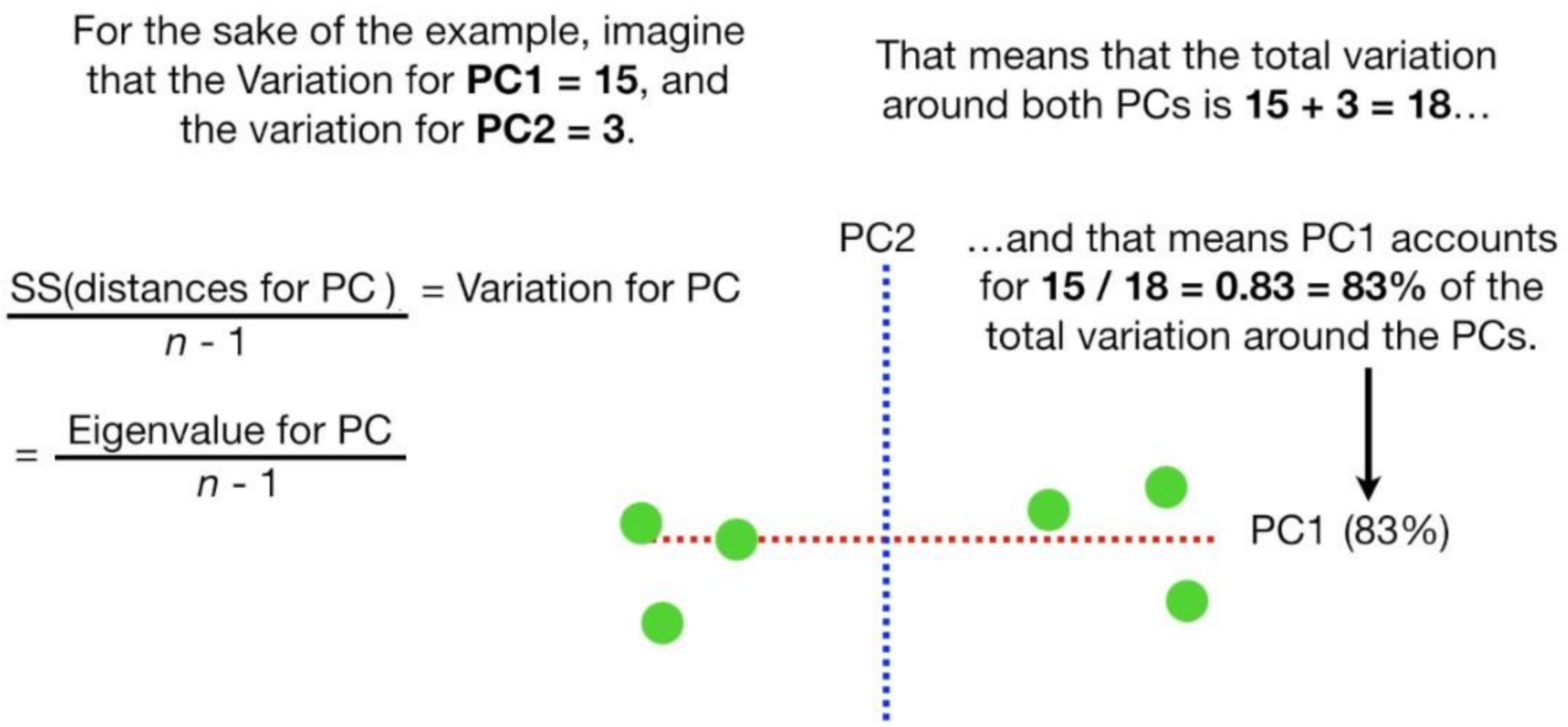
PCA란?

- PC_2는 PC_1과 수직인 직선 중 정사영 했을 때의 분산이 가장 큰 직선
- PC_2의 싱귤러 벡터는 공분산 행렬의 두번째 큰 고유값(λ_2)에 대한 고유벡터



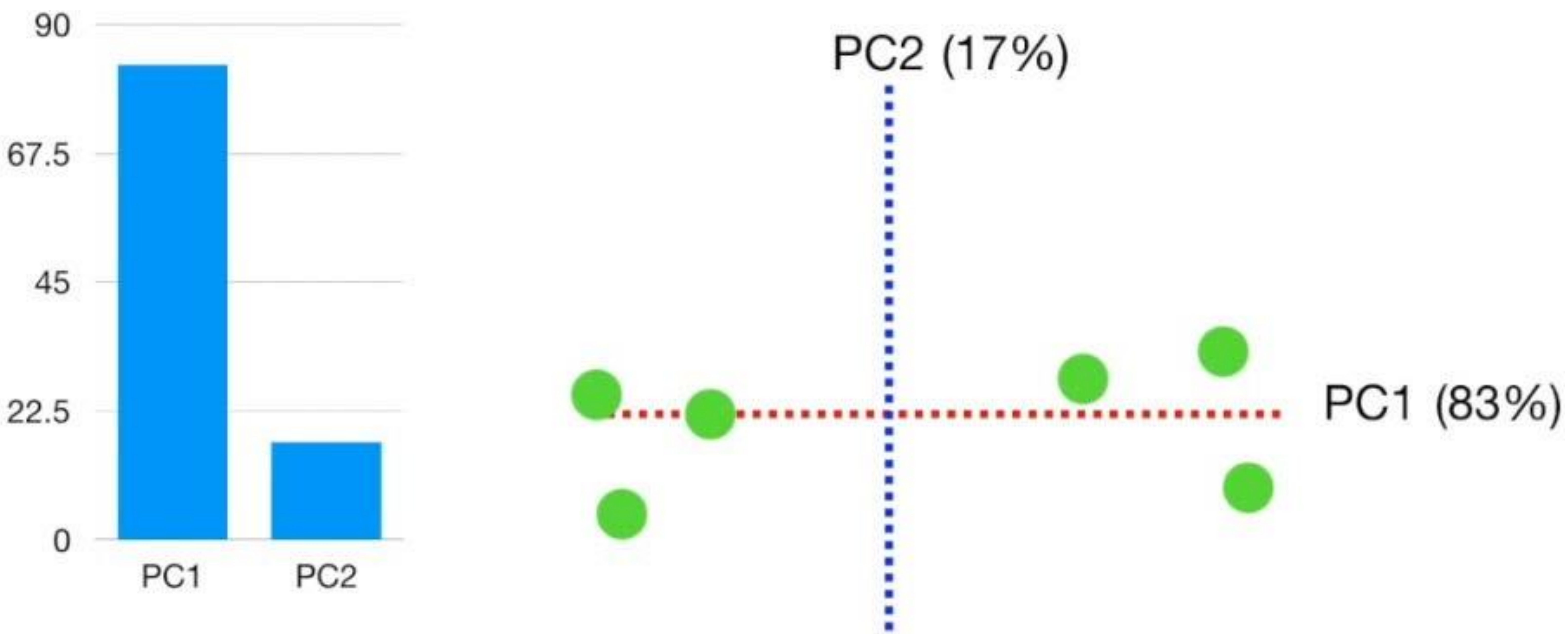
각 PC 별 중요도 계산

- PC_1과 PC_2가 각각 얼마나 중요한지 알아보기 위해 비율을 계산



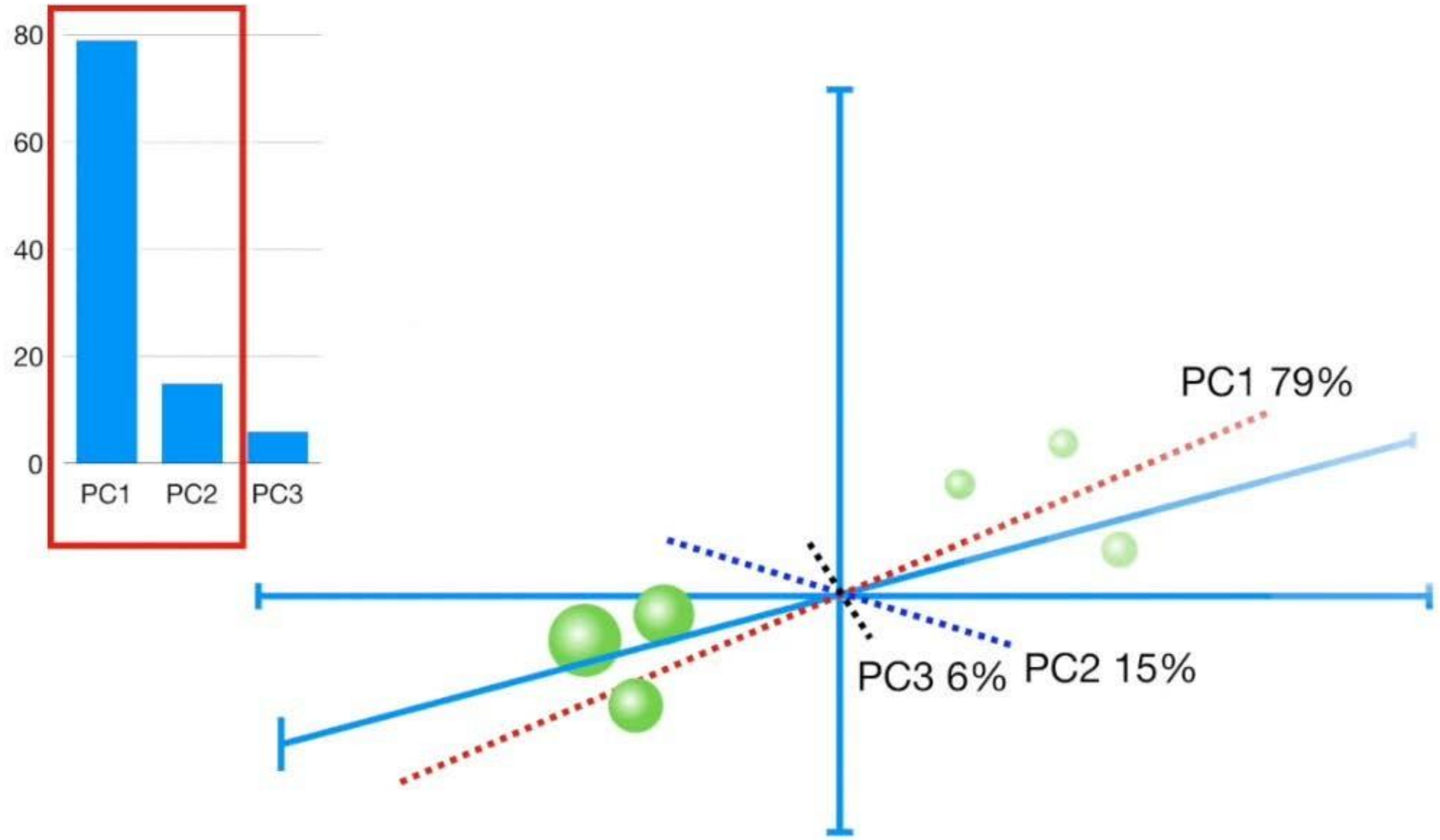
Scree Plot

- 앞에서 계산한 비율을 히스토그램으로 나타낸 것을 Scree Plot이라 함



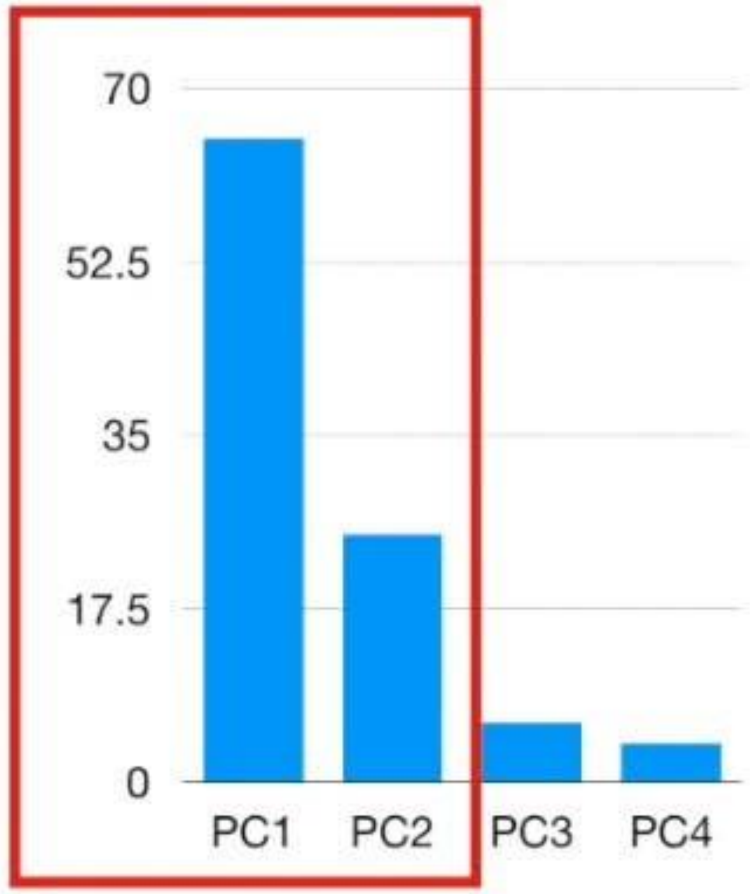
Scree Plot

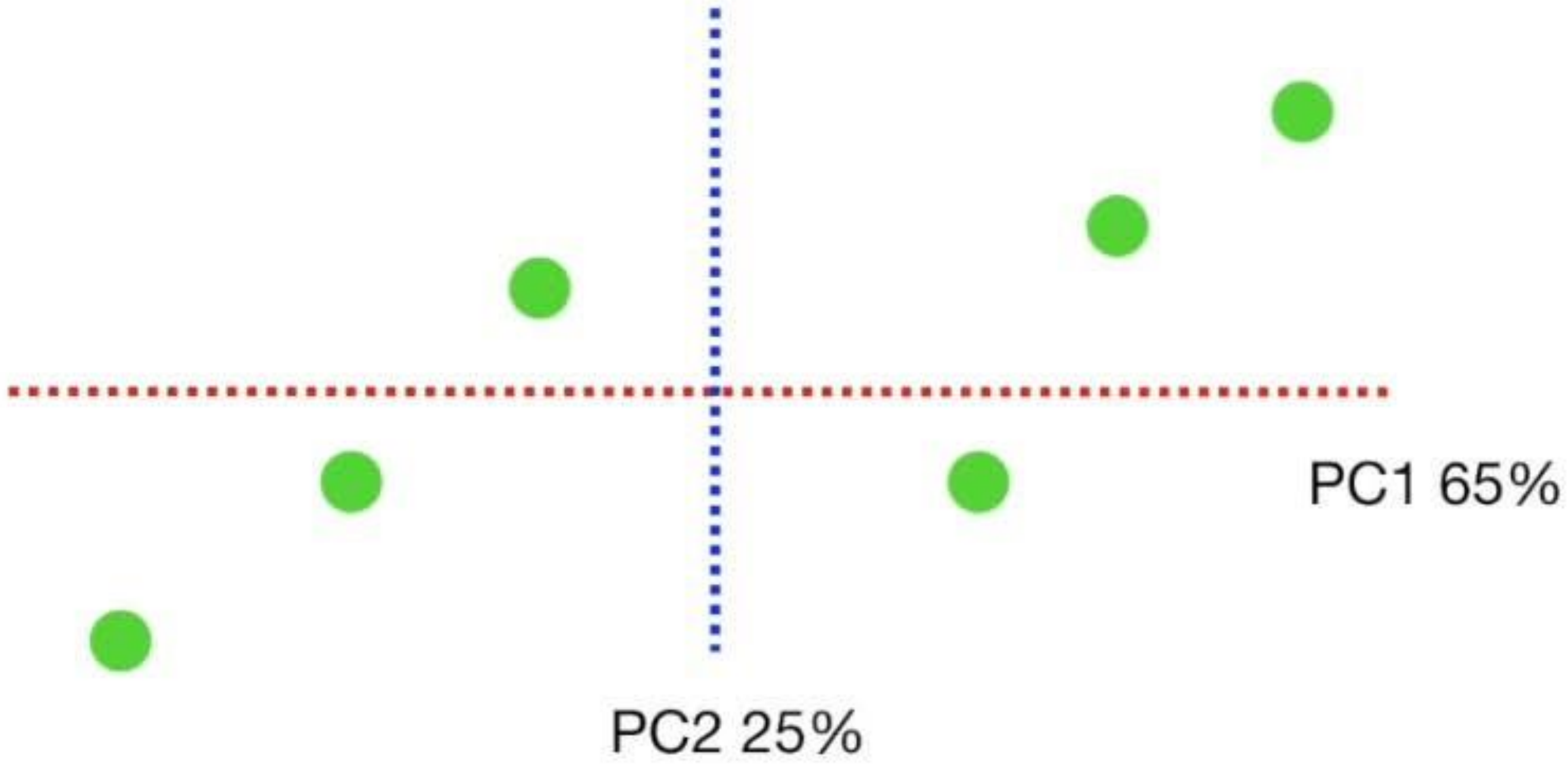
- 피쳐가 3개인 경우 Scree plot에서 기여도를 보고 2개만 선택할 수 있음



Scree Plot

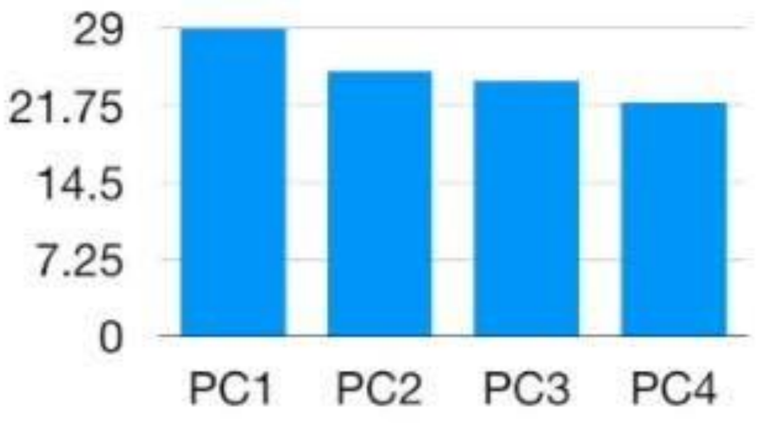
- 피쳐가 4개인 경우 다음과 같이 기여도가 큰 2개를 선택하면 좋음

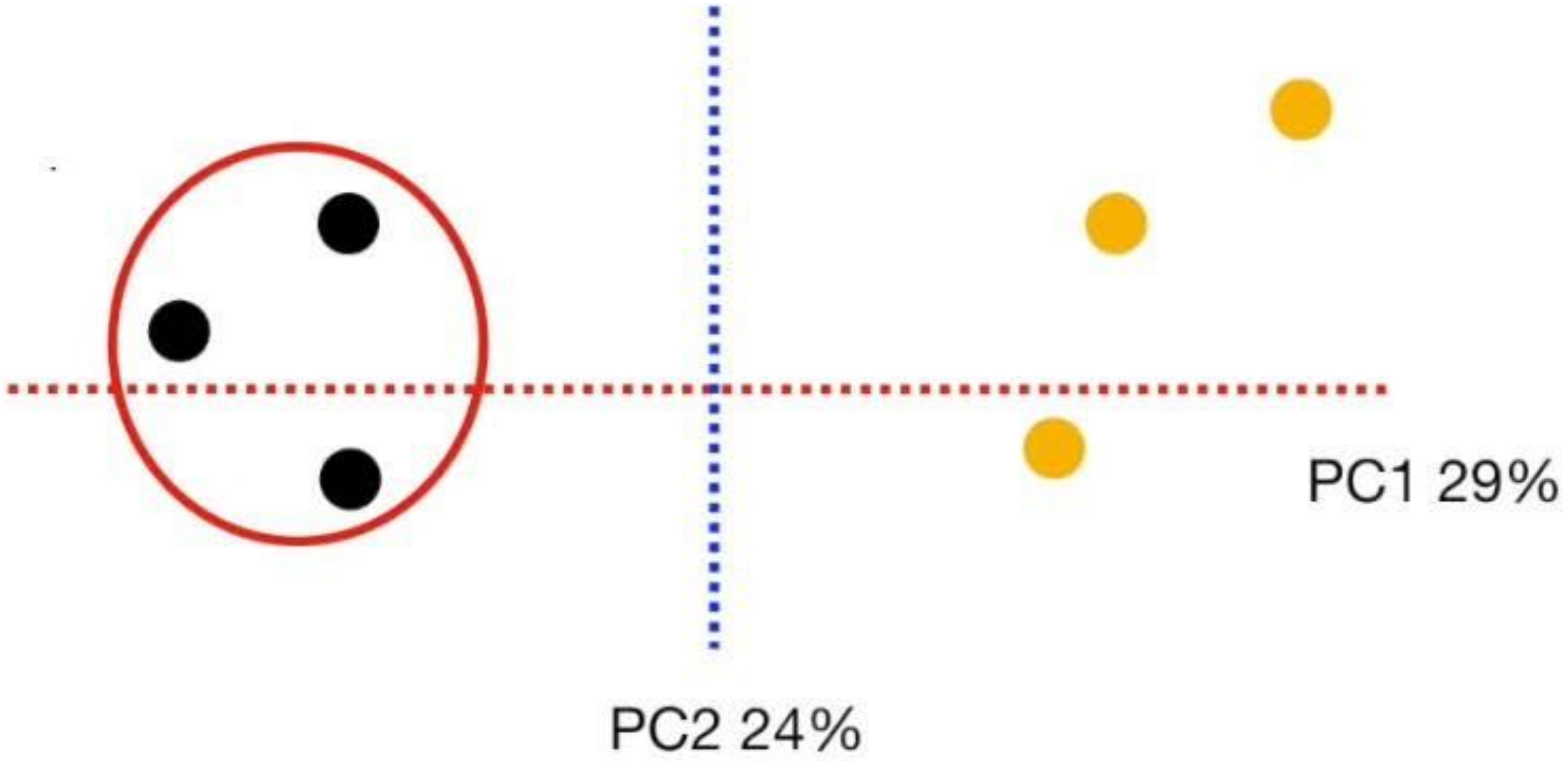




Scree Plot

- 다음 그림과 같이 기여도 차이가 크지 않은 경우에도 2개를 선택해서 clustering을 하는데 사용 할 수 있다.
- 하지만 이런 경우 원래 데이터 복원에는 좋지 않다.

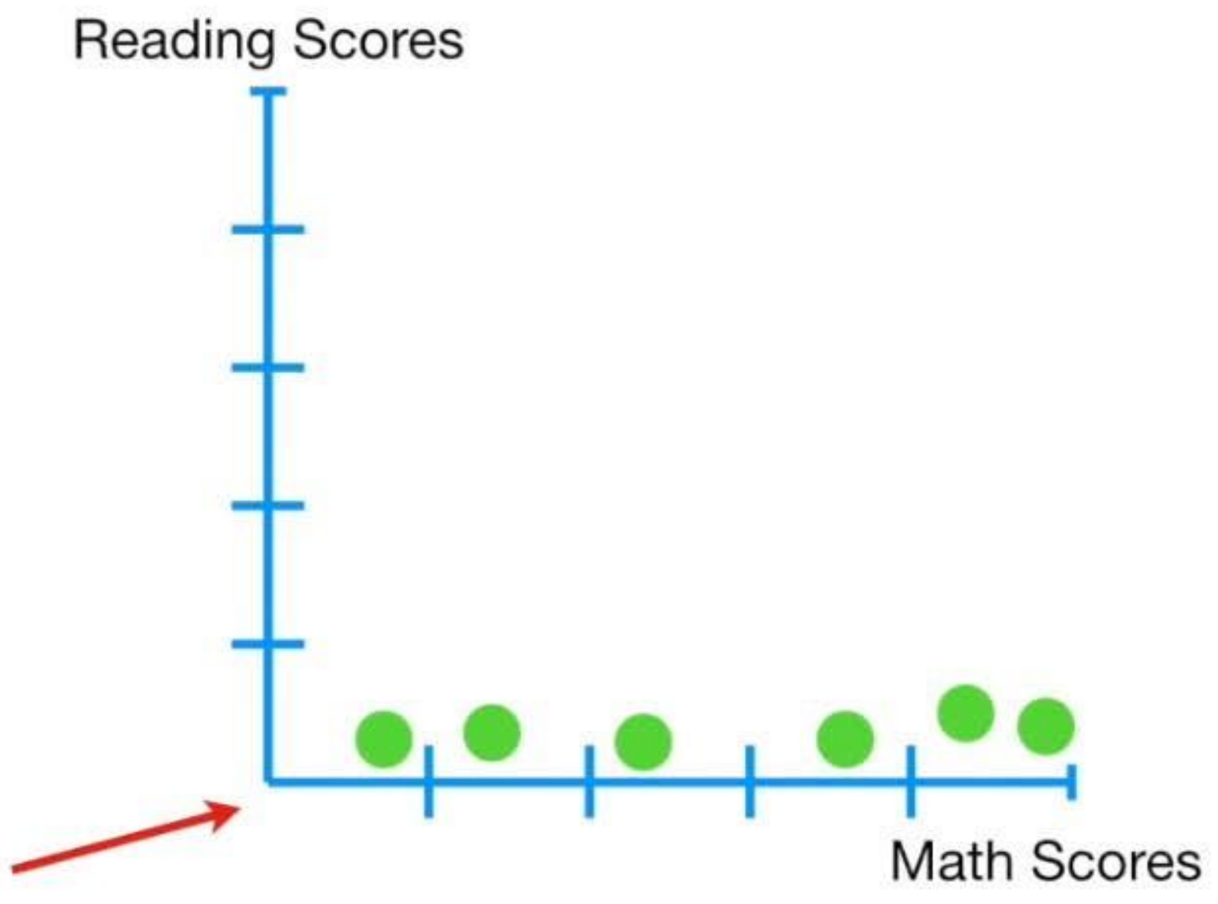




PCA 사용 팁

- 아래의 경우 PCA를 실행하면 수학 점수가 읽기 점수보다 10배 중요한 것으로 나오는데 이는 단지 Scaling이 되어있지 않기 때문임.
- 따라서 각 feature에 대한 Scaling 작업은 중요함.

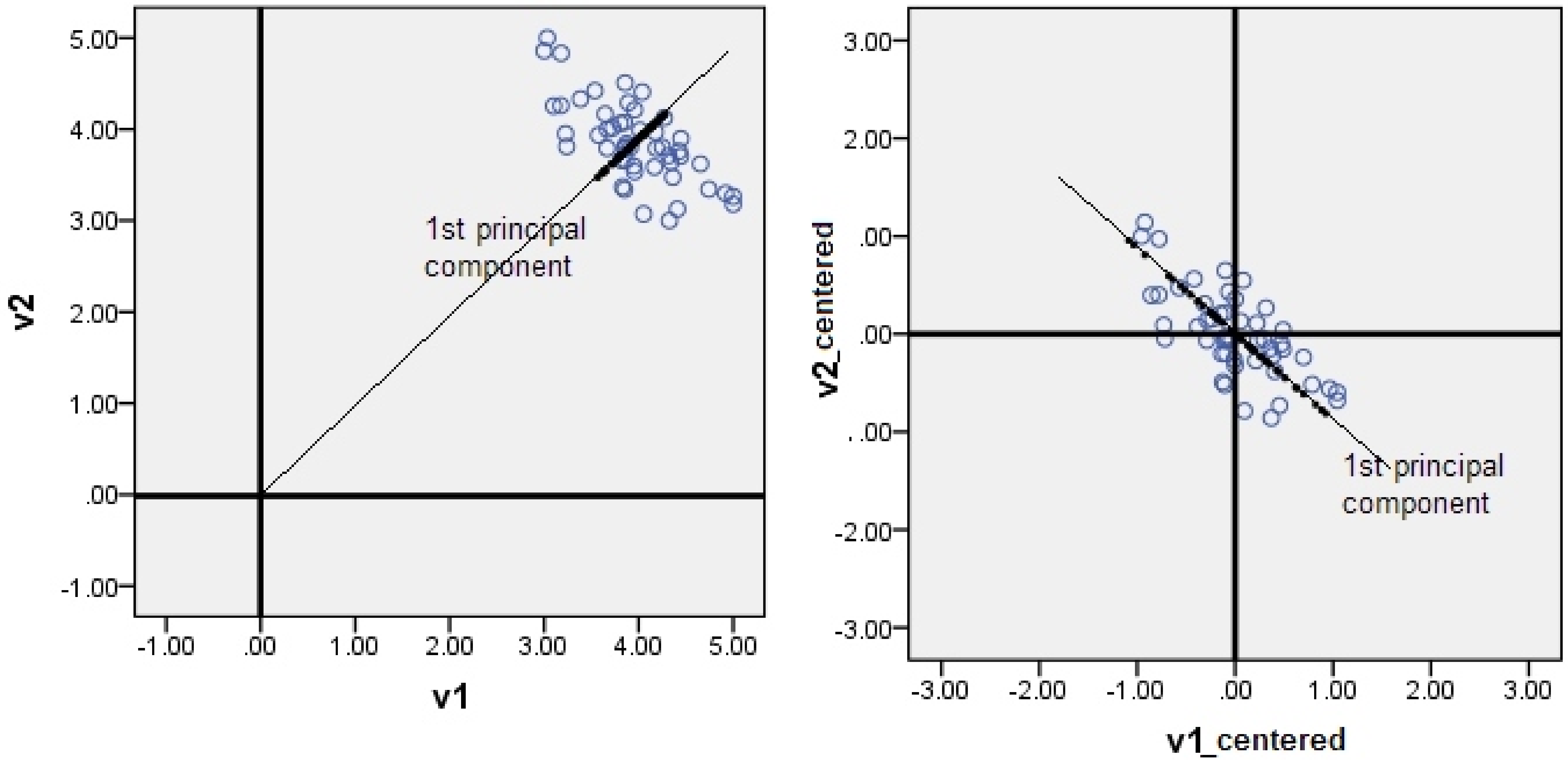
	Student 1	Student 2	Student 3	Student 4	...
Math	95	88	93	75	...
Reading	9	8	10	7	...



/* elice */

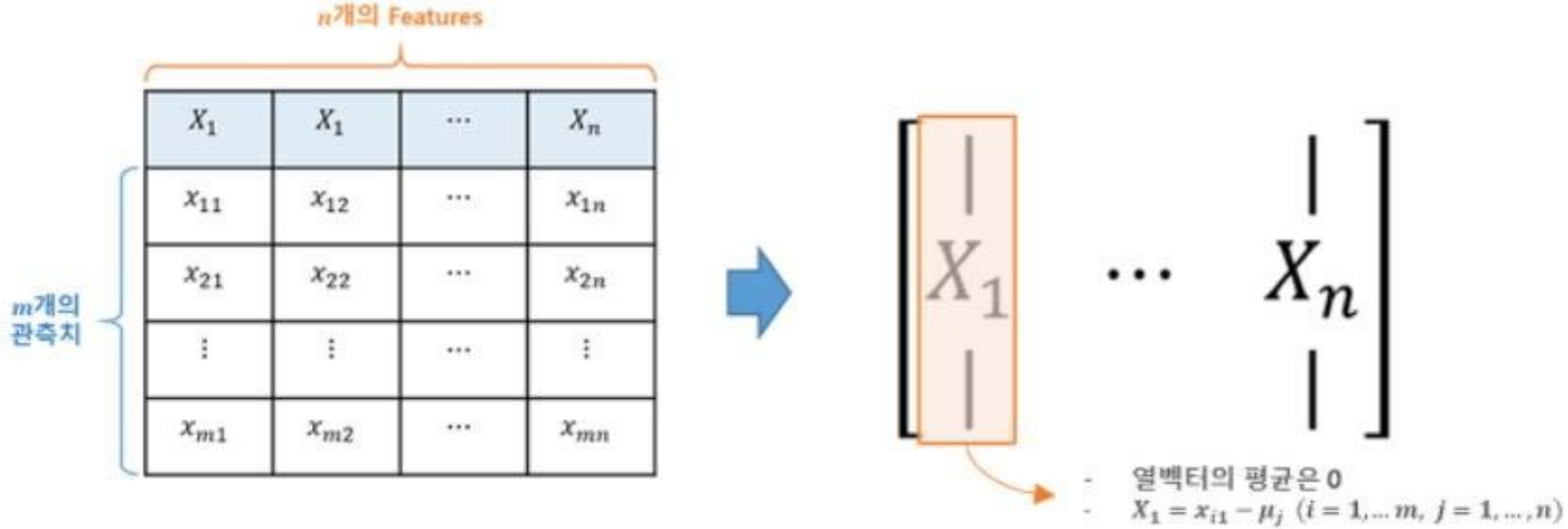
PCA 사용 팁

- PCA를 수행하는 프로그램들 중에 평균을 0으로 바꾸는 작업을 해주지 않는 경우가 있는데, 그럴 경우 원치 않는 결과가 나올 수 있으므로 미리 확인할 필요가 있음.



/* elice */

PCA 원리 설명



$$cov(\mathbf{X}) = \frac{1}{m-1}\mathbf{X}^T\mathbf{X}$$

PCA 원리 설명

PCA의 목적은 원 데이터(original data)의 분산을 최대한 보존하는 축을 찾아 투영(projection)하는 것이다. 예를 들어, 평균 0으로 조정한(편차를 구한) 데이터셋 $\mathbf{X}$를 단위벡터 $\vec{e}$인 임의의 축 P에 투영한다고 했을 때, $\mathbf{X}$의 투영된 결과는 $\mathbf{X}\vec{e}$로 표현할 수 있다. 이때의 분산은 다음과 같이 나타낼 수 있다.

$$
\begin{aligned}
Var\left[\mathbf{X}\vec{e}\right] &= \frac{1}{m-1}\sum_{i=1}^{m}\left[X\vec{e} - E\left(X\vec{e}\right)\right]^2 \\
&= \frac{1}{m-1}\sum_{i=1}^{m}\left[X\vec{e} - E(X)\vec{e}\right]^2, \quad (E(X)=0) \\
&= \frac{1}{m-1}\sum_{i=1}^{m}\left(X\vec{e}\right)^2 \quad = \frac{1}{m-1}\left(\mathbf{X}\vec{e}\right)^T\left(\mathbf{X}\vec{e}\right) \\
&= \frac{1}{m-1}\vec{e}^T\mathbf{X}^T\mathbf{X}\vec{e} \quad = \vec{e}^T\left(\frac{\mathbf{X}^T\mathbf{X}}{m-1}\right)\vec{e} \\
&= \vec{e}^T\mathbf{C}\vec{e}
\end{aligned}
$$

따라서, PCA는 $Var\left[\mathbf{X}\vec{e}\right] = \vec{e}^T\mathbf{C}\vec{e}$를 목적함수로 하는 최대화 문제이며 이때 제약조건은 $\|\vec{e}\|^2 = 1$ 이다.

$$
\begin{aligned}
&\text{maximize} \quad \vec{e}^T\mathbf{C}\vec{e} \\
&\text{s.t.} \quad \|\vec{e}\|^2 = 1
\end{aligned}
$$

PCA 원리 설명

라그랑제 승수법을 이용하여 계산할 수 있다. 위의 식을 라그랑지안 함수 L로 나타내면 다음과 같다.

$$L(\vec{e}, \lambda) = \vec{e}^T \mathbf{C} \vec{e} - \lambda \left(\vec{e}^T \vec{e} - 1 \right)$$

라그랑지안 함수 L을 $\vec{e}$에 대해 편미분 하면 다음과 같다.

$$\begin{aligned} \frac{\partial L}{\partial \vec{e}} &= \left(\mathbf{C} + \mathbf{C}^T\right) \vec{e} - 2\lambda \vec{e} \\ &= 2\mathbf{C}\vec{e} - 2\lambda\vec{e} = 0 \end{aligned}$$

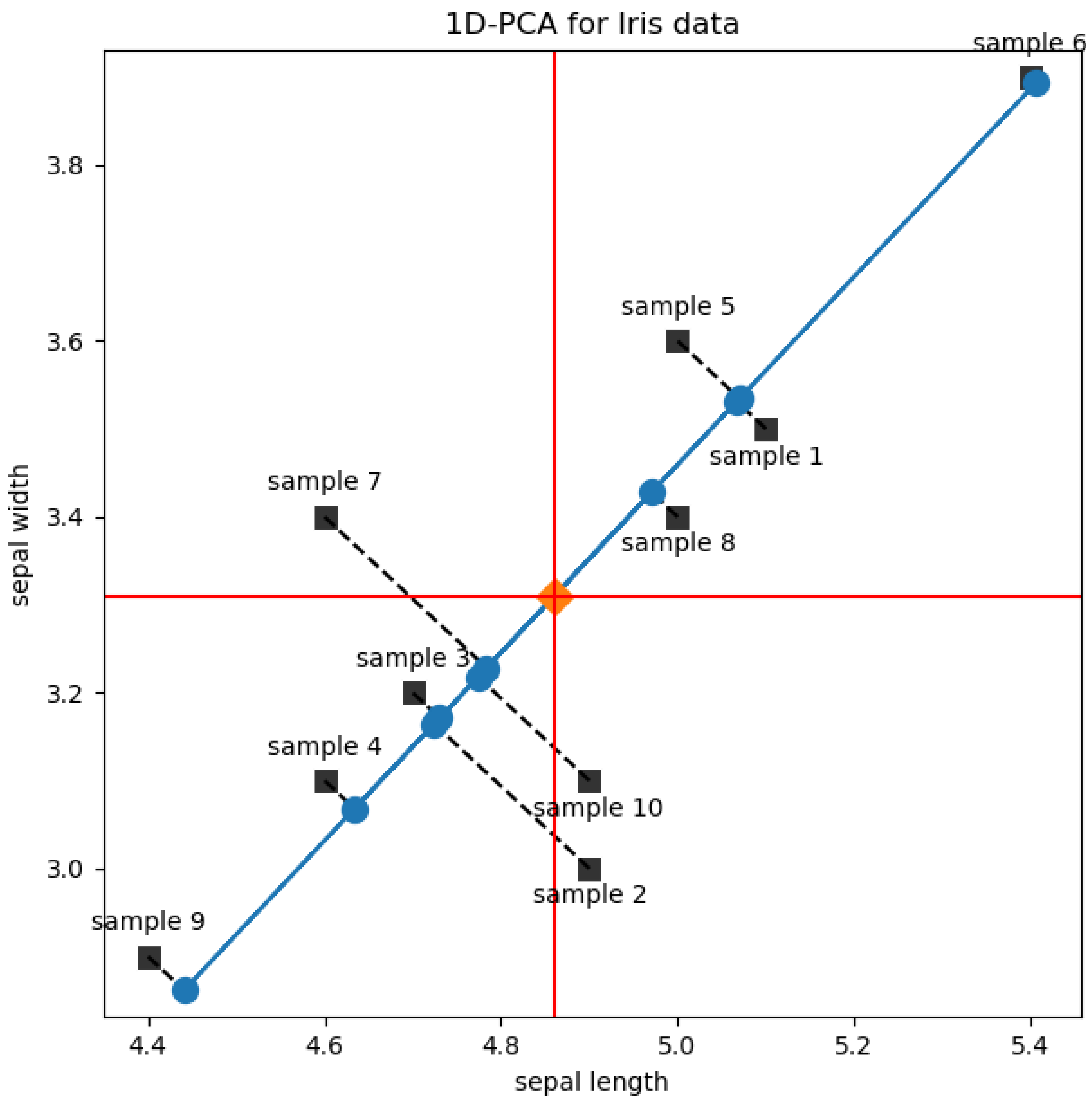
$$\therefore \mathbf{C}\vec{e} = \lambda\vec{e}$$

$$\therefore \mathbf{C} = \vec{e}\lambda\vec{e}^T$$

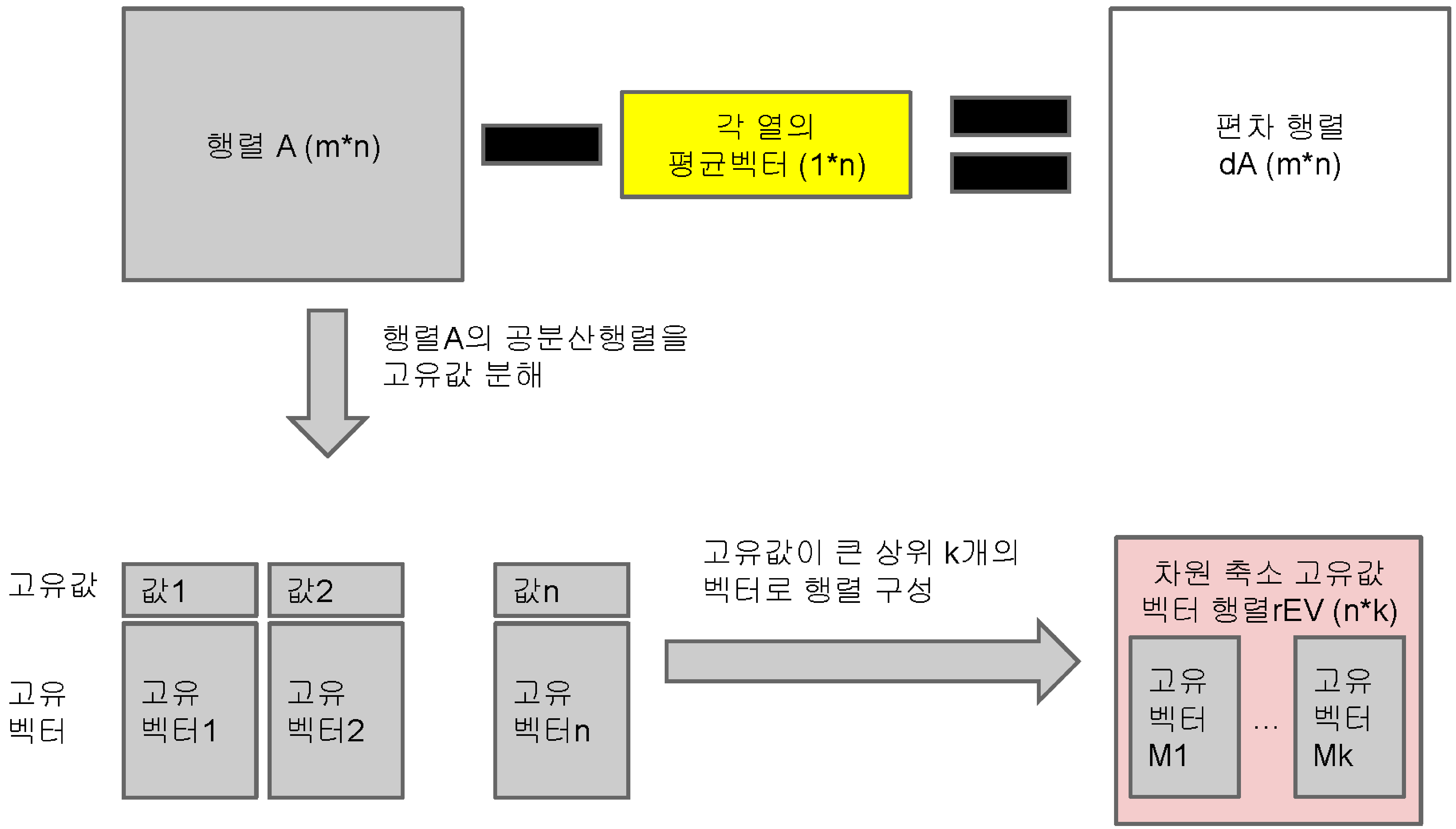
즉, $\mathbf{C}\vec{e} = \lambda\vec{e}$를 만족하는 $\vec{e}$가 바로 분산 $Var[\mathbf{X}\vec{e}]$를 최대화한다.
위의 식에서 $\vec{e}$는 공분산 $\mathbf{C}$의 **고유벡터**(eigenvector)이며, λ는 $\mathbf{C}$의 **고유값**(eigenvalue)이자 eigenvector로 투영했을 때의 **분산**(variance)이다. 이때, 고유벡터의 열벡터를 **주성분**(PC, principal component)이라고 한다. 따라서 고유벡터(eigenvector)에 투영하는 것이 분산이 최대가 된다.

간단한 예제

- iris 데이터에 대하여 PCA 수행한 결과
- 2차원 데이터에 대하여 1차원 축소

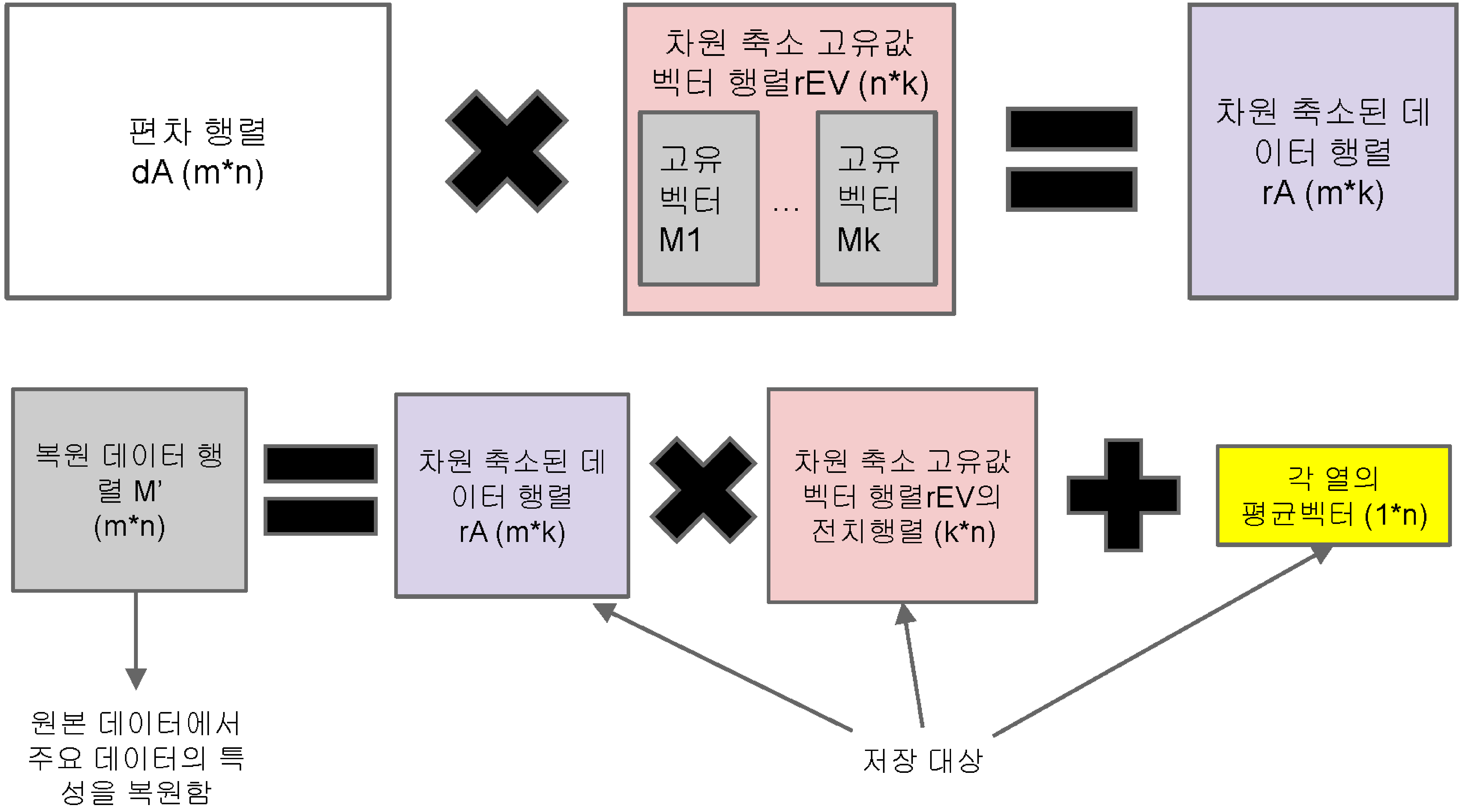


데이터 압축과 복원



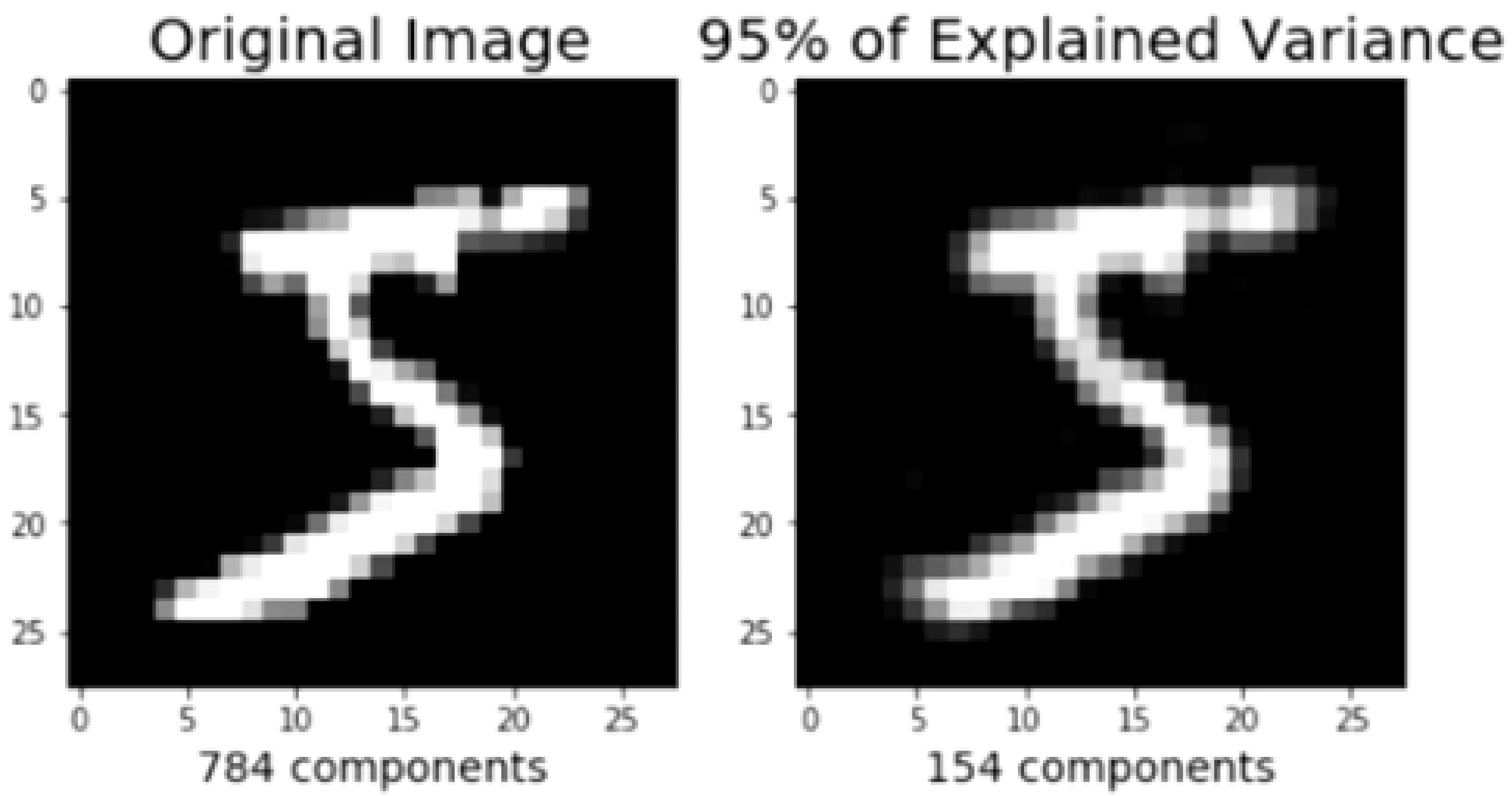
/* elice */

데이터 압축과 복원

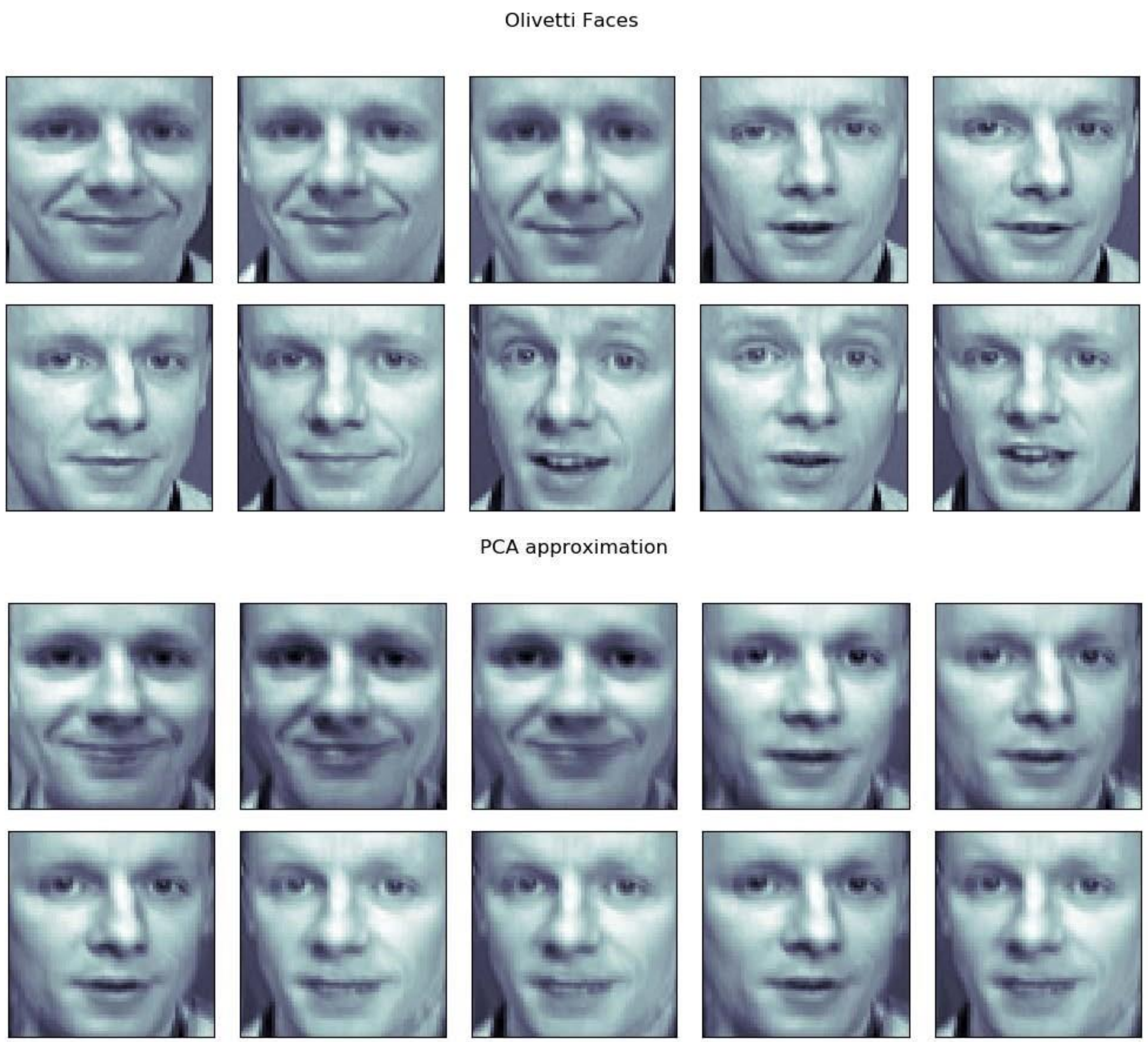

편차 행렬
dA (m*n)
차원 축소 고유값
벡터 행렬rEV (n*k)
고유
벡터
M1
…
고유
벡터
Mk
차원 축소된 데
이터 행렬
rA (m*k)
복원 데이터 행
렬 M'
(m*n)
차원 축소된 데
이터 행렬
rA (m*k)
차원 축소 고유값
벡터 행렬rEV의
전치행렬 (k*n)
각 열의
평균벡터 (1*n)
원본 데이터에서
주요 데이터의 특
성을 복원함
저장 대상

데이터 압축과 복원 예시 (1)

- 28 × 28 = 784개의 feature를 PCA를 사용해 95 분산인 154개의 feature로 줄인 후 다시 복원



데이터 압축과 복원 예시 (2)



03

CUR Decomposition

Confidential all right reserved

CUR Decomposition의 목적

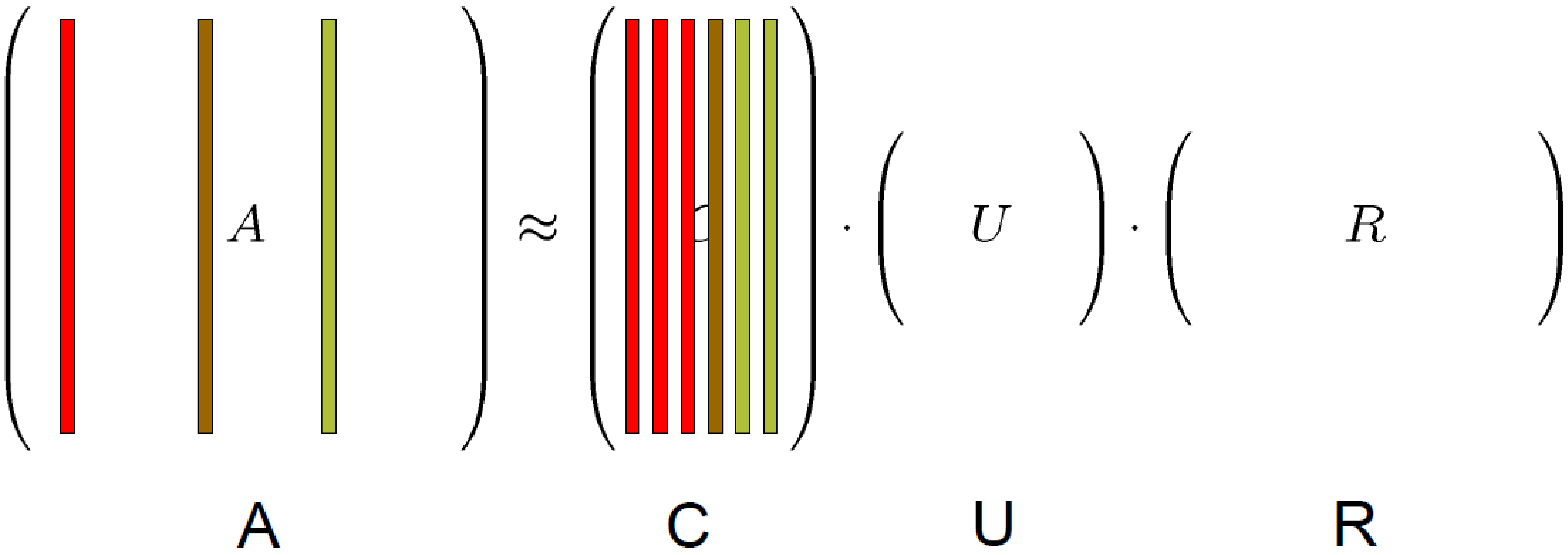
- 데이터 행렬 X를 행렬 C, U, R의 곱으로 표현
- 목표

Make $\|A - C \cdot U \cdot R\|_F$ small

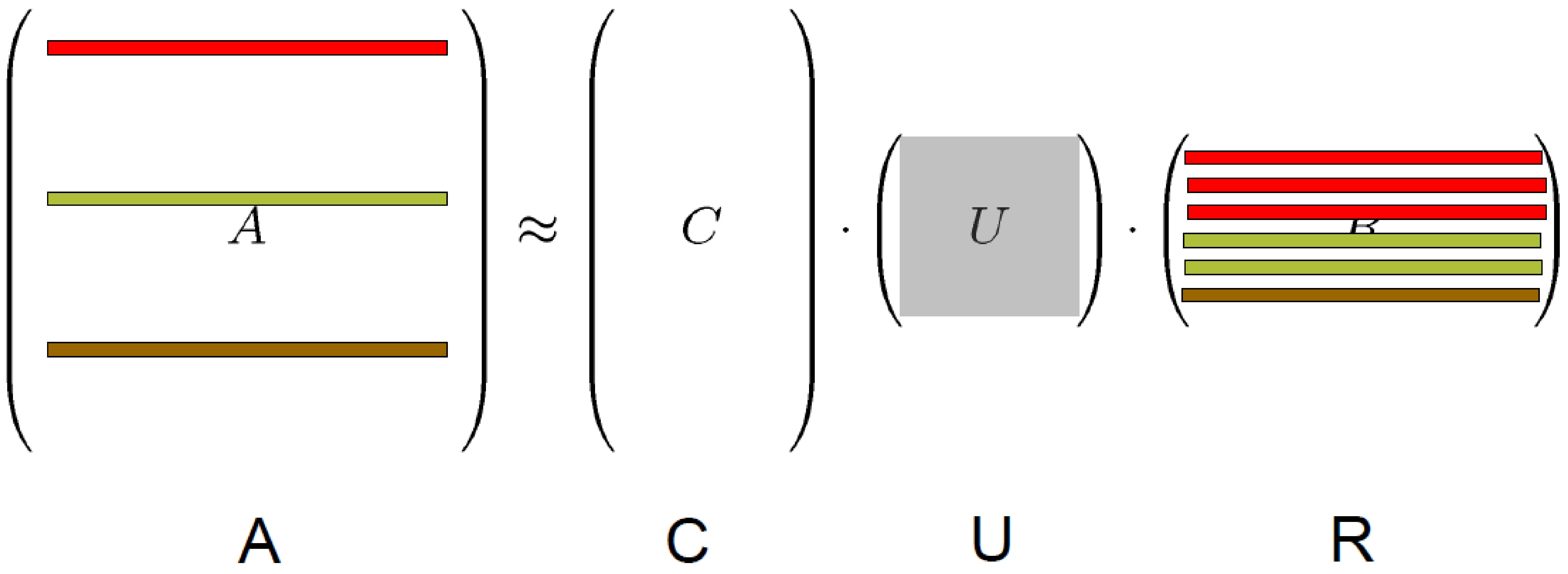
Frobenius norm:

$$\|X\|_F = \sqrt{\sum_{ij} X_{ij}^2}$$

행렬 C의 의미



행렬 R의 의미



Column 샘플링 과정 (row도 유사)

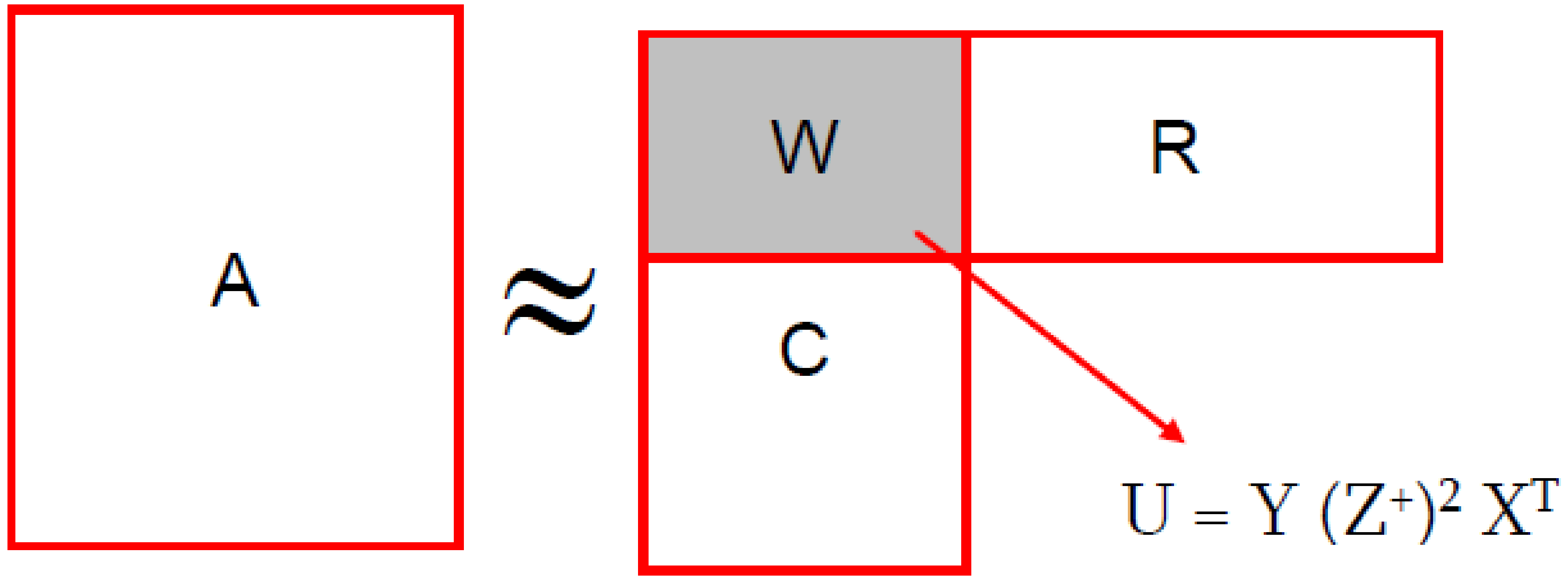
- 랜덤 알고리즘으로 같은 행(혹은 열)이 중복으로 샘플링이 될 수 있음

Input: matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, sample size c
Output: $\mathbf{C}_d \in \mathbb{R}^{m \times c}$

1. for $x = 1 : n$ [column distribution]
2. $P(x) = \sum_i \mathbf{A}(i, x)^2 / \sum_{i,j} \mathbf{A}(i, j)^2$
3. for $i = 1 : c$ [sample columns]
4. Pick $j \in 1 : n$ based on distribution $P(x)$
5. Compute $\mathbf{C}_d(:, i) = \mathbf{A}(:, j) / \sqrt{cP(j)}$

행렬 U 계산 방법

- W를 sampling된 column의 행렬 C와 row의 행렬 R의 intersection이라고 가정
- 이후 W에 대하여 SVD 적용 ($W = XZY^{\mathrm{T}}$)
- $U = Y(Z^{+})^{2}X^{\mathrm{T}}$
- Z^{+} : non-zero singular values의 역수 ($Z^{+}_{ii} = 1/Z_{ii}$)
- Z^{+}는 Z의 pseudo inverse라고 불림



CUR 분해의 good approximation

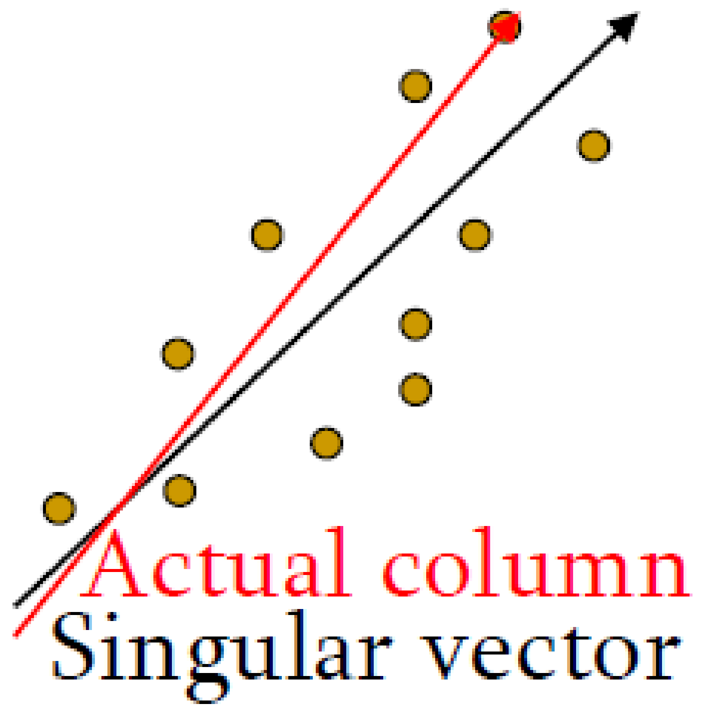
- 만약 sampling할 행과 열의 수를 잘 선택했다면 98%의 확률로 아래 수식을 만족

$$\underbrace{\|A - CUR\|_F}_{\text{CUR error}} \leq (2 + \epsilon) \overbrace{\|A - A_k\|_F}^{\text{SVD error}}$$

- 일반적으로 rank-k approximation을 위해 4k개의 행과 열의 수를 선택

CUR 분해 장단점

- 장점
  - 쉬운 해석 (basis vector가 실제 행과 열로 구성됨)
  - Sparse한 basis (마찬가지 이유)
- 단점
  - Error 최소화를 위한 optimal한 방법은 아님
  - 중복되는 행과 열 문제 (큰 가중치에 대하여 많이 sampling함)
    - 해당 문제를 해결한 CMD 기법이 2007년 나옴



04

t-Stochastic Neighborhood Embedding (tSNE)

Confidential all right reserved

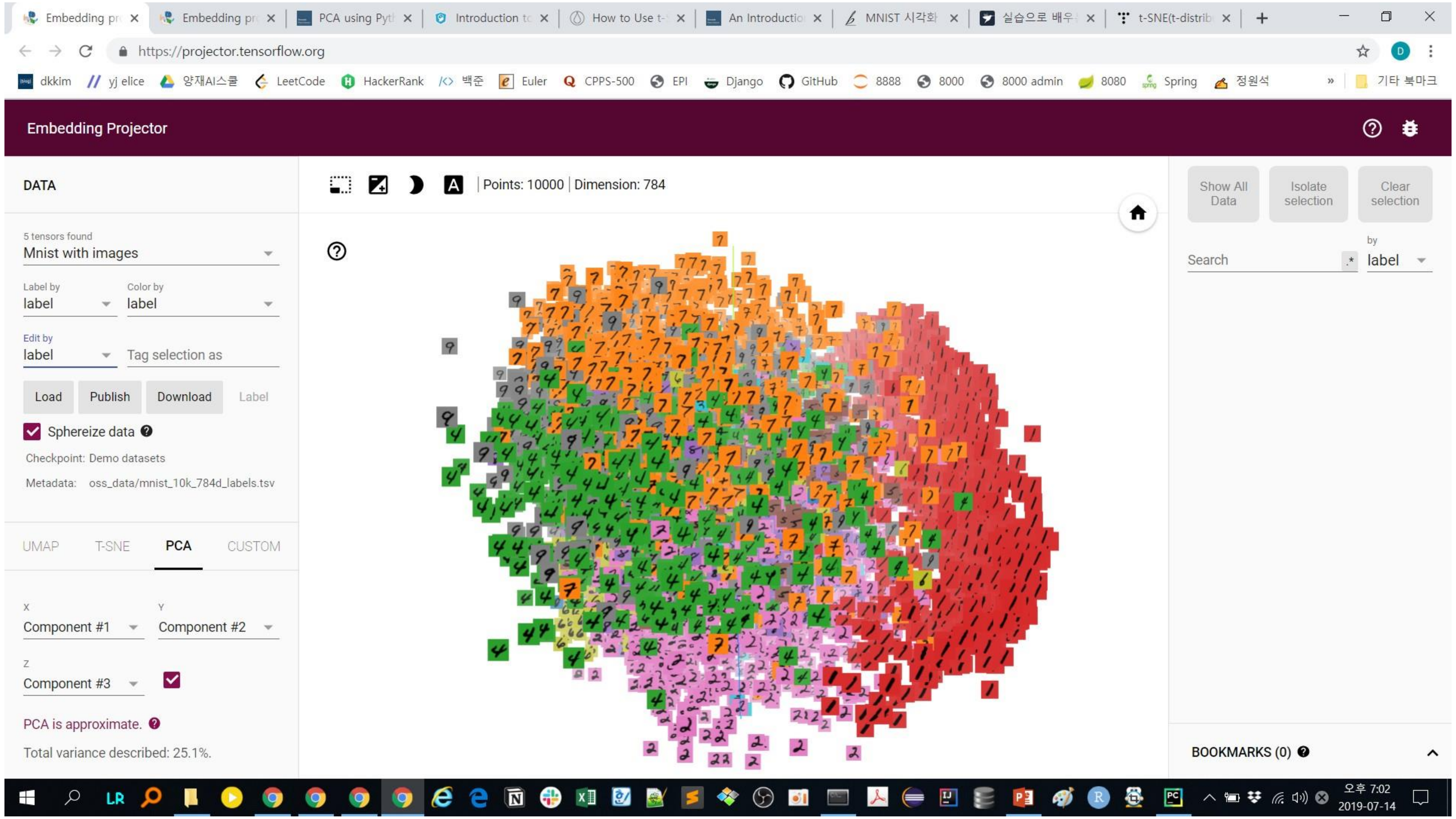
SNE

- 고차원 공간에서 유클리드 거리를 데이터 포인트의 유사성을 표현하는 조건부 확률로 변환하는 방식으로 저차원 공간에 표시
- KL(Kullback–Leibler) divergence 사용

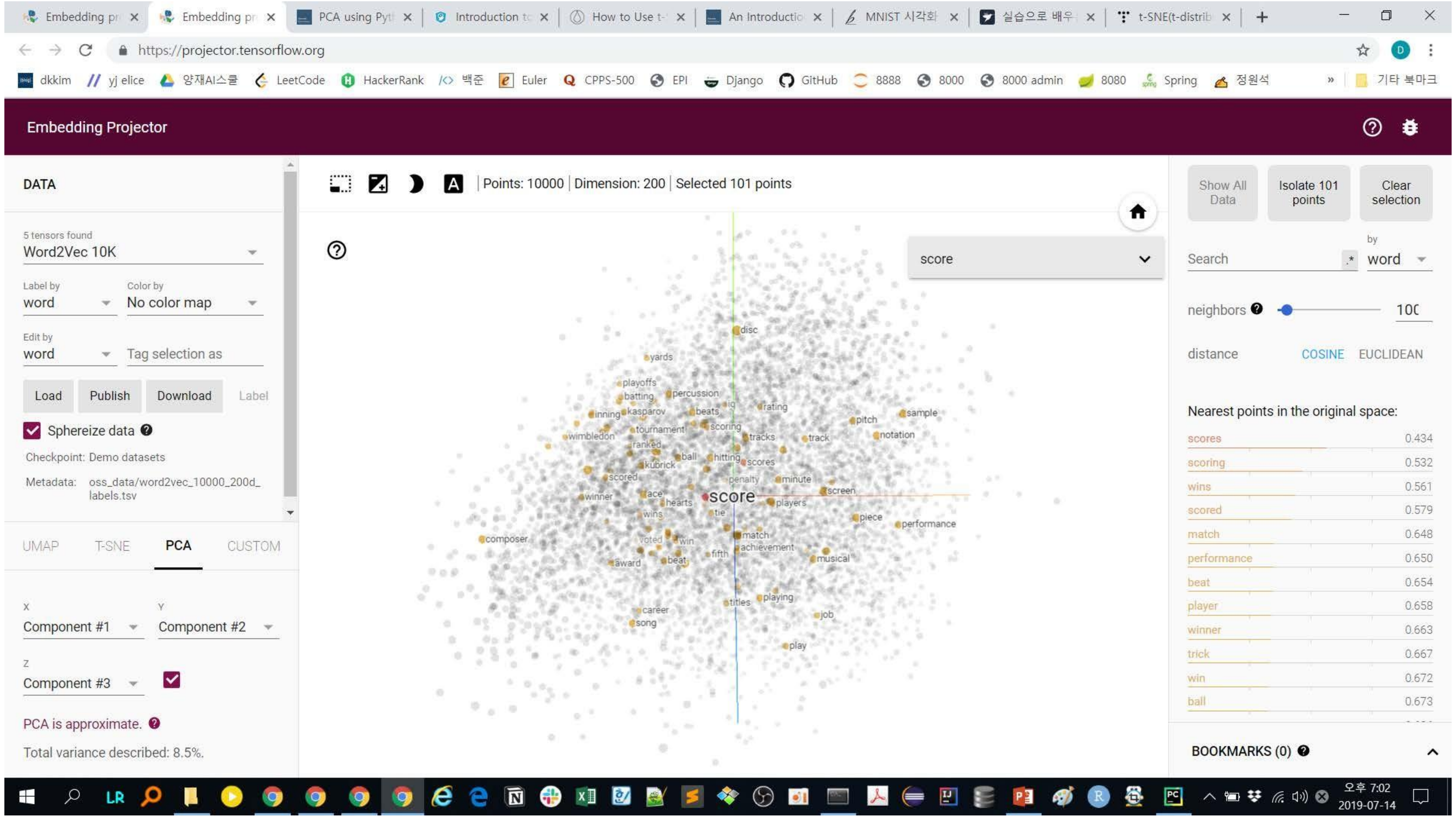
tSNE 등장

- 2008년 Laurens van der Maaten과 Geoffrey Hinton가 기존의 SNE의 단점을 보완하는 tSNE(t-Stochastic Neighbor Embedding) 제안
- 손실함수를 대칭 버전으로 변경
- 정규분포 대신 Student-t 분포 사용

tSNE를 이용한 MNIST 시각화 예시



tSNE를 이용한 Word2Vec 시각화 예시



Credit

/* elice */

코스 매니저

콘텐츠 제작자
정민수

강사
정민수

감수자

디자인

Contact

TEL
070-4633-2015

WEB
https://elice.io

E-MAIL
contact@elice.io