Dimension Reduction

CS534

Why dimension reduction?

- High dimensionality – large number of features
  - E.g., documents represented by thousands of words, millions of bigrams
  - E.g., Images represented by thousands of pixels
- Redundant and irrelevant features (e.g., not all words are relevant for classifying/clustering documents)
- Difficult to interpret and visualize
- Curse of dimensionality
  - Distances to nearest and furthest neighbors will become similar as the dimension goes higher

고차원 data에 대해서는 일캐 쓸데없는 feature들이 너무 많다!

cause

Extract Latent Linear Features

- Linearly project *n*-d data onto a *k*-d space
  - e.g., project space of 10^4 words into 3-dimensions
- There are infinitely many k-d subspaces that we can project the data into, which one should we choose
- This depends on the task at hand
  - If supervised learning: maximize the separation among classes, i.e., Linear discriminant analysis (LDA)
  - If unsupervised, we may wish to retain as much data variance as possible, i.e., principal component analysis (PCA)

LDA: linear discriminant analysis

- Also named Fisher Discriminant Analysis
- It can be viewed as
  - *a dimension reduction* method

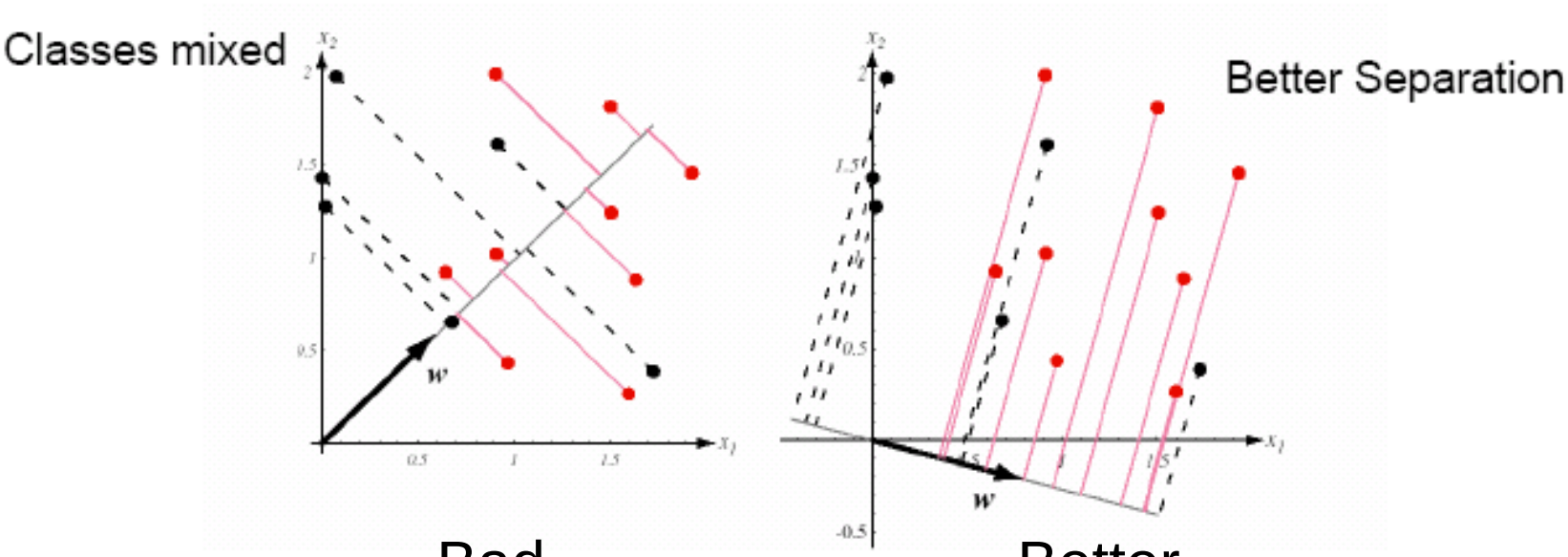
  or
  - a generative classifier p(x|y): Gaussian with distinct $\boldsymbol{\mu}$ for each class but a shared Σ (covariance matrix)
- We will look at its **dimension reduction** interpretation and derive it that way

Intuition

projection specifies vector w.

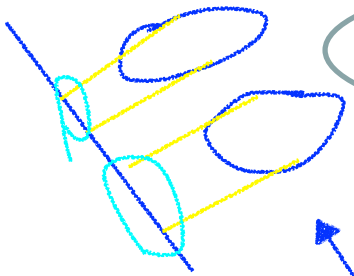
- Find a projection direction to maximize the separation between classes

before projection : X
After projection : W^T*X <- projected coordinate



Objectives of LDA

- One way to measure separation is to look at the class means



$$\mathbf{m}_1 = \frac{1}{N_1}\sum_{\mathbf{x}\in c_1}\mathbf{x} \qquad \mathbf{m}_2 = \frac{1}{N_2}\sum_{\mathbf{x}\in c_2}\mathbf{x}$$

Original means

$$m'_1 = \frac{1}{N_1}\sum_{\mathbf{x}\in c_1}\mathbf{w}^T\mathbf{x} \qquad m'_2 = \frac{1}{N_2}\sum_{\mathbf{x}\in c_2}\mathbf{w}^T\mathbf{x}$$

Projected means

W가 이 방향이믄,
center에서는 멀더라도
두 클러스터가 떨어지게되자낭!!

A possible goal: find the projection that maximize

$$\left|m'_1 - m'_2\right|^2 = \left|\mathbf{w}^T\mathbf{m}_1 - \mathbf{w}^T\mathbf{m}_2\right|^2$$

=|W(m1-m2)|^2

w should be the exactly same direction
with (m1-m2) to have maximum.
Is this good? No.
This obj is not sufficient.

subject to $|\mathbf{w}|^2 = 1$?

왜냠, w에 대한 constrain이 없으면
걍 W를 크게하면 크게할 수록 위 obj가 maximized되자낭

Objectives of LDA

- Maximizing the mean separation is insufficient
- We also want the data points from the same class to be as close as possible
- This can be measured by the within-class (or interclass) ***scatter*** *(i.e., variance within the class)*

$$s_i^2 = \sum_{x \in c_i} (\mathbf{w}^T \mathbf{x} - m'_i)^2$$

Total within-class scatter for projected class i, where m'_i is the mean of class i after projection

$$s_1^2 + s_2^2$$

Total within-class scatter considering both classes

Combining the two sides

- There are a number of different ways to combine these two sides of the objective
- LDA seeks to optimize the following objective:

$$\underset{\mathrm{w}}{\operatorname{argmax}} \frac{|\mathbf{m}_1' - \mathbf{m}_2'|^2}{S_1^2 + S_2^2}$$

$$|m_1' - m_2'|^2 = |w^T m_1 - w^T m_2|^2$$

$$= w^T (m_1 - m_2)(m_1 - m_2)^T w \quad \boxed{= w^T \Sigma_B w}$$

$$s_1^2 + s_2^2 = w^T (\Sigma_1 + \Sigma_2) w$$

$$\boxed{= w^T \Sigma_w w}$$

$$s_1^2 = \sum_{x \in C_1} (w^T x - w^T m_1)^2 = \sum_x w^T (x - m_1)(x - m_1)^T w$$

$$= w^T \left(\sum_x (x - m_1)(x - m_1)^T \right) w = w^T \Sigma_1 w$$

The LDA Objective

find w maximizing this objective

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^T \Sigma_B \boldsymbol{w}}{\boldsymbol{w}^T \Sigma_w \boldsymbol{w}}$$

$\Sigma_B = (m_1 - m_2)(m_1 - m_2)^T$
the between-class scatter matrix

앱실론 w 는 within이고

$\Sigma_w = \Sigma_1 + \Sigma_2$
the total within-class scatter matrix, where

$$\Sigma_i = \sum_{x \in C_i} (x - m_i)(x - m_i)^T$$

- The above objective is known as generalized Reyleigh quotient, and it's easy to show a w that maximizes $J(w)$ must satisfy $\Sigma_B w = \lambda \Sigma_w w$

- Noticing that $\Sigma_B w = (m_1 - m_2)(m_1 - m_2)^T w$ always take the direction of $m_1 - m_2$

vector 1 $(m_1 - m_2)$ vector 2 $(m_1 - m_2)^T$ vector 3 w

Scalar

시그마B W 는 vector 1이 scalar의 direction으로 됨

- Ignoring the scalars, this leads to:

$$(m_1 - m_2) = \Sigma_w w$$

$$w = \Sigma_w^{-1}(m_1 - m_2)$$

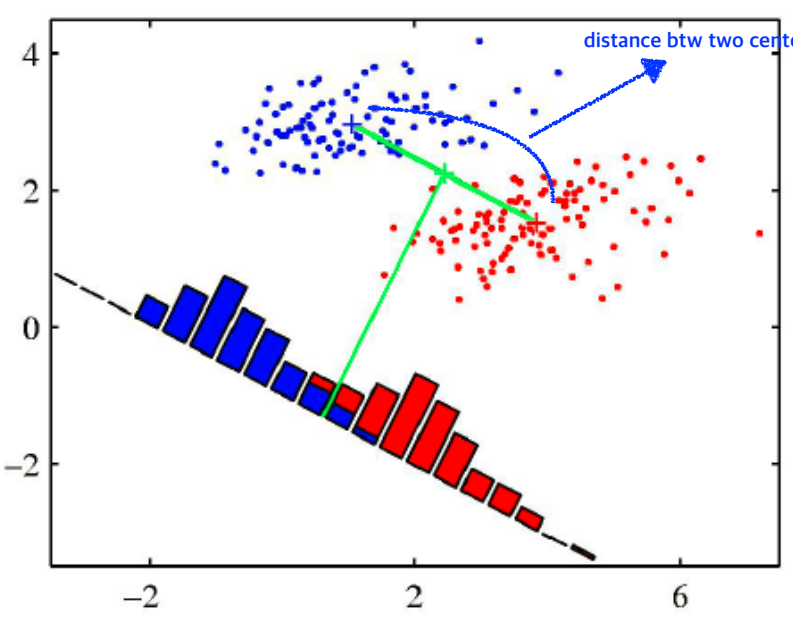
LDA for two classes

only need one decision boundary
그래서 need single w

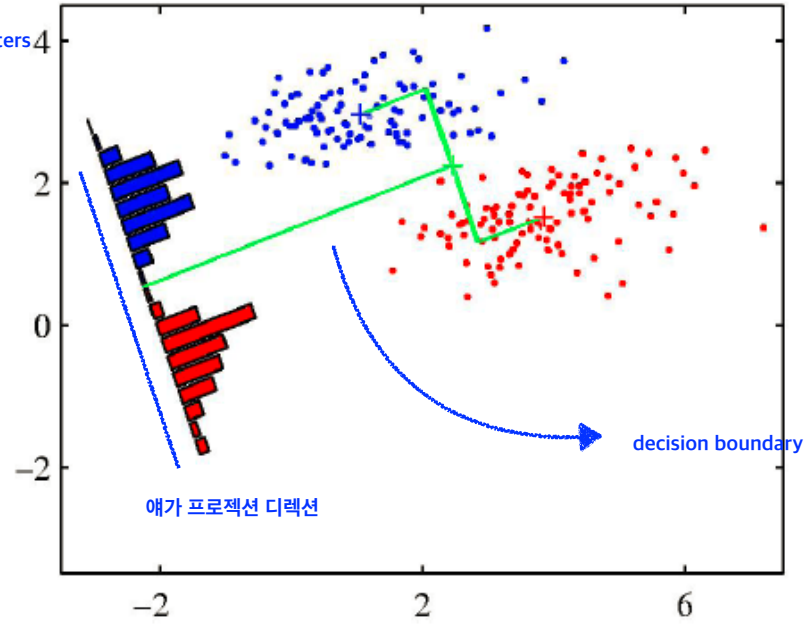
$$w = \Sigma_w^{-1}(m_1 - m_2)$$

Maximize the distance between projected mean

Maximize $\dfrac{\text{between scatter}}{\text{within scatter}}$



$w = m_1 - m_2$



$w = \Sigma_w^{-1}(m_1 - m_2)$

LDA for Multi-Classes

- Many variants exist. This is one of the commonly used ones:

$$J(w) = \frac{w^T \Sigma_B w}{w^T \Sigma_w w}$$

- Objective remains the same, with slightly different definition for between-class scatter:

$$\Sigma_B = \frac{1}{k} \sum_{i=1}^{k} (m_i - m)(m_i - m)^T$$

앱실론 B

mean of each class - overall mean 을 제곱해서 sum up 한거

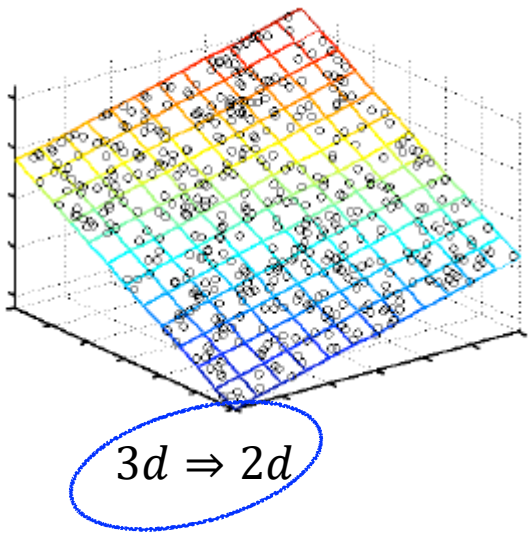
m is the overall mean

- Solution: k-1 eigenvectors of $S_w^{-1} S_B$

S가 아니라 앱실론

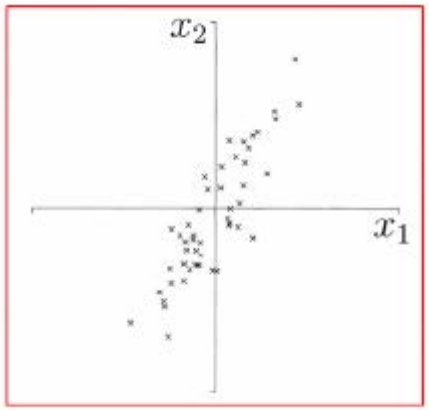
Unsupervised Dimension Reduction

- Consider data without class labels
- Try to find a more compact representation of the data



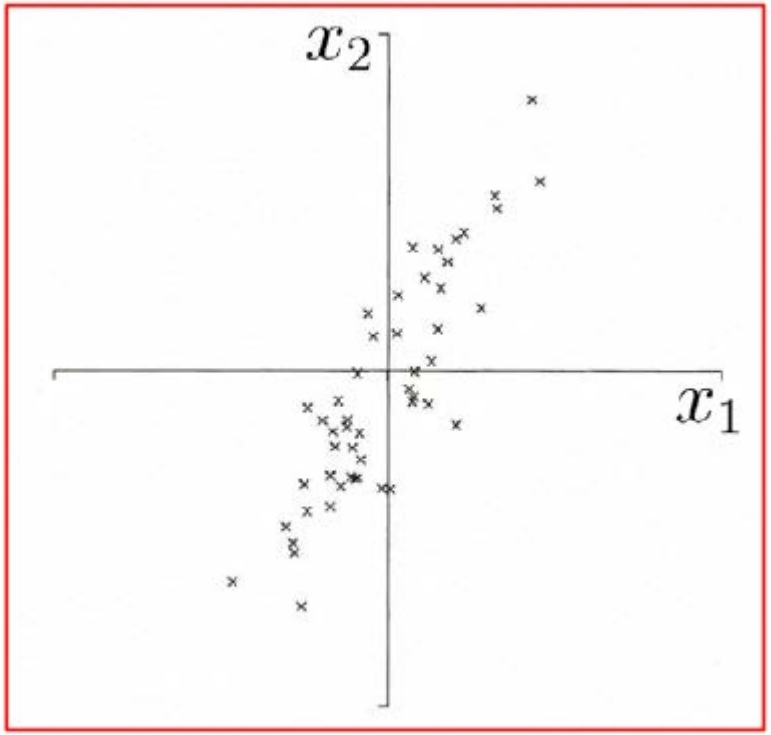
- Assume that the high dimensional data actually resides in a inherent low-dimensional space
- Additional dimensions are just random noise or redundancy from same measurement
- Goal is to recover these inherent dimensions and discard noise dimensions

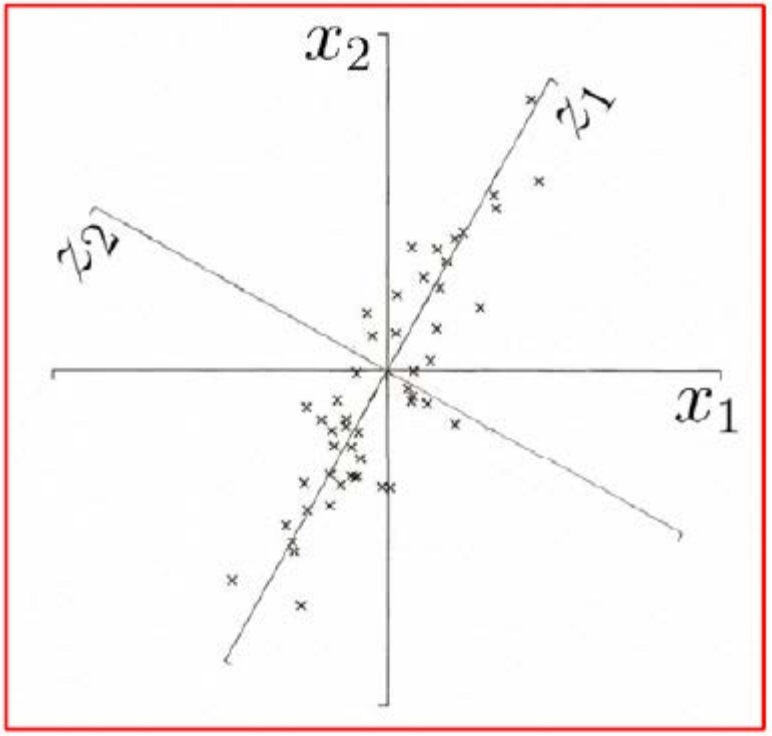
Geometric picture of principal components (PCs)



Goal: to account for the variation in the data in as few dimensions as possible

Geometric picture of principal components (PCs)





second direction should be 수직의 to the first direction
근데 모든 direction들이 다 서로 수직이어야한대

- The 1st PC is the projection direction that maximizes the variance of the projected data
- The 2nd PC is the projection direction that is orthogonal to the 1st PC and maximizes the variance …

PCA: variance maximization

- Given n data points: $x_1, \cdots, x_n$ 이때 x와 v의 dimension은 같다
- Consider a linear projection specified by v
- The projection of x onto v is $z = v^T x$
- The variance of the projected data is $var(z) = var(v^T x) = v^T Cov(x) v = v^T \Sigma v$ (covariance of x)
- The 1st PC maximizes the variance $v^T \Sigma v$ subject to the constraint $v^T v = 1$, where (v is a projection vector, the norm of the projection vector = 1)

$$\Sigma = Cov(x) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})(x_i - \bar{x})^T$$

Maximizing Variance After Projection

- Goal: $\max v^T \Sigma v$, s.t. $v^T v = 1$
- Lagrange:

$$L = v^T \Sigma v - \lambda(v^T v - 1)$$

$$\frac{\partial L}{\partial v} = 0 \Rightarrow \Sigma v = \lambda v$$

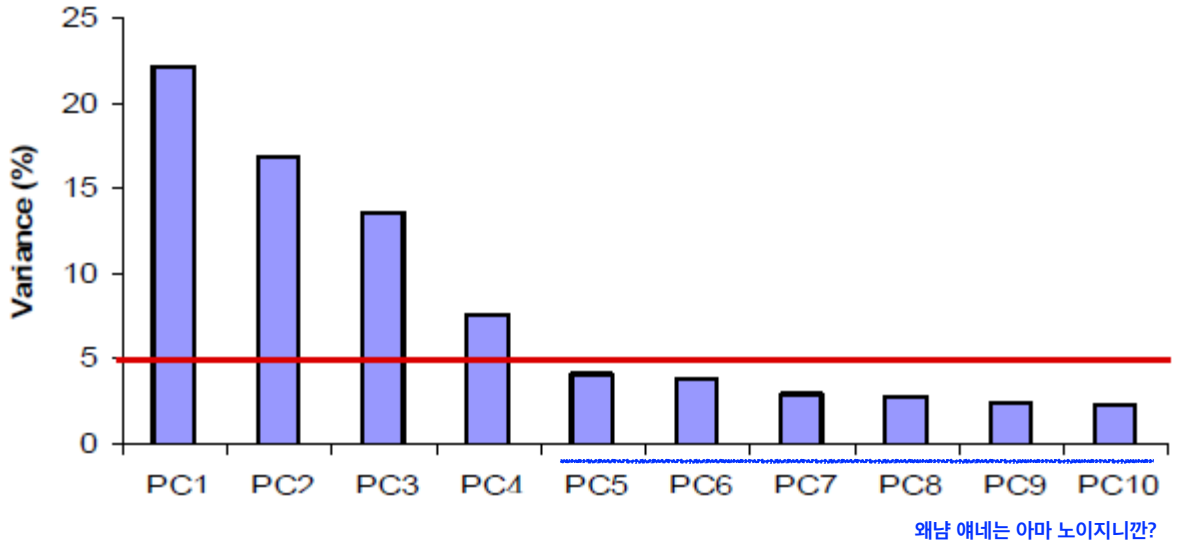
- Thus v is the eigen-vector of Σ with eigen-value λ
- Sample variance of the projected data:

$$v^T \Sigma v = \lambda \underbrace{v^T v}_{=1} = \lambda$$

- The eigen-values λ = the amount of variance captured by each eigen-vector
- Sort all eigen vectors by λ in decreasing order:
  - 1st PC = The first (largest) eigen-vector, the projected variance = λ_1
  - 2nd PC = the second eigen-vector, the projected variance = λ_2
  - …

Dimension Reduction Using PCA

- Calculate the covariance matrix of the data Σ
- Calculate the eigen-vectors/eigen-values of Σ
- Rank the eigen-values in decreasing order
- Select a fixed number of eigen-vectors, or just enough to retain a fixed percentage of the variance, (e.g., 75%, the smallest d such that $\frac{\sum_{i=1}^{d} \lambda_i}{\sum_i \lambda_i} \geq 75\%$)



You might loose some info. But if the eigen-values are small, not much is lost.

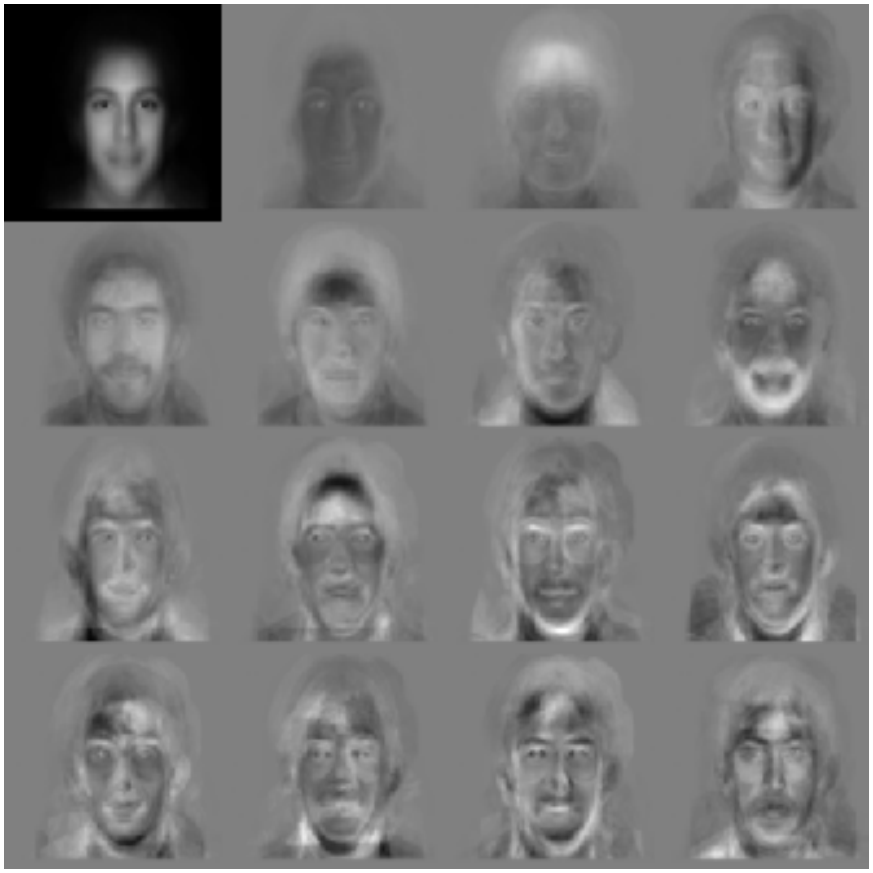
Example: Face Recognition

- A small image of size 256 x 128 is described by n = 256x128 = 32768 dimensions
- Each face image lies somewhere in this high-dimensional space
- Images of faces are generally similar in overall configuration, thus
  - They cannot be randomly distributed in this space
  - We should be able to describe them in a much low-dimensional space

PCA for Face Images: Eigenfaces

aligne안되어있으면
막 전신사진이랑 여권사진이랑 비교를 해야되면
눈이 어디있는지 그 corresponding이 안돼 to compute our objective

- Database of 128 carefully-aligned faces.
- Here are the mean and the first 15 eigenvectors.
- Each eigenvector can be shown as an image
- These images are face-like, thus called eigenface

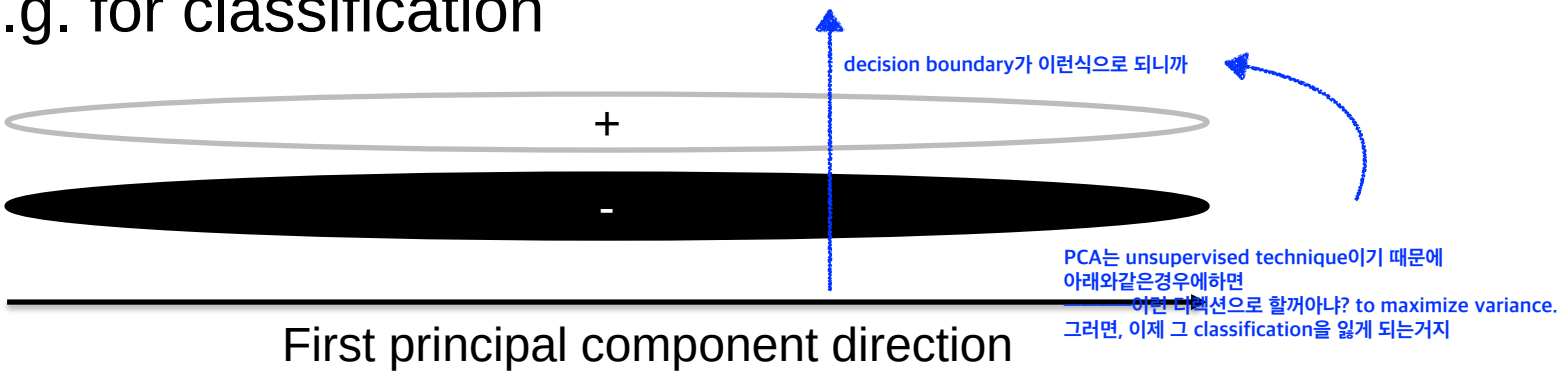


Face Recognition in Eigenface space
(Turk and Pentland 1991)

- Nearest Neighbor classifier in the eigenface space
- Training set always contains 16 face images of 16 people, all taken under the same conditions of lighting, head orientation, and image size
- Accuracy:
  - variation in lighting: 96% 왜냠 조명은 걍 noise니까
  - variation in orientation: 85%
  - variation in image size: 64% alignment가 망가지믄 일캐 정확도가 낮아져

PCA: A Useful Preprocessing Step

- Helps to reduce the computational complexity
- Helps supervised learning
  - Reduced dimension ⇒ simpler hypothesis space
  - Reduced dimension ⇒ less over-fitting
- PCA can also be seen as noise reduction
- May loose important information when the small variance directions contain useful information:
  - E.g. for classification

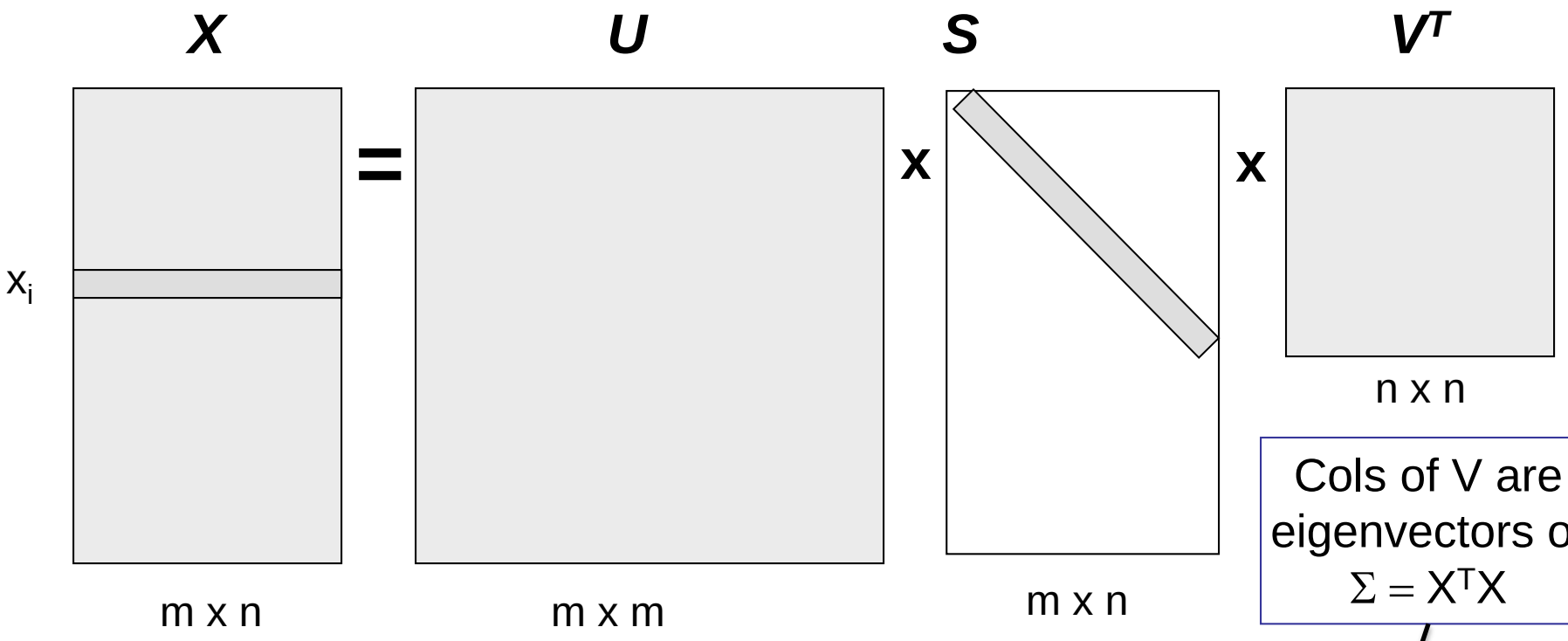


Practical Issue: Scaling Up (Optional)

- Covariance of the image data is BIG!
  - size of $\Sigma = 32768 \times 32768$
  - finding eigenvector of such a matrix is slow.
- SVD comes to rescue!
  - Can be used to compute principal components
  - Efficient implementations available

Singular Value Decomposition: $X=USV^T$

(Optional)



Cols of V are eigenvectors of $\Sigma = X^TX$

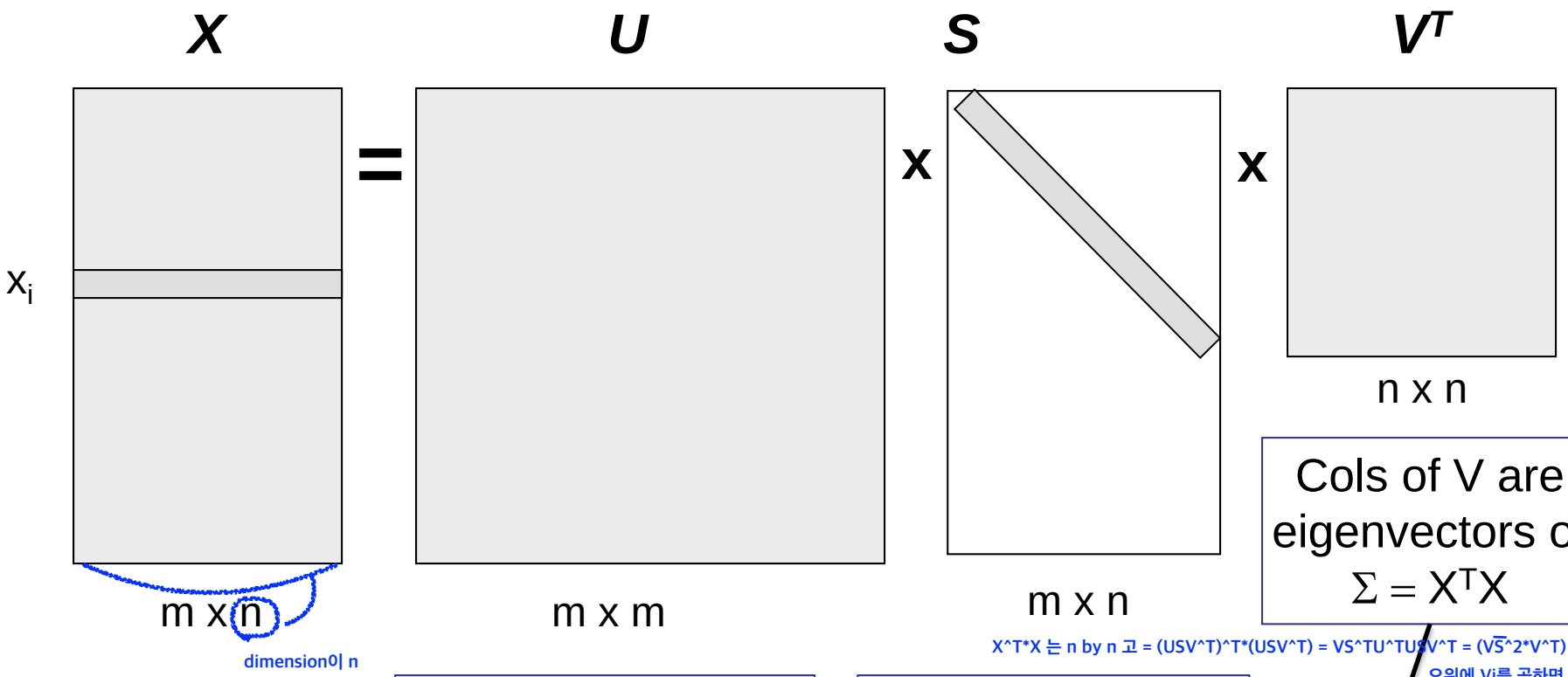
X: our m x n data matrix, one row per data point

Each row of US gives coordinates of a data point in the projected space

Singular matrix: a diagonal matrix, S_i^2 is Σ's i-th eigenvalue

Singular Value Decomposition: $X=USV^T$

(Optional)



X: our m x n data matrix, one row per data point

Each row of US gives coordinates of a data point in the projected space

Singular matrix: a rectangle diagonal matrix, S_i^2 is Σ's i-th eigenvalue

Cols of V are eigenvectors of $\Sigma = X^TX$

X^T*X 는 n by n 고 = (USV^T)^T*(USV^T) = VS^TU^TUSV^T = (VS^2*V^T)

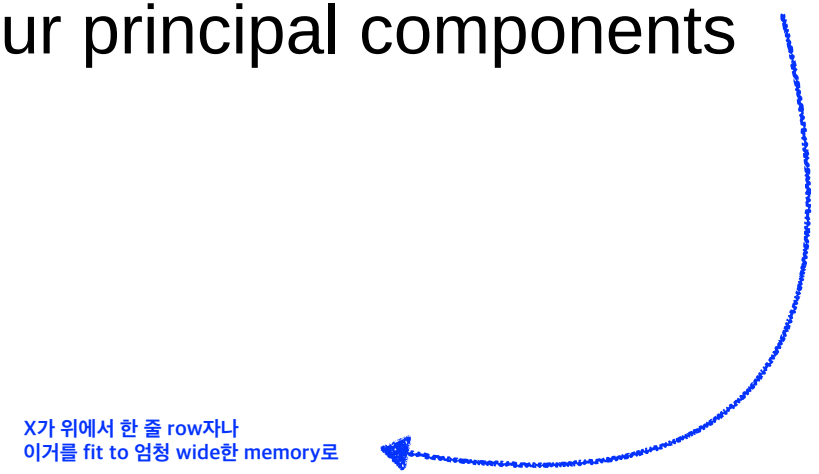
요위에 Vi를 곱하면 Si*Vi

뭔말이냐믄,
V가 eigen vector가된다고

If X is centered, then cols of V are the principal components

SVD for PCA (Optional)

- Create centered data matrix X s.t. $mean(X) = 0_d$
- Solve SVD: $X = USV^T$
- Columns of V are the eigenvectors of Σ sorted from largest to smallest eigenvalues – select the first *k* columns as our principal components

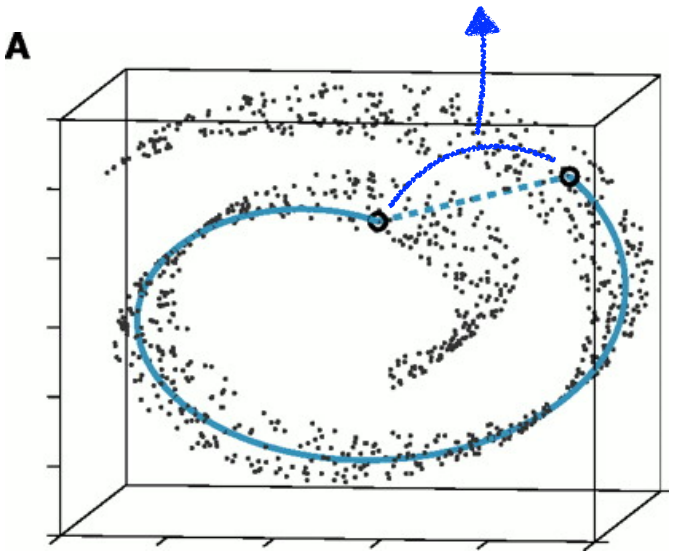


Nonlinear Dimension Reduction

Nonlinear Methods

- Data often lies on or near a nonlinear low-dimensional curve
- We call such low dimension structure manifolds

3D니까 요렇게 디스턴스가있자낭 그래서 말린걸 unroll하려는거양

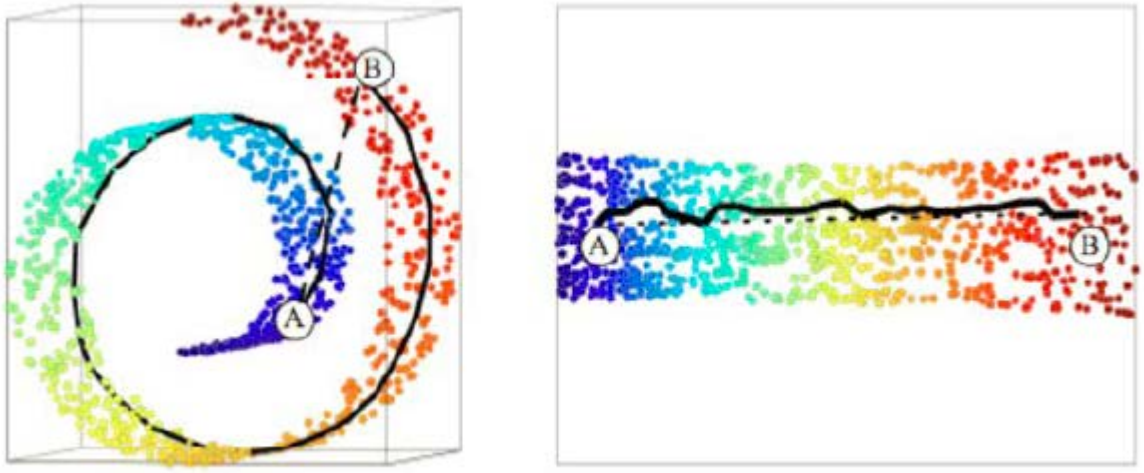


3D공간에 있지만
저 동그랗게 말린 리본에서 데이터들이 안 벗어나잖아? 그래서 2D야 actually
요거를 말린거를 unroll해가지고 2D공간으로 PCA를 계산하눈거양

Swiss roll data

ISOMAP: Isometric Feature Mapping
(Tenenbaum et al. 2000)

- A nonlinear method for dimensionality reduction
- Preserves the global, nonlinear geometry of the data by preserving the geodesic distances
- Geodesic: originally geodesic means the shortest route between two points on the surface of the manifold



ISOMAP

- Two steps
  1. Approximate the geodesic distance between every pair of points in the data
     - The manifold is locally linear
     - Euclidean distance works well for points that are close enough
     - For the points that are far apart, their geodesic distance can be approximated by summing up local Euclidean distances
  2. Find a Euclidean mapping of the data that preserves the geodesic distance

Geodesic Distance

- Construct a graph by
  - Connecting i and j if
    - d(i, j) < ε (ε-isomap) or
    - i is one of j's k nearest neighbors (k-isomap)
  - Set the edge weight equal d(i, j) – Euclidean distance
- Compute the Geodesic distance between any two points as the ***shortest path distance***

Compute the Low-Dimensional Mapping

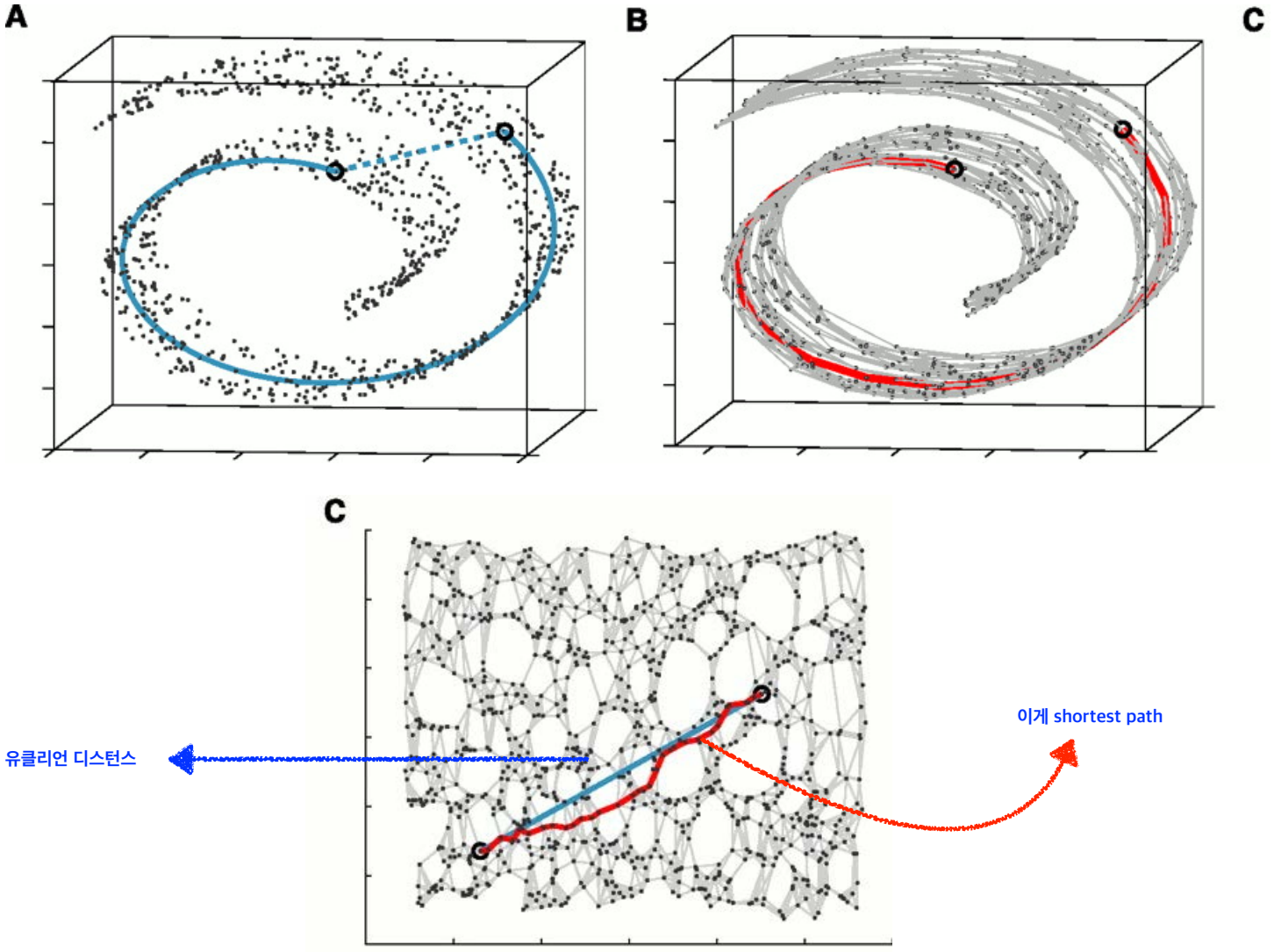
- We can use **Multi-Dimensional scaling** (MDS), a class of statistical techniques that

Given:

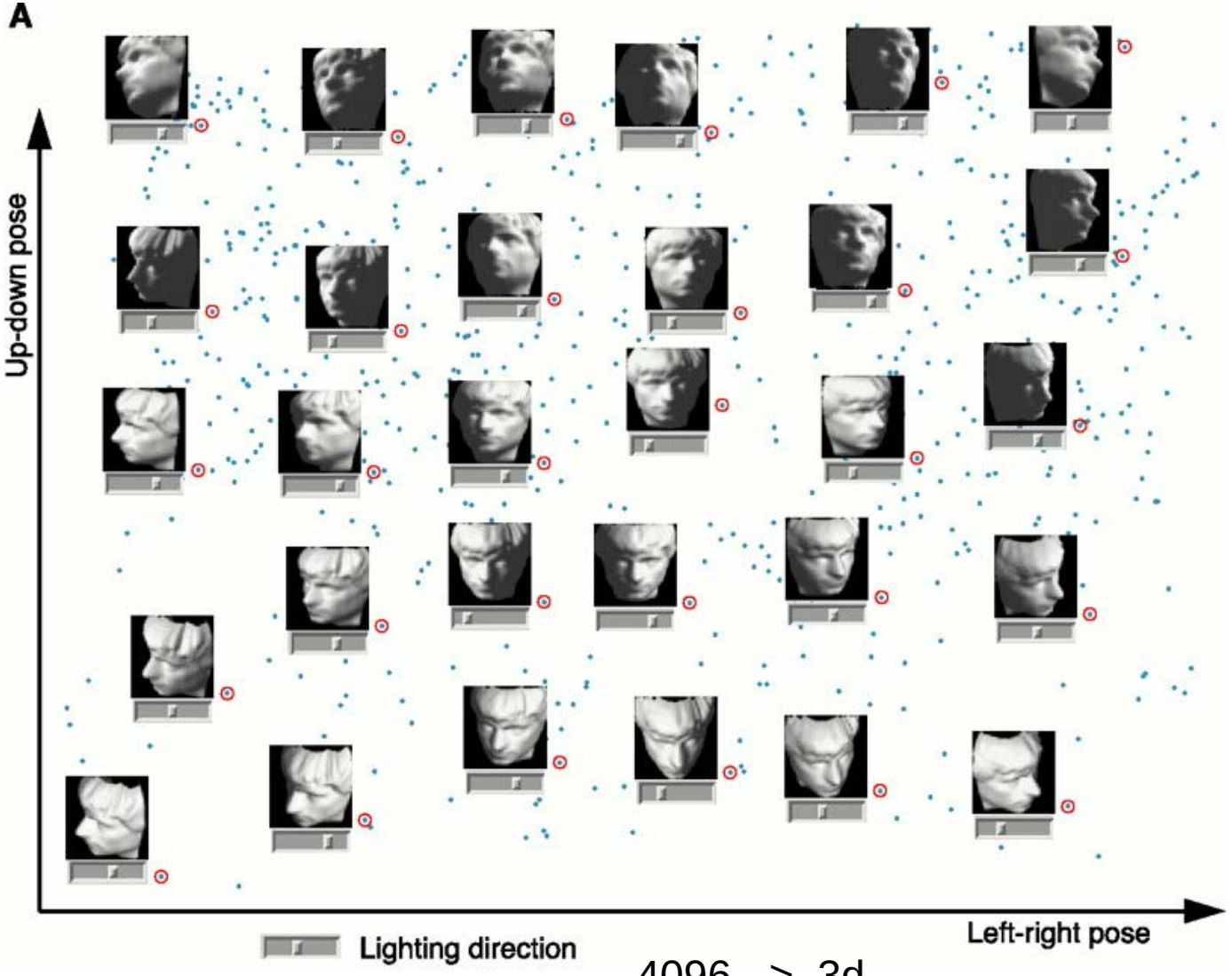
n x *n* matrix of dissimilarities between *n* objects

Outputs: a coordinate configuration of the data in a low-dimensional space R^d whose Euclidean distances closely match given dissimilarities.

ISOMAP on Swiss Roll Data

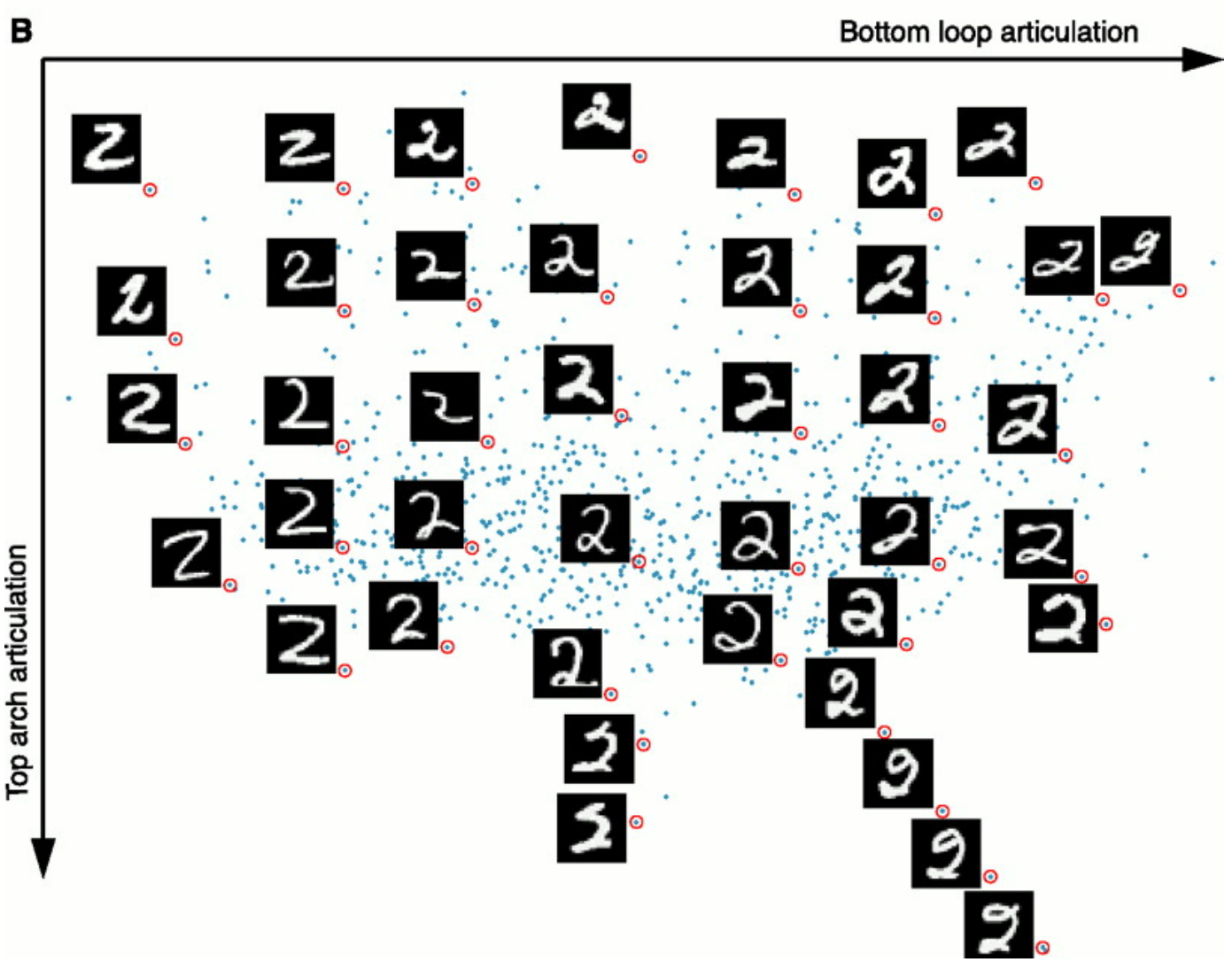


ISOMAP Examples



4096 -> 3d

ISOMAP Examples



Summary of ISOMAP

- Preserve global nonlinear structure by approximating geodesic distance
- Sensitive to the parameters used in the graph construction
  - K: for k-isomap
  - ϵ: for ϵ-isomap 앱실론이 넘 크면 감긴 그 사이 간격, 롤에서 그 초코 롤 간격이 크롭스 너무작으면 disconnect된당
- If data is overly sparse, the shortest path approximation to the geodesic distances can be poor