Chapter 6

김선
서울대학교 컴퓨터 공학부
생물정보 연구소

6

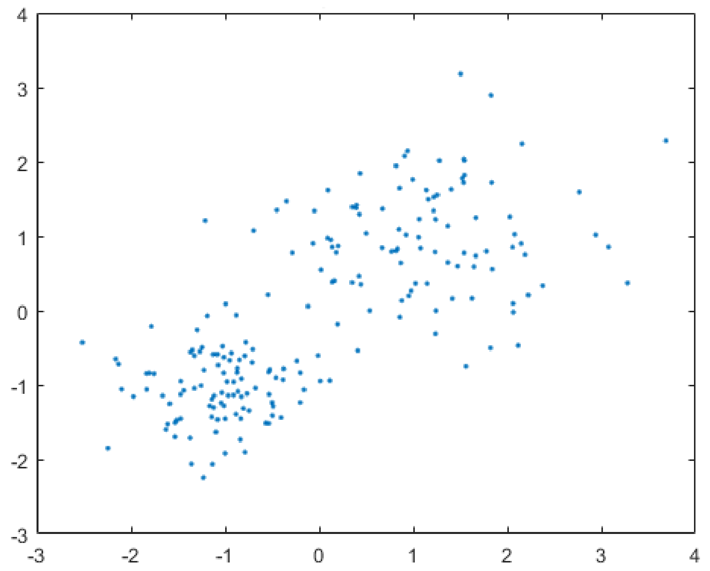
K-평균법: 비지도 학습모델 기초
(K-means algorithm)

김선
서울대학교 컴퓨터 공학부
생물정보 연구소

Unsupervised learning (비지도 학습)

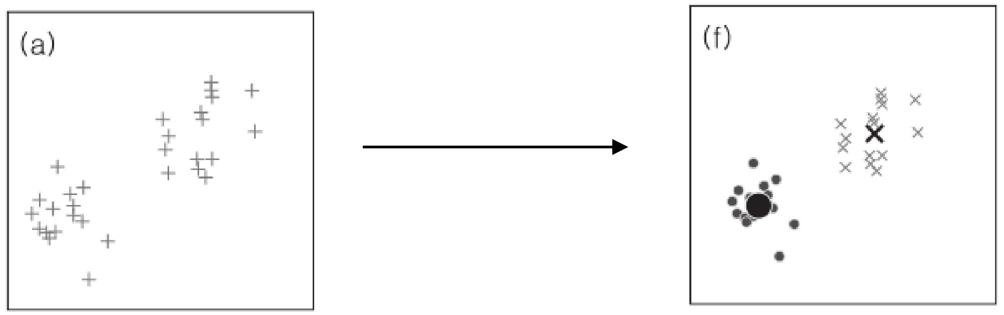
- Unsupervised learning is analyzing data that is not labeled or not classified (목적변수가 없음)

- How many classes can you recognize in the figure?



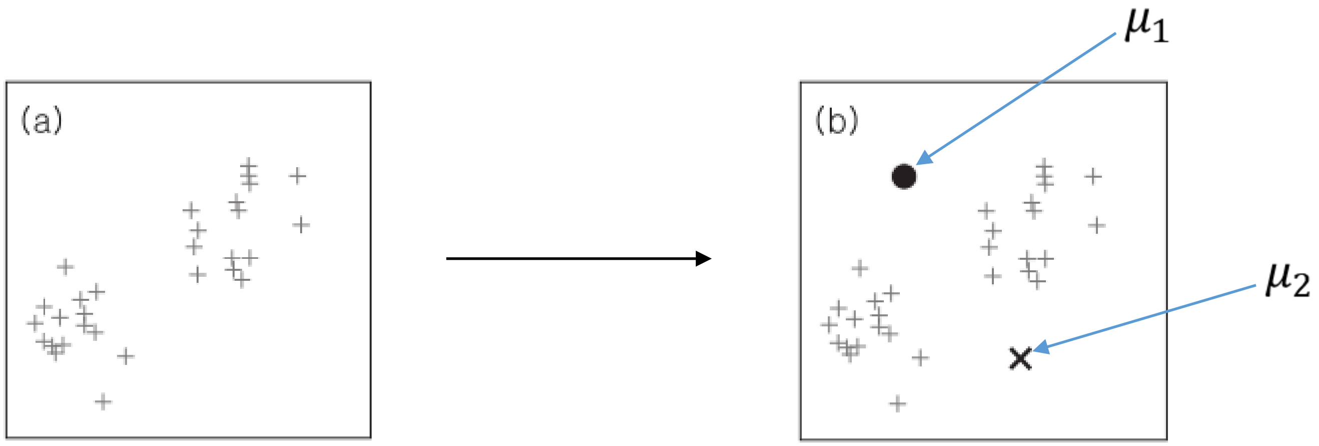
K-means clustering algorithm

- K-means clustering algorithm is an unsupervised machine learning technique
- It finds 'K' clusters where
  - the data **within** a cluster have **high similarity**
  - and the data **across** clusters have **low similarity**
- However, we cannot be 100% sure of the cluster correctness
- So the cluster results need to be further validated using error measures or context knowledge (e.g., within tightness, silhouette score, context enrichment)
- It is similar to EM (Expectation Maximization) clustering



K-means clustering method (1)

- K-means is performed in an iterative way until membership of data does not change or error is converged
- Example:
  - Data: 2-dimensional data (x,y)
    - $\{(x_n, y_n)\}_{n=1}^{N}$, where N is the number of data
  - Select 'K' and randomly initialize them (there are smart ways to initialize K but we will not discuss them here. Also, since the random initialization, the results may change when run again(non-deterministic))
  - Here, K=2 and their initialized value $\{\mu_k\}_{k=1}^{2}$ (centroids)

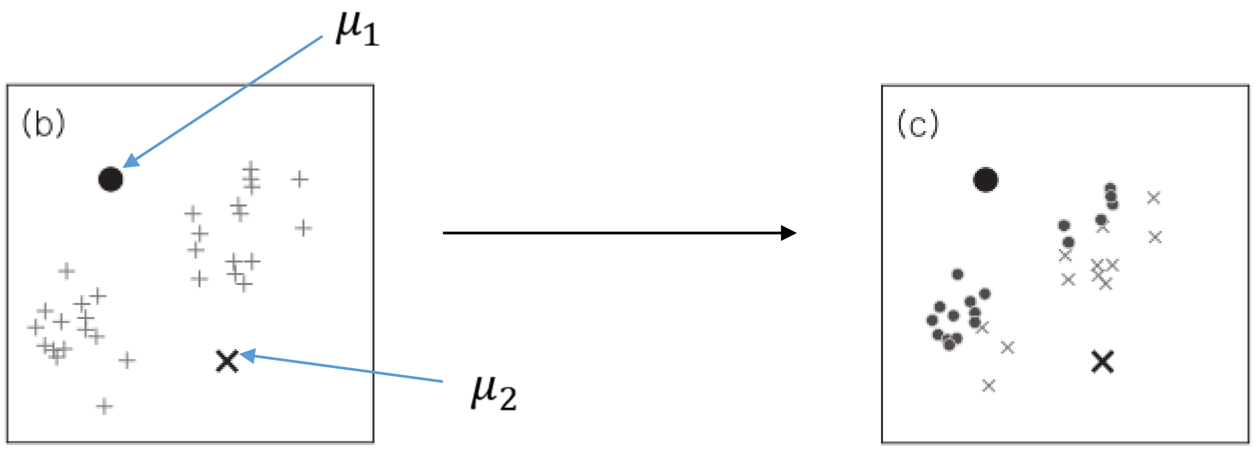


K-means clustering method (2)

- Measure distance from each data $\mathbf{x}_n = (x_n, y_n)^T$ to $\{\mu_k\}_{k=1}^{2}$
- For each data, select the closest μ_k and assign its membership to r_{nk} (indicator function)

$$r_{nk} = \begin{cases} 1 & if\ x_n\ is\ assigned\ to\ \mu_k \\ 0 & otherwise \end{cases}$$

$$ar \min_{\mu_k \in K} dist(\mu_k, \mathrm{x}_n)$$



K-means clustering method (3)

- Update the centroids $\{\mu_k\}_{k=1}^{2}$

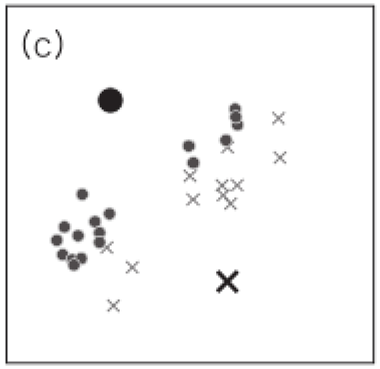
A vector of $(x_n, y_n)^T$

$$\mu_k = \frac{\boxed{\sum \mathbf{x}_n}}{N_k}$$

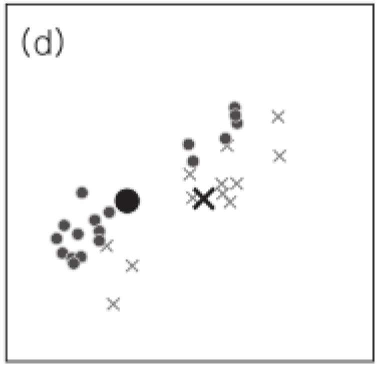
- Here, k=1,2 and N=number of data
- We only want to update each centroid using the data with membership to the centroid. So exactly, μ_k will be

$$\mu_k = \frac{\boxed{\sum_{n=1}^{N} r_{nk}\mathbf{x}_n}}{\sum_{n=1}^{N} r_{nk}}$$

For a data with no membership to r_{nk} will become 0 and not summed (e.g., $0 * \mathbf{x}_n = 0$).
This is the same for the denominator

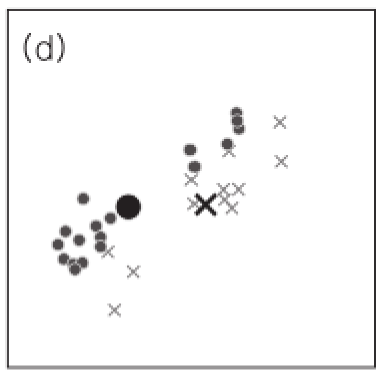


$$\xrightarrow{\mu_k = \frac{\sum \mathbf{x}_n}{N_k}}$$

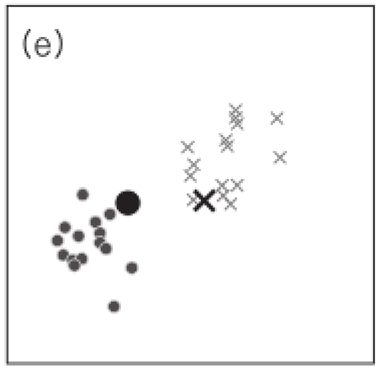


K-means clustering method (4)

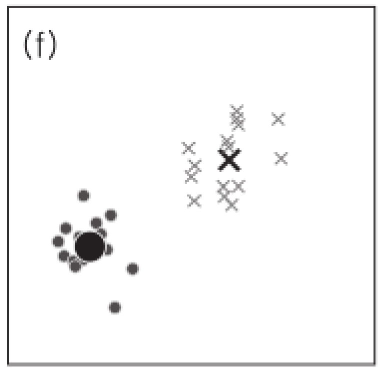
- Now, repeat step (2) and (3) until membership of data does not change or error is minimized (converge by 0.01)



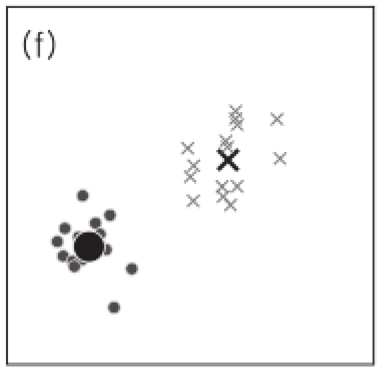
Update centroids $\{\mu_k\}_{k=1}^{2}$



Update memberships r_{nk}



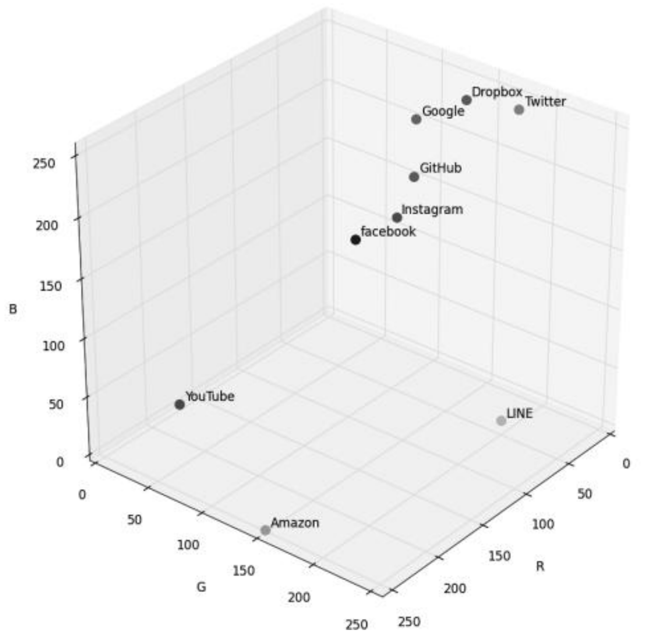
Update centroids $\{\mu_k\}_{k=1}^{2}$



No change in membership. STOP K-means

K-means on image data

- We want to reduce the colors in an image to reduce the file size of the image
  - This is possible if we use less number of colors to represent the image
  - The images can be drawn using RGB (Red, Green, Blue), where each can have a value of range 0~255
  - What colors do we have to choose? Or what colors do we want to remove?
  - K-means clustering can be applied to such problem where it finds the most representative 'K' colors in the image

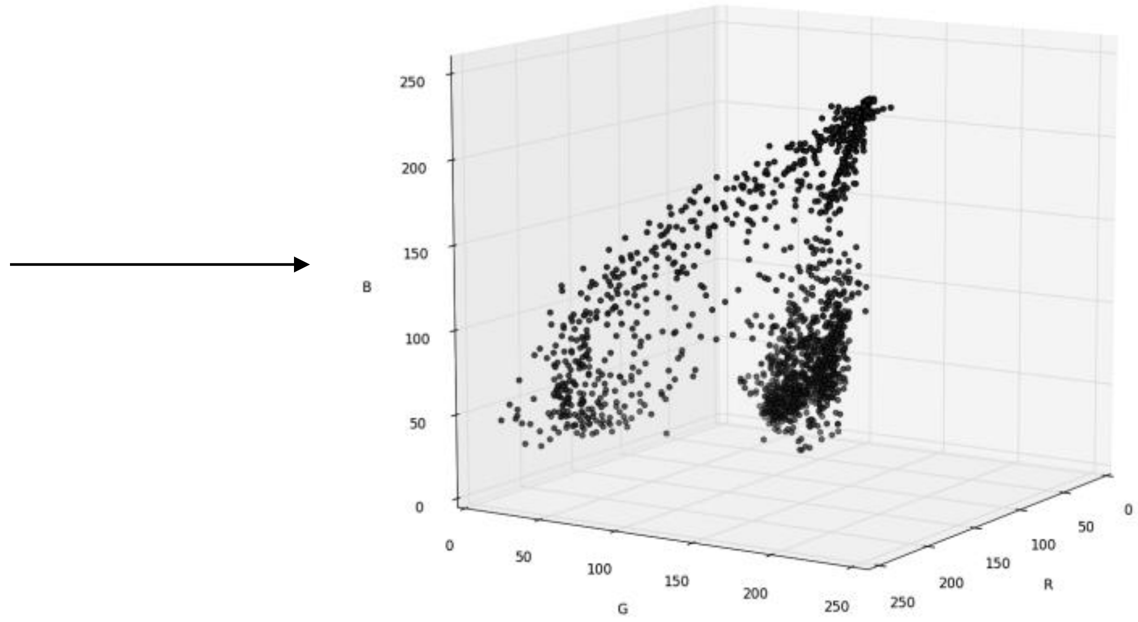


Brand colors

Brand	(R, G, B)
Twitter	(0, 172, 237)
Facebook	(30, 50, 97)
Google	(66, 133, 244)

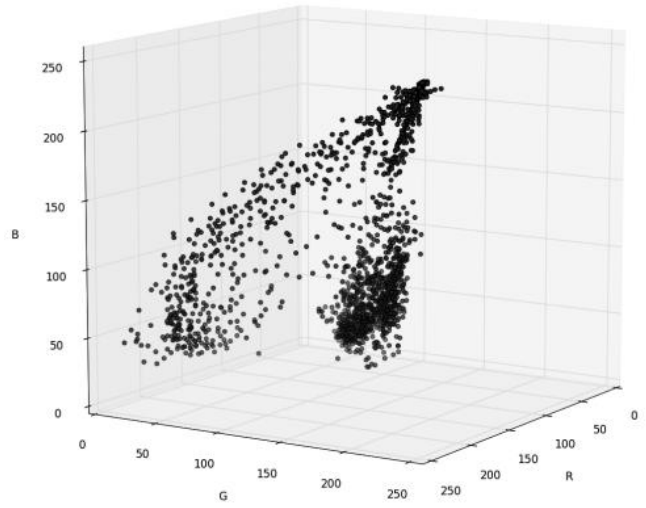
K-means on image data (1)

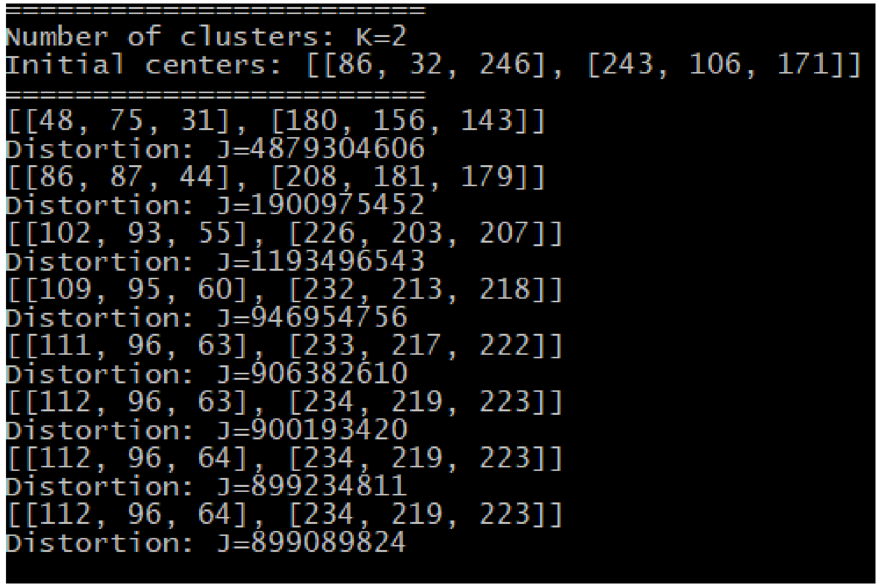
- To find the 'K' representative colors, project each pixel data on the RGB, 3-dimensional space



K-means on image data (2)

- How many representative colors can you identify?
  - Lets set K=2
- For each ‘K’ perform K-means clustering until change in distortion J (error) is converged by < 0.001 (0.1%)
- K-means stopped at 8th iterations



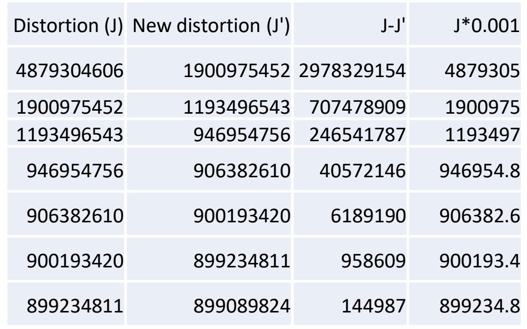


```
========================
Number of clusters: K=2
Initial centers: [[86, 32, 246], [243, 106, 171]]
========================
[[48, 75, 31], [180, 156, 143]]
Distortion: J=4879304606
[[86, 87, 44], [208, 181, 179]]
Distortion: J=1900975452
[[102, 93, 55], [226, 203, 207]]
Distortion: J=1193496543
[[109, 95, 60], [232, 213, 218]]
Distortion: J=946954756
[[111, 96, 63], [233, 217, 222]]
Distortion: J=906382610
[[112, 96, 63], [234, 219, 223]]
Distortion: J=900193420
[[112, 96, 64], [234, 219, 223]]
Distortion: J=899234811
[[112, 96, 64], [234, 219, 223]]
Distortion: J=899089824
```

Example of running K-means on the image data

Stop if
Distortion – new distortion < distortion * 0.001

Stop

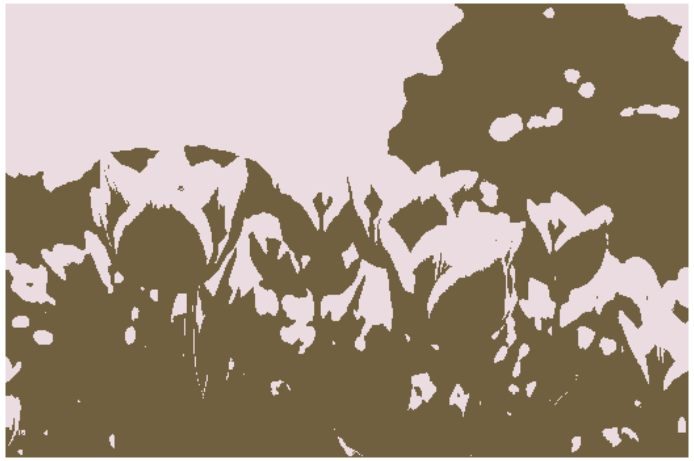


Distortion (J)	New distortion (J')	J-J'	J*0.001
4879304606	1900975452	2978329154	4879305
1900975452	1193496543	707478909	1900975
1193496543	946954756	246541787	1193497
946954756	906382610	40572146	946954.8
906382610	900193420	6189190	906382.6
900193420	899234811	958609	900193.4
899234811	899089824	144987	899234.8

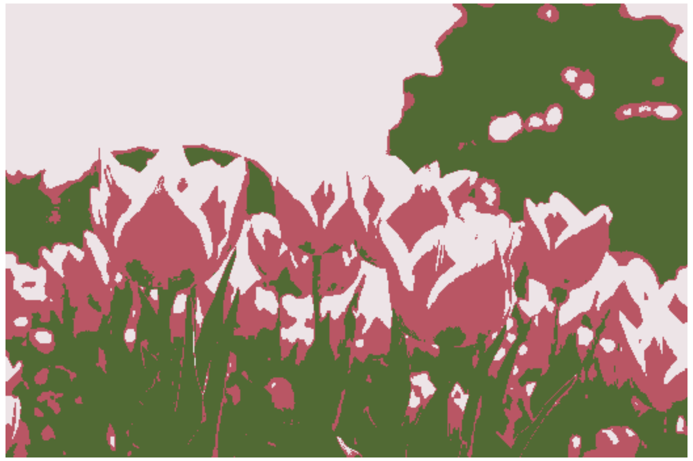
K-means on image data (3)

- K-means clustering results with K={2,3,5,16}

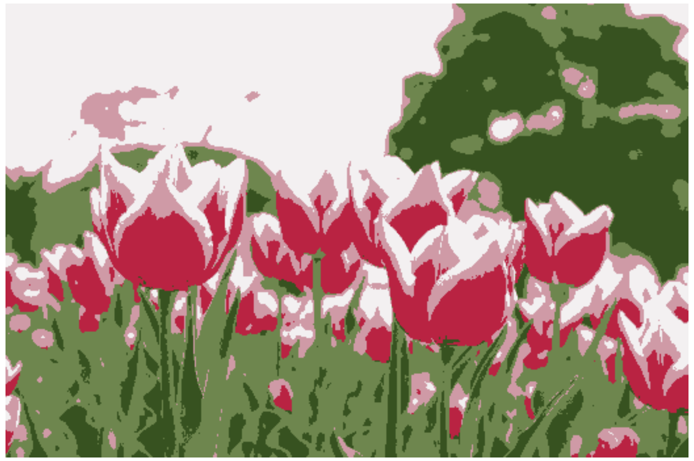
K=2



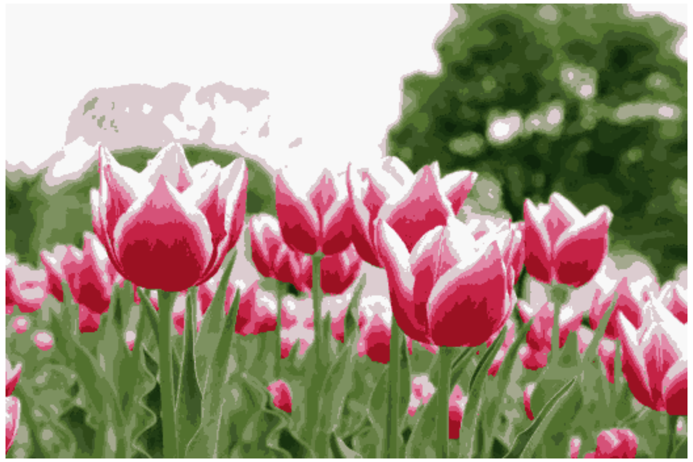
K=3



K=5



K=16



Distortion J (error)

- Distortion J is the sum of squared distance between the data and the 'K' centroids. Also known as the cluster tightness.

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|^2$$

  - N=number of data,
  - K=number of clusters,
  - r_{nk}=membership of data,
  - x_n data,
  - μ_k=centroid of cluster *k*

- A smaller J of cluster *k* implies that the data in cluster k are more close, or similar, to each other
- Distortion J always decreases with each iteration, which can be proved mathematically

Why does J decrease?

- [수학을 배우는 작은방] p.177

$$r_{nk} = \begin{cases} 1 & k = \underset{k'}{armin}\|x_n - \mu_{k'}\| \\ 0 & otherwise \end{cases}$$

Data x_n is a vector with i elements

$$[x_n]_i = [\overset{i=1}{255}, \overset{i=2}{0}, \overset{i=3}{0}]$$

- Get derivative of $J = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\,\|[x_n]_i - [\mu_k]_i\|^2$
- $\frac{\partial J}{\partial[\mu_k]_i} = -2\sum_{n=1}^{N} r_{nk}([x_n]_i - [\mu_k]_i) = 0$

$$= -2\sum_{n=1}^{N} r_{nk}[x_n]_i + 2\sum_{n=1}^{N} r_{nk}\,[\mu_k]_i = 0$$

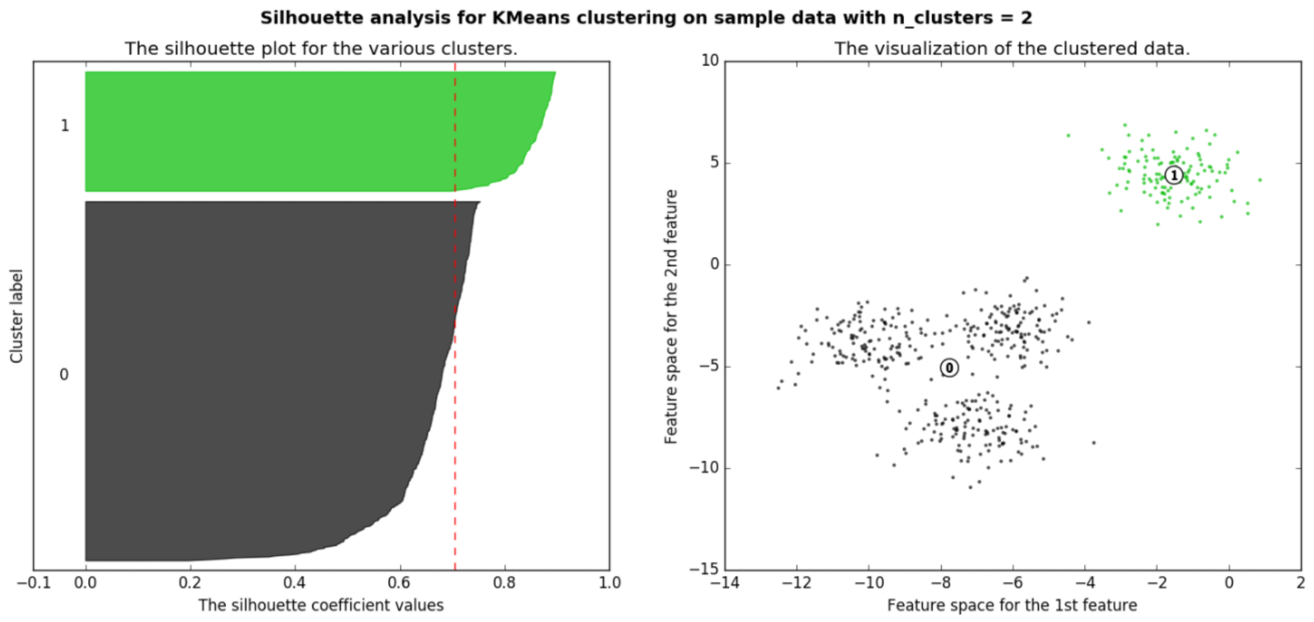
$$= \; 2\sum_{n=1}^{N} r_{nk}\,[\mu_k]_i = 2\sum_{n=1}^{N} r_{nk}[x_n]_i$$

- So we can see that
  - $[\mu_k]_i = \frac{\sum_{n=1}^{N} r_{nk}[x_n]_i}{\sum_{n=1}^{N} r_{nk}}$
- $\mu_k = \frac{\sum_{n=1}^{N} r_{nk}\mathbf{x}_n}{\sum_{n=1}^{N} r_{nk}}$ $\longleftarrow$ This is the same equation we have seen before. (Centroid equation)

By this, we can see that J never increases during K-means iteration

Measuring cluster quality

- There are several ways of evaluating the quality of the K-means clustering result
  1. **Cluster tightness** (distortion), which measures only the similarity of data within each cluster. The smaller the distortion, the tighter a cluster is.
  2. **Silhouette score**, which measures how well separate the clusters are (dissimilarity between clusters). The larger the silhouette score, the more distant are the clusters from each other. (range is from -1 to 1)
  3. **Rand index**, which can be measured only if labels are available for the clusters. Hence, this is done for simulated data where the labels (answers) are pre-embedded in the data and test how well the clustering algorithm clusters the labels.



Example:

K=2

Data: 2 dimensional space

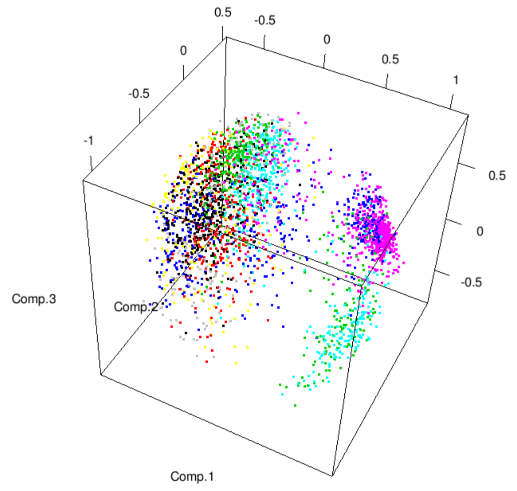
A **silhouette score** for each cluster is calculated and the average score is the result.

Here, the average silhouette score is 0.7, which is quite good.

Applications of K-means

- Many applications make use of K-means due to its simplicity and effectiveness
  - Image clustering: cluster similar images together
  - Gene clustering: find genes with similar expression patterns
  - Audio clustering: find songs with similar melodies
  - Document clustering: cluster documents that have similar contents, such as news

- Also there are several distance metrics used for clustering
  - Euclidean distance
  - Cosine similarity distance
  - Manhattan distance



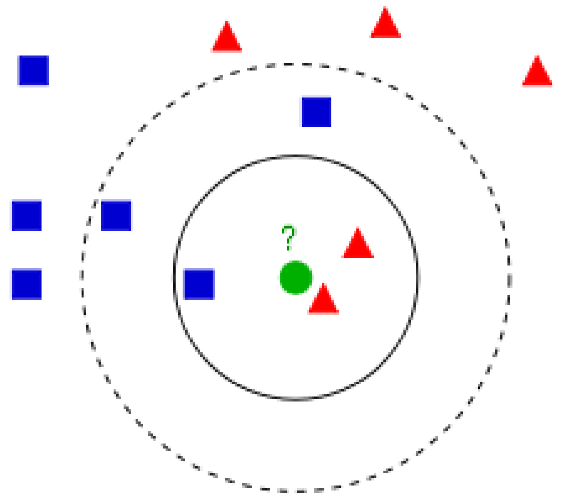
Spherical K-means clustering projects data points to high dimensional space where data are transformed to unit vectors so that they are projected to a sphere.

K-Nearest Neighbor (KNN, k-최근접이웃 기법)

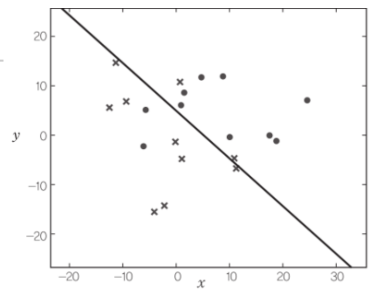
- The KNN is not a clustering algorithm but a classification and therefore a supervised machine learning technique
- KNN is very simple and also searches for hidden pattern in data
- KNN is a type of lazy or instance-based learning because it does not make effort in computing the entire data set to learn any hidden patterns. Instead it locally classifies the data at the time when the data is given. No pre-learned model.

KNN method

- Simply, the KNN classifies a unknown input data (green) by its ‘K’ nearest data by a voting procedure
- If there are two classes, 1) blue square, 2) red triangle, what class should the green data be classified to if ‘K’=3?
- The three nearest data are 1 blue square and 2 red triangles
- By voting, it should be classified as a red triangle



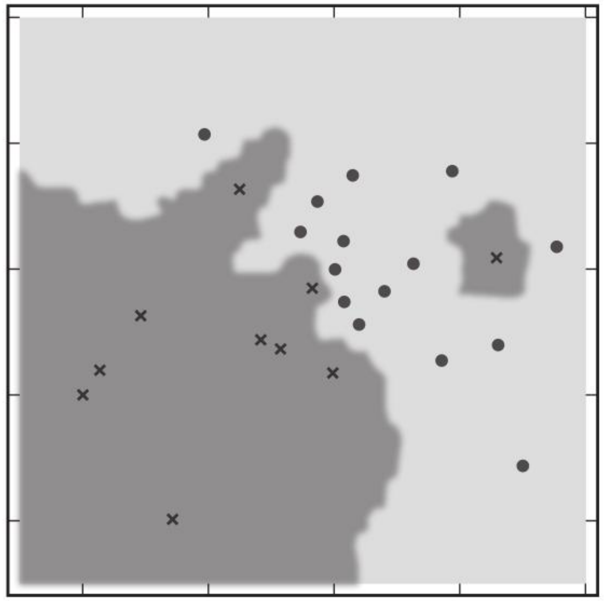
KNN example



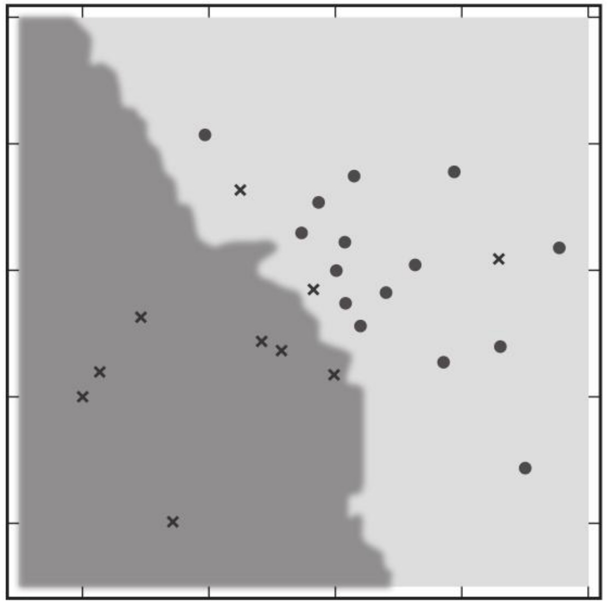
• = virus infected
X = non-infected

- For a labeled training set $\{(x_n, y_n, t_n)\}_{n=1}^{N}$ as in Ch 1.3.2 perform KNN
- Set K={1, 3}
- In case of K=1, the data itself votes for its class. So the classification error will be 0%, however this will suffer from overfitting
- K=3 shows a more reliable classification result

K=1



K=3



KNN drawbacks

- The KNN suffers from two major drawbacks

1. The classification of newly incoming data takes time for large data. In case of linear regression or perceptron, they will find optimized weights for a formula. Incoming data can be simply be classified by using the formula. However, KNN needs to lazily decide the class of incoming data by selecting the 'K' nearest neighbor and performing the voting process.
2. KNN is a non-parametric, non-model ML technique. Thus, even if KNN succeeded in identifying a hidden pattern from the data, it is difficult to formalize it. There is a big chance that the pattern wont be able to be presented as a linear formula making it impractical to use in business models. Also, it will be difficult to comprehend such pattern.

6

Principal Component Analysis (PCA)

김선
서울대학교 컴퓨터 공학부
생물정보 연구소

What is PCA?

- A statistical technique used to examine the interrelations among a set of variables in order to identify the underlying structure of those variables. Also called factor analysis.

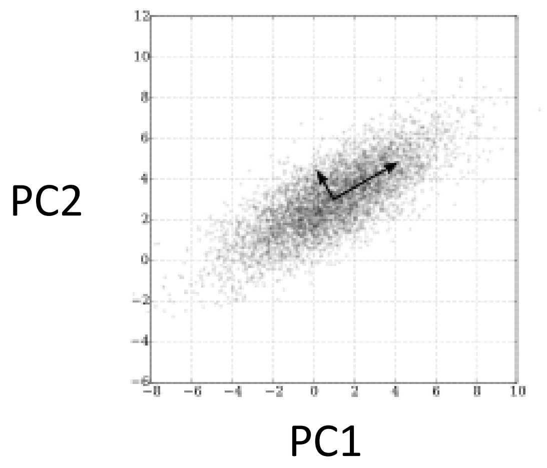
- It is a **non-parametric** analysis and the answer is unique and independent of any hypothesis about data distribution.

- These two properties can be regarded as weaknesses as well as strengths.
  - Since the technique is non-parametric, no prior knowledge can be incorporated.
  - PCA data reduction often incurs a loss of information.

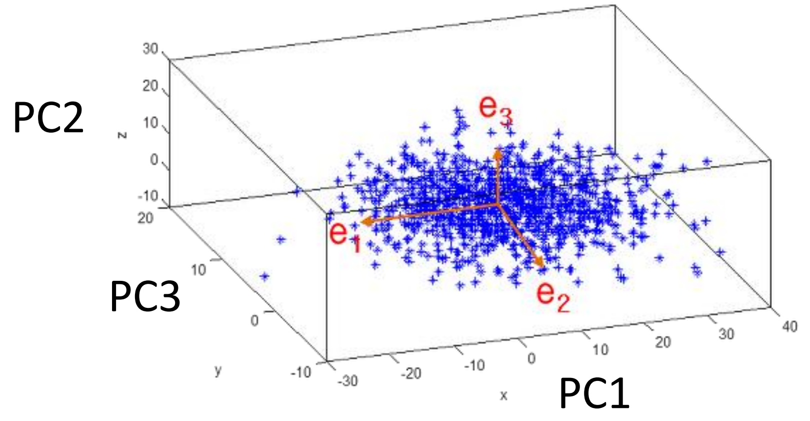
Why use PCA?

- With a large number of features, the dispersion matrix may be too large to study and interpret properly.
- Graphical display of data may also not be of particular help incase the data set is very large.
- With 12 features, there will be more than 200 three-dimensional scatterplots to be studied!
- To interpret the data in a more meaningful form, it is therefore necessary to **reduce the number of feature**s to a few, interpretable **linear combinations** of the data.
- Each linear combination will correspond to a principal component.

2D PCA



3D PCA



Assumptions of PCA

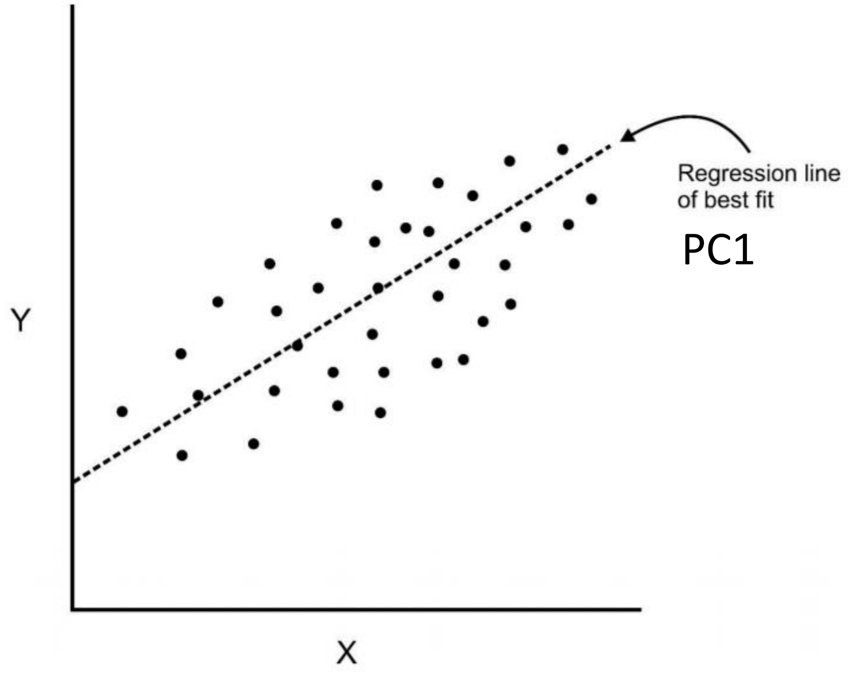
1. **Linearity**
   - Assumes the data set to be linear combinations of the variables

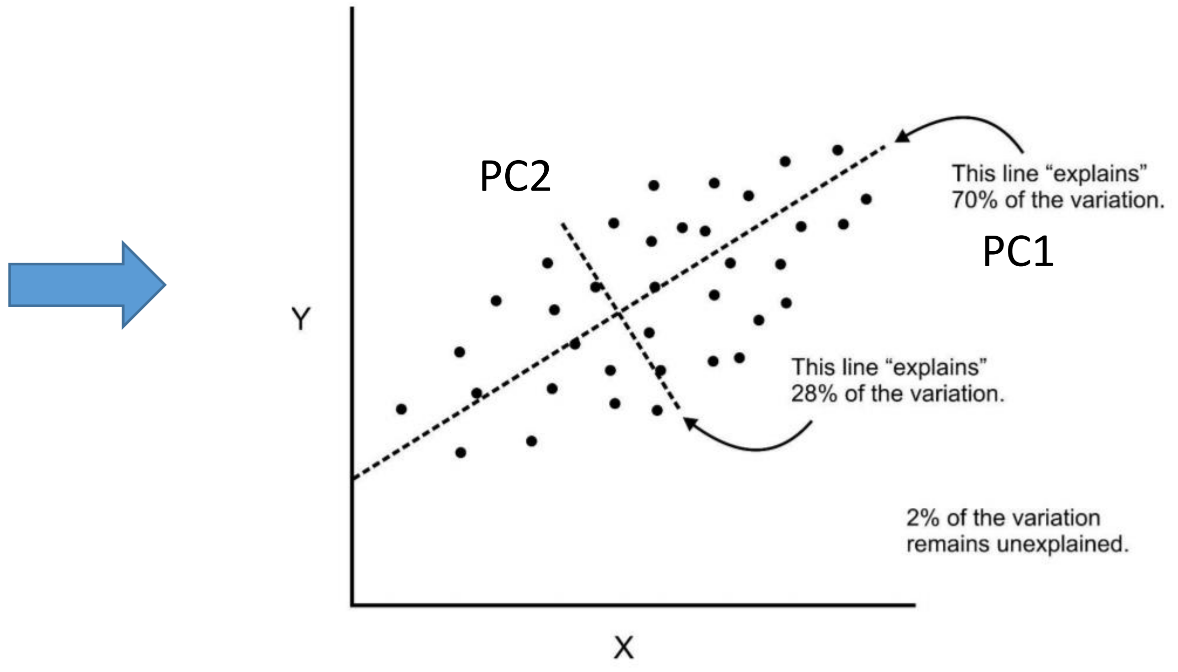
2. **Importance of mean and covariance**
   - There is no guarantee that the directions of maximum variance will contain good features for discrimination.

3. **Large variances have important dynamics**
   - Assumes that components with larger variance correspond to interesting dynamics and lower ones correspond to noise

PCA method

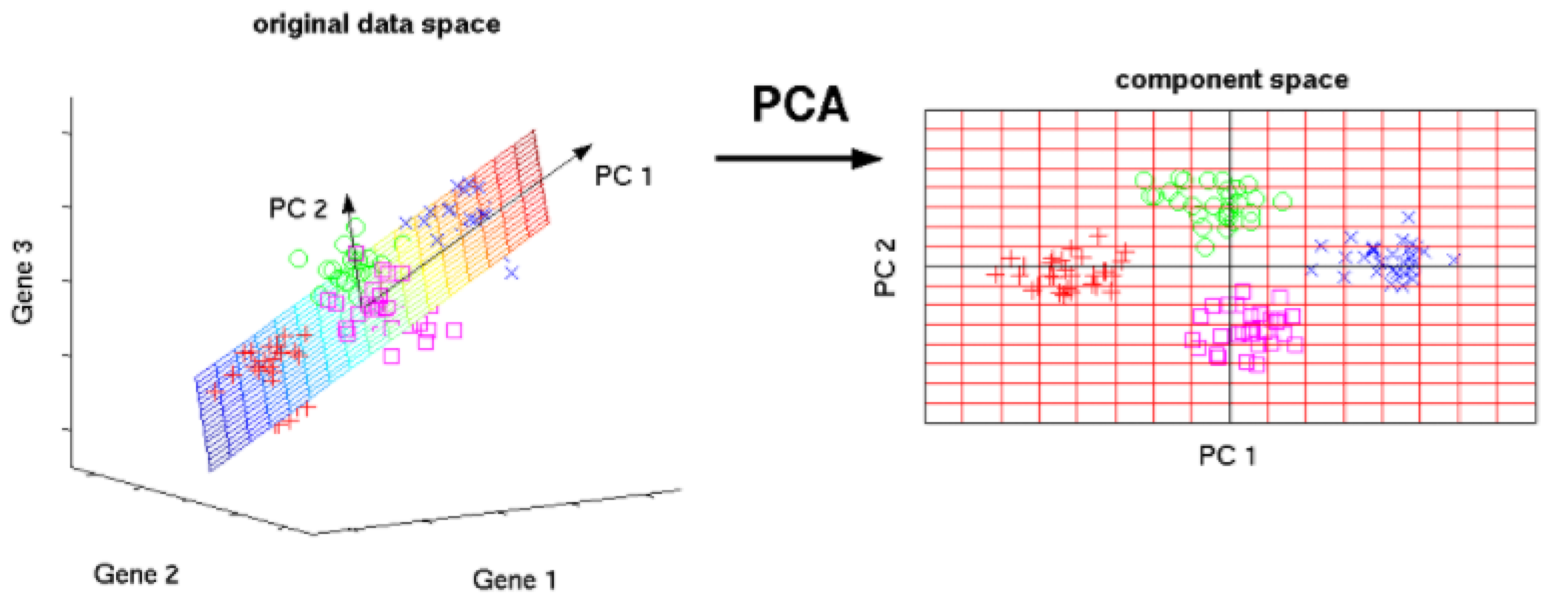
- Select regression line of best fit as PCA1
- Select the 2nd as PCA2 and so forth.
- PCAs are uncorrelated to each other so are **orthogonal** to each other





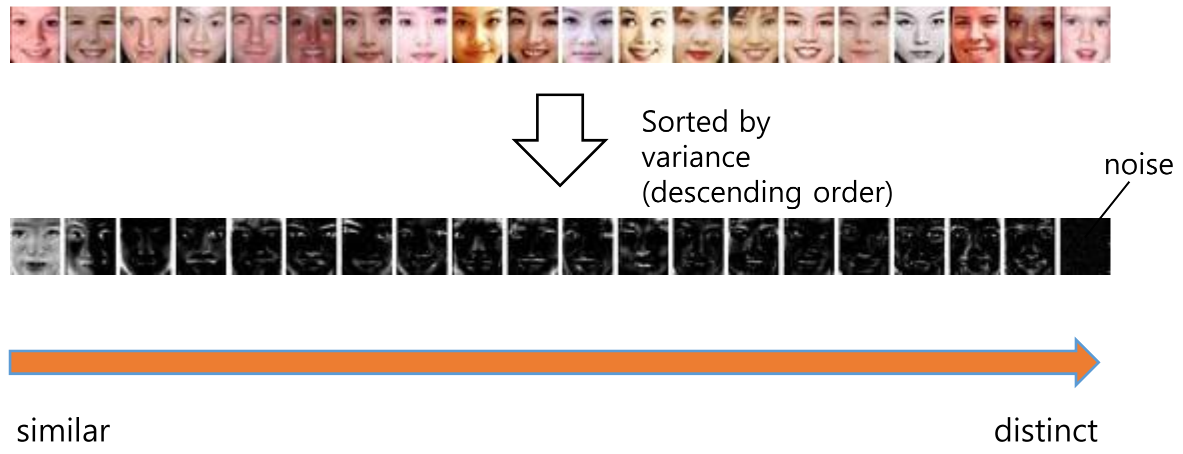
PCA Example

- For three genes, their expression values in 4 different diseases are measured
- By performing PCA, we can observe in the component space that their expression values clearly vary between the diseases



PCA applications

- PCA used in many domains
  - Face recognition, statistical data mining, data compression (dimension reduction), noise reduction etc.
- Eg.) 20 images of 45x40 pixel size (1800 dimensional vectors)



Using PCA for reconstructing images

- Each vector represent a coefficient of each attribute
- The original data can be represented using a set of PCs, which will be dimension reduction – faster computation (or the whole PC set)
- Hence, using the PC coefficients, linear regression can be done