Artificial Intelligence III:
Artificial Intelligence and Deep Learning

Lecture 6

Unsupervised Learning

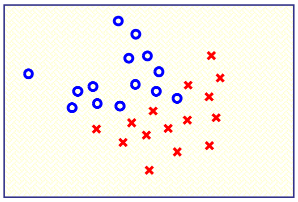
Dr. Patrick Chan
patrickchan@ieee.org
South China University of Technology, China

Agenda

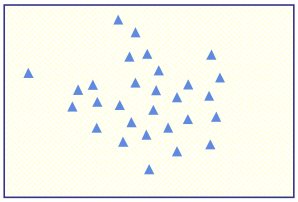
- Introduction
- Clustering
  - Similarity Measure
  - Criterion Function
  - Algorithm
- Feature Extraction
  - Principal Components Analysis

Supervised VS Unsupervised

- Supervised Learning
  - Label is given
  - Someone (a supervisor) provides the true answer



- Unsupervised Learning
  - No Label is given
    "Learning without a teacher"
  - Much harder than supervised learning
  - You never know the correct answer
    - How to evaluate the result?



Unsupervised Learning

- No Supervision (No Label)
  How to evaluate the result?
  - External: Expert comments
    - Expert may be wrong
  - Internal: Objective functions
    - E.g. Distance between samples and centers
    - Very intuitive

- Different from Supervised Learning, the evaluation method is subjective

Why No Label?

- Label is expensive
  - Especially for a huge dataset
  - E.g. Medical application
- Sometimes, the objective is not clear
  - Data Mining
- Gain some insight about the data structure before designing classifiers
  - E.g. Feature selection

Unsupervised Learning Type

- Parametric Approach
  - Assume distribution is known
  - Estimate parameters of distribution
    - E.g. Maximum-Likelihood Estimate
- Non-Parametric Approach
  - No assumption on the distribution
  - Group data into clusters
    - Samples in the same group share something in common

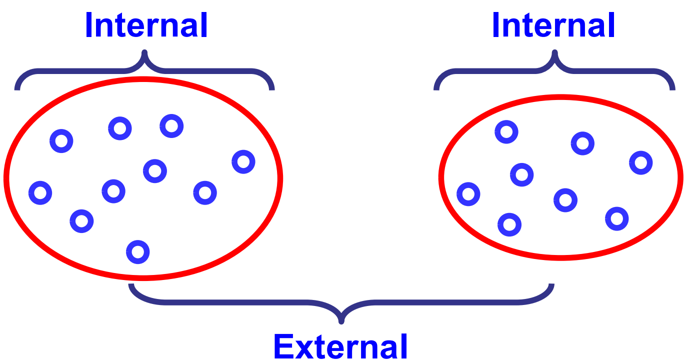
Clustering

- How many clusters (groups) are there?

- How can you know it?

Clustering

- How many clusters (groups) are there?

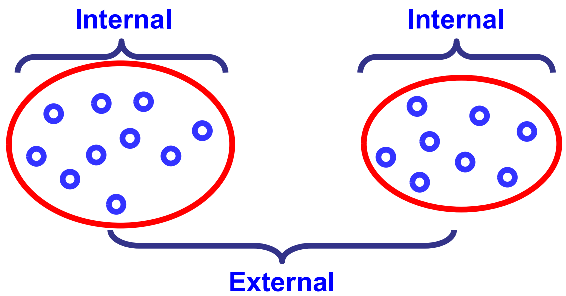


- Assumptions
  - Internal Characteristic (intra-cluster):
    - Distance within a cluster should be small
  - External Characteristic (inter-cluster):
    - Distance between clusters should be large

Clustering

Three Important Factors

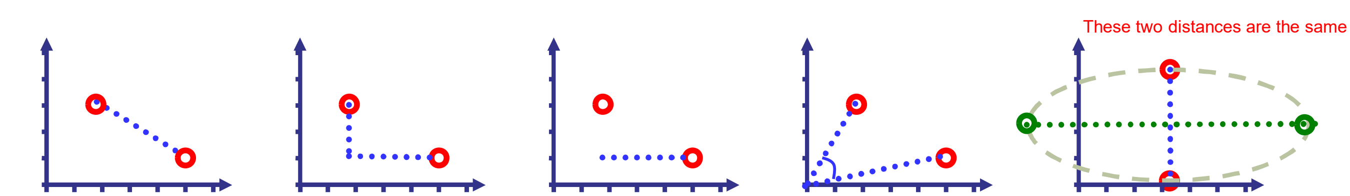
- Distance (Similarity) Measure
  - How similar between two samples?
- Criterion Function
  - What kind of clustering result is expected?
- Clustering Algorithm
  - E.g. optimize the criterion function



Clustering

Similarity Measure

- No best measure for all cases
- Application dependent



Euclidean Distance (L2)

$$d(x_1,x_2)=\sqrt{\sum_{k=1}^{n}(x_{1_k}-x_{2k})^2}$$

Manhattan Distance (L1)

$$d(x_1,x_2)=\sum_{k=1}^{n}|x_{1_k}-x_{2k}|$$

Chebyshev Distance

$$d(x_1,x_2)=\max_{1\le k\le n}(|x_{1_k}-x_{2k}|)$$

Cosine Similarity

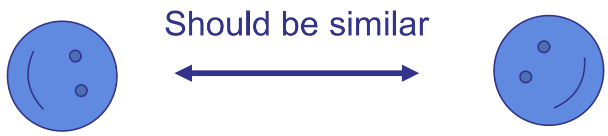
$$d(x_1,x_2)=\frac{x_1{}^t x_2}{\|x_1\|\|x_2\|}$$

Mahalanobis Distance

$$d(x_1,x_2)=\sqrt{\sum_{k=1}^{n}\frac{(x_{1_k}-x_{2k})^2}{\sigma_k^2}}$$

Similarity Measure

- No best measure for all cases
- Application dependent
  - Examples:
    - Rotation Invariance in Face Recognition



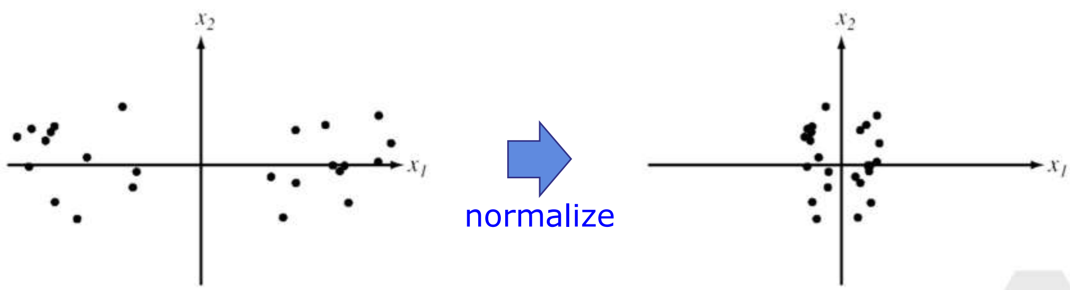
    - NO Rotation Invariance in Character Recognition

9 ⟷ **6**

Should be different

Similarity Measure

- Scale of features may be different
  - Different Ranges (Weight: 80 – 300, waist width: 28 – 45)
  - Different Units (Km VS mile, cm VS meter)
- May be solved by normalized, e.g. [0, 1]
  - Sometimes may not be suitable
  - normalization reduces cluster effect (right diagram)



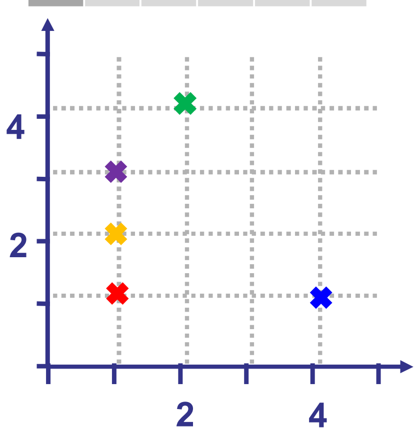
Naïve Clustering Algorithm

- A naïve clustering algorithm can be developed only based on similarity measure between samples
- Algorithm:
  - Calculate similarity for each sample pair
  - Group the samples in the same cluster if the measure between them is less than a threshold (d_0)

Naïve Clustering Algorithm

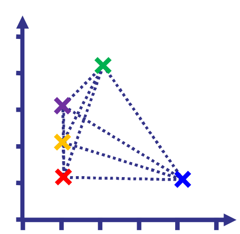
* Euclidean Distance is used

	✖	✖	✖	✖	✖
✖	0	1.0	2.0	3.2	3.0
✖	1.0	0	1.0	2.2	3.2
✖	2.0	1.0	0	1.4	3.6
✖	3.2	2.2	1.4	0	3.6
✖	3.0	3.2	3.6	3.6	0



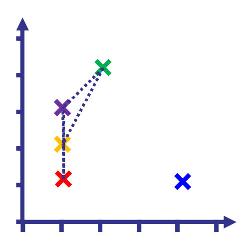
Large threshold
(d_0 = 4.0)
1 cluster

	×	×	×	×	×
×	0	1.0	2.0	3.2	3.0
×	1.0	0	1.0	2.2	3.2
×	2.0	1.0	0	1.4	3.6
×	3.2	2.2	1.4	0	3.6
×	3.0	3.2	3.6	3.6	0



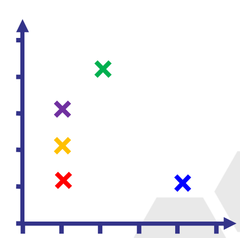
Medium threshold
(d_0 = 2.5)
2 clusters

	×	×	×	×	×
×	0	1.0	2.0	3.2	3.0
×	1.0	0	1.0	2.2	3.2
×	2.0	1.0	0	1.4	3.6
×	3.2	2.2	1.4	0	3.6
×	3.0	3.2	3.6	3.6	0



Small threshold
(d_0 = 0.5)
5 clusters

	×	×	×	×	×
×	0	1.0	2.0	3.2	3.0
×	1.0	0	1.0	2.2	3.2
×	2.0	1.0	0	1.4	3.6
×	3.2	2.2	1.4	0	3.6
×	3.0	3.2	3.6	3.6	0



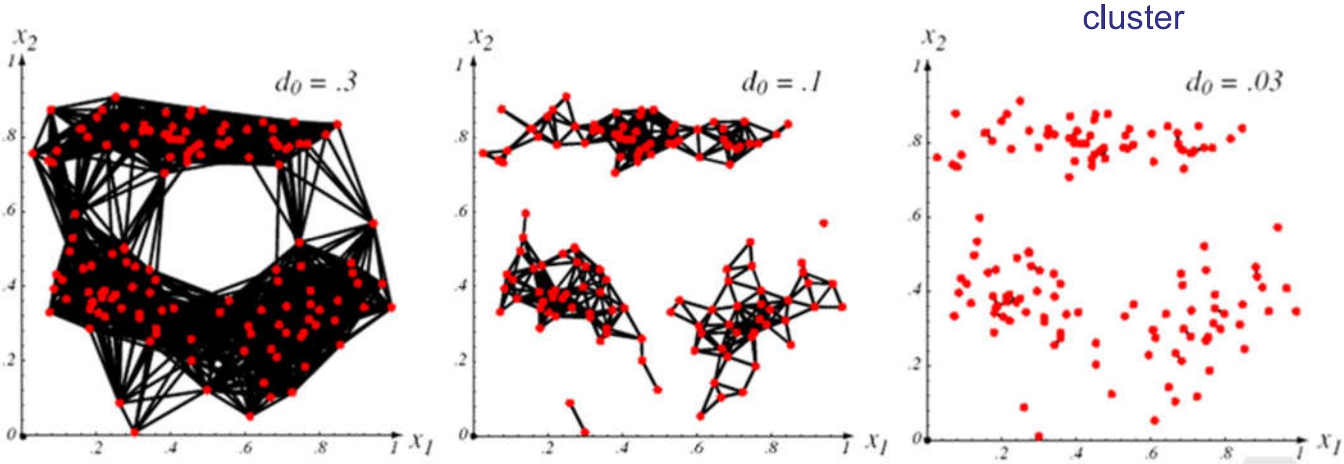
Naïve Clustering Algorithm

- Another Example

Large threshold
One Cluster

Medium threshold
Reasonable Result

Small threshold
Every sample is a cluster



Naïve Clustering Algorithm

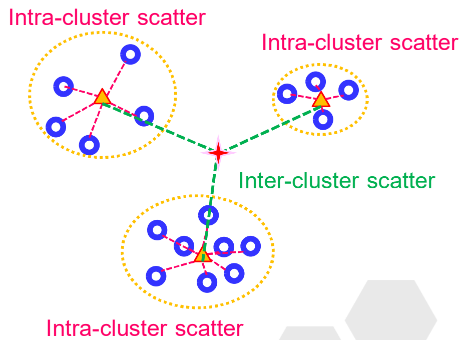
- Advantage:
  - Easy to understand
  - Simple to implement
- Disadvantage:
  - Only local information is considered
  - Highly dependent on the threshold

Criterion Function

- Commonly used criteria
  - **Intra-cluster scatter**
    (Variance of each cluster)
    Smaller is better
    $$S_A = \sum_{i=1}^{c} \sum_{x \in D_i} (x-\mu_i)(x-\mu_i)^t$$
  - **Inter-cluster scatter**
    (Distance between clusters)
    Larger is better
    $$S_E = \sum_{i=1}^{c} n_i(\mu_i-\mu)(\mu_i-\mu)^t$$
  - **Combination**
    $$S = \alpha S_A - (1-\alpha) S_E$$

$$\mu_i = \frac{1}{n_i}\sum_{x \in D_i} x \qquad \mu = \frac{1}{n}\sum_{x \in D} x$$

c : the number of cluster
n : the number of samples
n_i : the number of samples in cluster i
D : the set of all samples
D_i : the set of samples in cluster i
α : the tradeoff



Criterion Function: Example

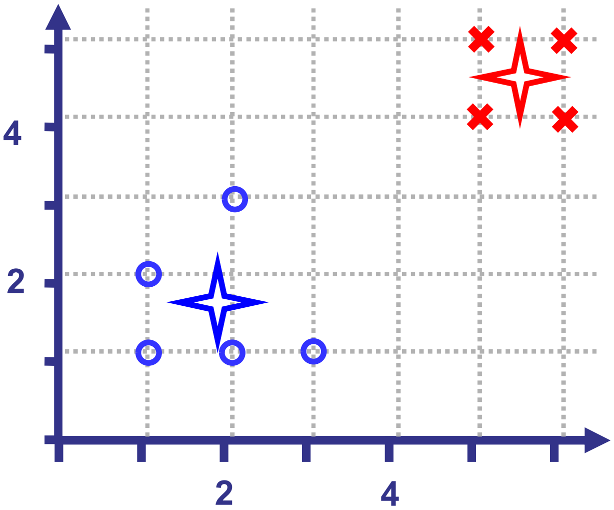
* Euclidean Distance is used

- Intra-cluster scatter

$$S_A = \sum_{i=1}^{c} \sum_{x \in D_i} (x-\mu_i)(x-\mu_i)^t$$

$$\mu_1 = \frac{1}{4}\begin{pmatrix} [5 \quad 4]+[5 \quad 5] \\ +[6 \quad 4]+[6 \quad 5] \end{pmatrix} = [5.5 \quad 4.5]$$

$$\mu_2 = \frac{1}{5}\begin{pmatrix} [1 \quad 1]+[1 \quad 2] \\ +[2 \quad 1]+[2 \quad 3] \\ +[3 \quad 1] \end{pmatrix} = [1.8 \quad 1.6]$$



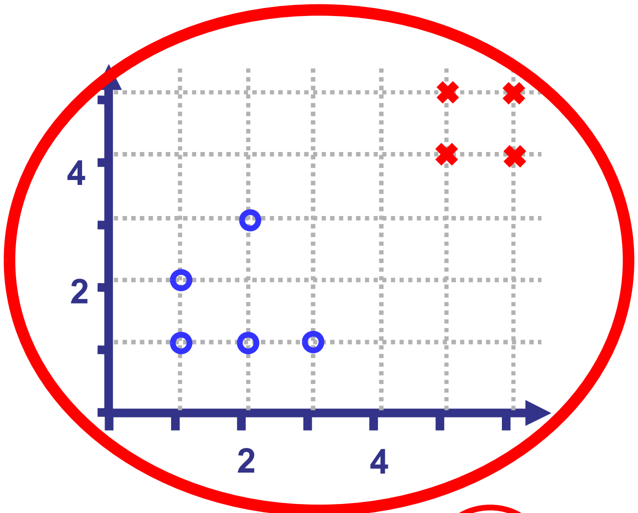
$$S_w = \begin{pmatrix} \begin{pmatrix} \sqrt{(5-5.5)^2+(4-4.5)^2}+\sqrt{(5-5.5)^2+(5-4.5)^2} \\ +\sqrt{(6-5.5)^2+(4-4.5)^2}+\sqrt{(6-5.5)^2+(5-4.5)^2} \end{pmatrix} \\ +\begin{pmatrix} \sqrt{(1-1.8)^2+(1-1.6)^2}+\sqrt{(1-1.8)^2+(2-1.6)^2} \\ +\sqrt{(2-1.8)^2+(1-1.6)^2}+\sqrt{(2-1.8)^2+(3-1.6)^2} \\ +\sqrt{(3-1.8)^2+(1-1.6)^2} \end{pmatrix} \end{pmatrix}$$

$$= 2.83 + 5.28 = 8.11$$

Clustering
Criterion Function: Example

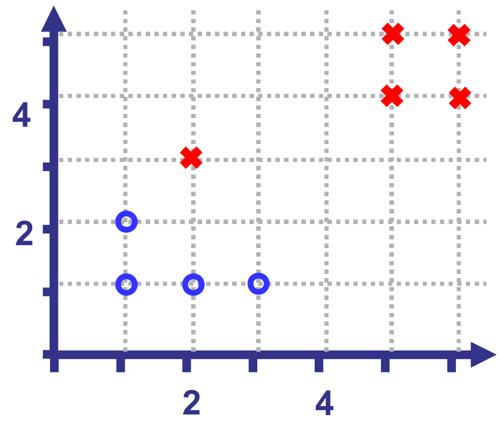
* Euclidean Distance is used

- Smaller S_A is preferred



$S_A = 2.83 + 5.28 = 8.11$

More reasonable result!



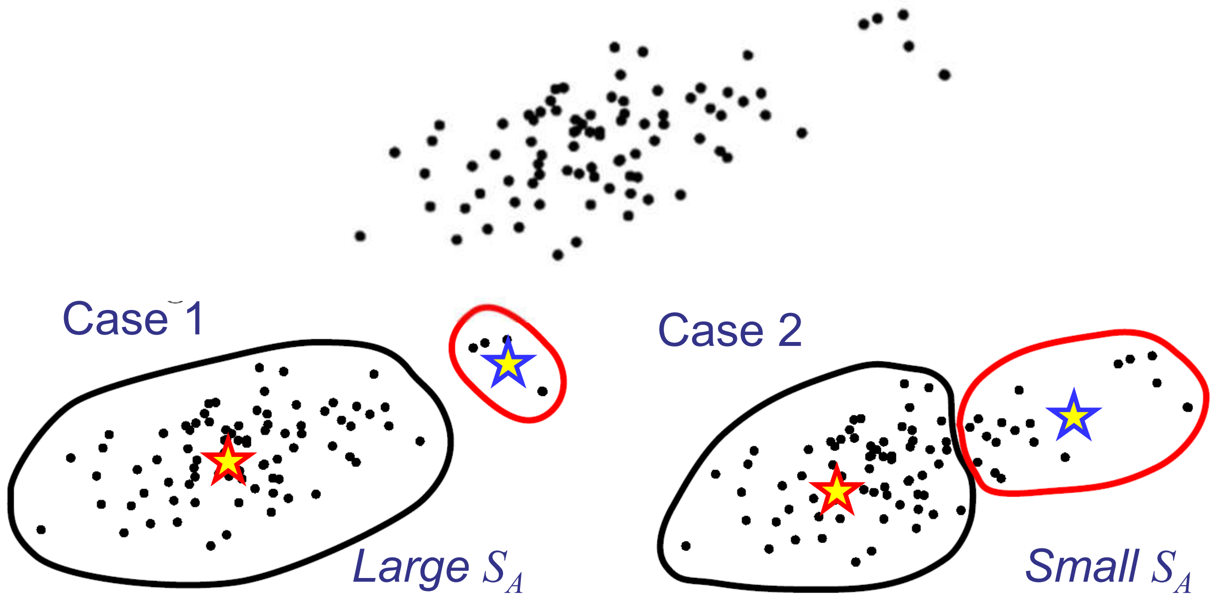
$S_A = 6.81 + 3.48 = 10.29$

Clustering
Criterion Function

- Is S_A (Intra-cluster scatter) a good criterion for all situations?

- How to separate the following samples into two clusters?

Criterion Function



Case 1 is more reasonable

However, it has a larger value of S_A due to the large cluster

Criterion Function

- S_A (Intra-cluster scatter) is
- Appropriate:
  - The clusters form compact groups
  - Equally sized clusters
- Not Appropriate
  - When natural groupings have very different sizes

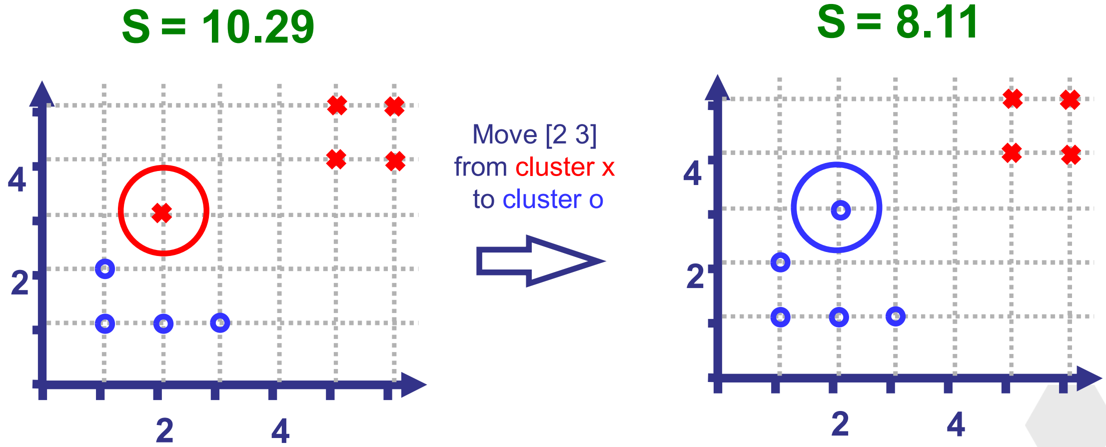
Clustering Algorithm

- Find the optimal clustering result
- Exhaustive search is impossible
  - ~C^n possible partitions
  - C: class number, n: sample number
- Methods:
  - Iterative Optimization Algorithm
    - K-means
  - Hierarchical Clustering
    - Bottom Up Approach
    - Top Down Approach

Iterative Optimization Algorithm

* Euclidean Distance is used

1. Find a reasonable initial cluster result
2. Move sample(s) from one cluster to another such that the objective function is improved the most
3. Goto 2 until stable



Clustering: Iterative Optimization Algorithm
K-means

- A well-known technique: **K-means**
  - Assume there are k clusters
  - Minimize Criterion Function (Intra-class scatter):

$$S = \sum_{i=1}^{k} \sum_{x \in D_i} \| x - \mu_i \|^2 \qquad \mu_i = \frac{1}{n_i} \sum_{x \in D_i} x$$

k : the number of cluster
n_i : the number of samples in cluster i
D_i : the set of samples in cluster i

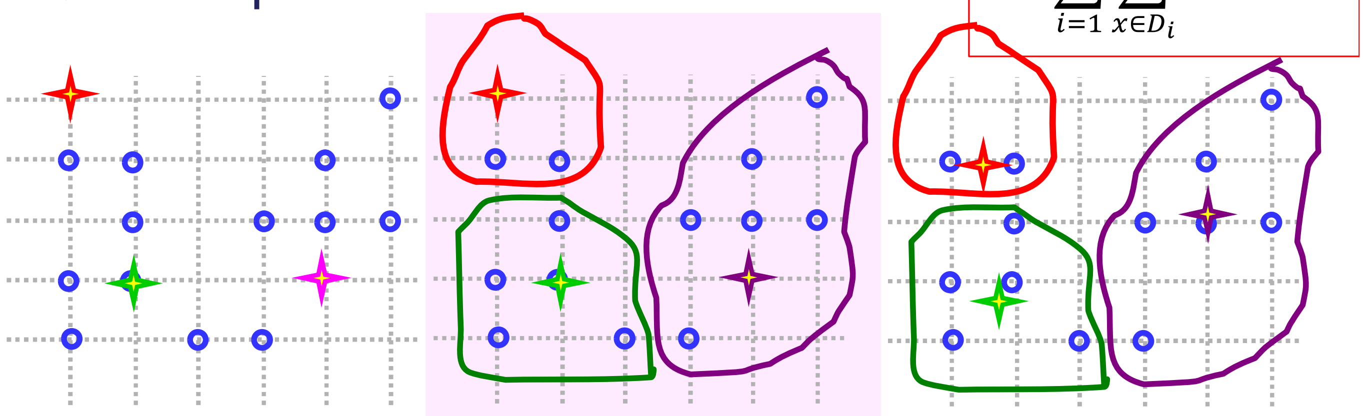
  - In each iteration, assign a sample to its closest cluster

Clustering: Iterative Optimization Algorithm
K-means

- Example: ***k*=3**

* Euclidean Distance is used

$$S = \sum_{i=1}^{k} \sum_{x \in D_i} \| x - \mu_i \|^2$$



1. Initialization
Randomly assign the center of each cluster

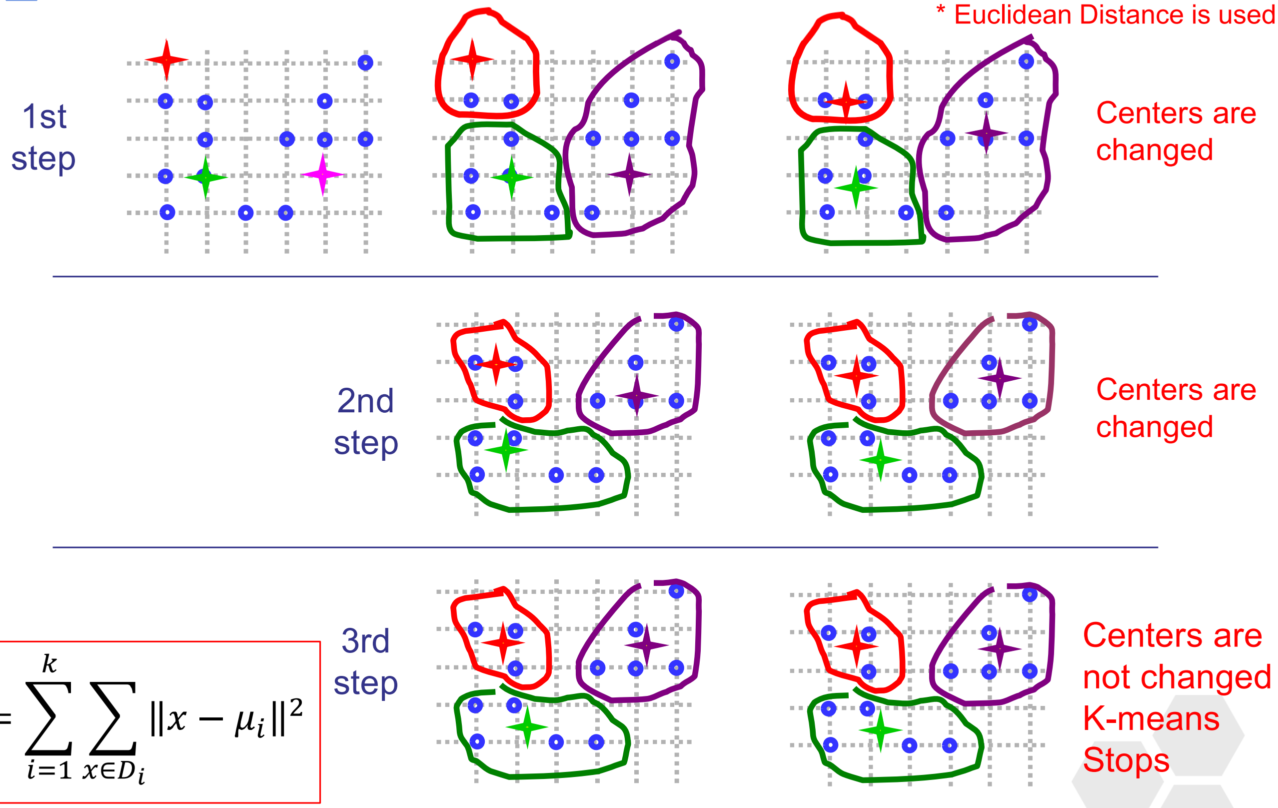
2. Assign Samples
Assign samples to closest center

3. Re-calculate mean
Compute the new means using new samples

Repeat until stable (no sample moves again)

K-means

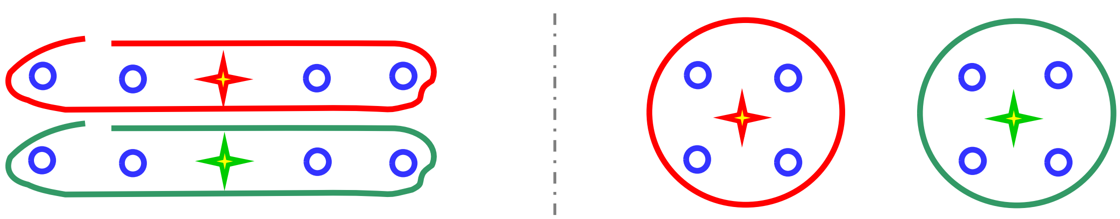
* Euclidean Distance is used



$$S = \sum_{i=1}^{k} \sum_{x \in D_i} \|x - \mu_i\|^2$$

K-means

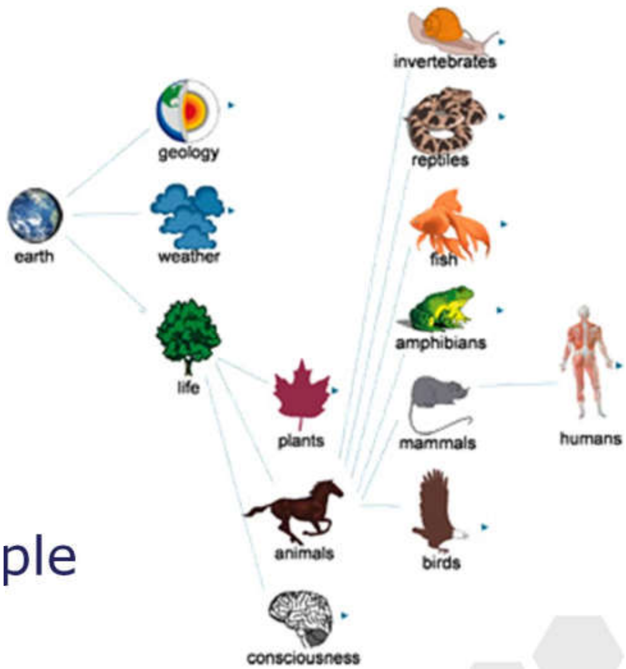
- Pros:
  - Optimize the objective function efficiently
  - Algorithm converges
- Cons:
  - May be trapped at local minimum
    (similar to gradient descent)



Trapped at Local Minimum   Global Minimum

Hierarchical Clustering

- Sometimes, clusters have subclusters, and so on
  - A cluster can further be broken down into smaller clusters
- Hierarchical cluster
- Taxonomy is an example



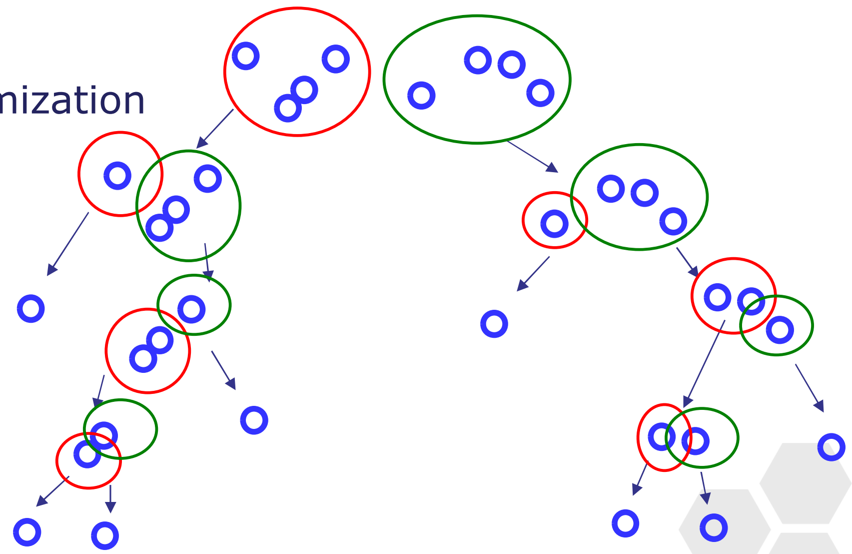
Hierarchical Clustering

- Two types:
  - **Top Down Approach**
    - Start with 1 cluster
      - One cluster contains all samples
    - Form hierarchy by splitting the most dissimilar clusters
  - **Bottom Up Approach**
    - Start with n clusters
      - Each cluster contains one sample
    - Form hierarchy by merging the most similar clusters
    - Not efficient if a large number of samples but a number of clusters is needed

Top Down Approach

- Start from one cluster
- Break down a cluster with more than one sample into two

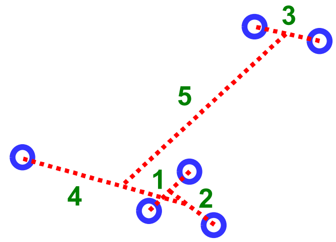
- Any Iterative Optimization Algorithm can be applied by setting c = 2

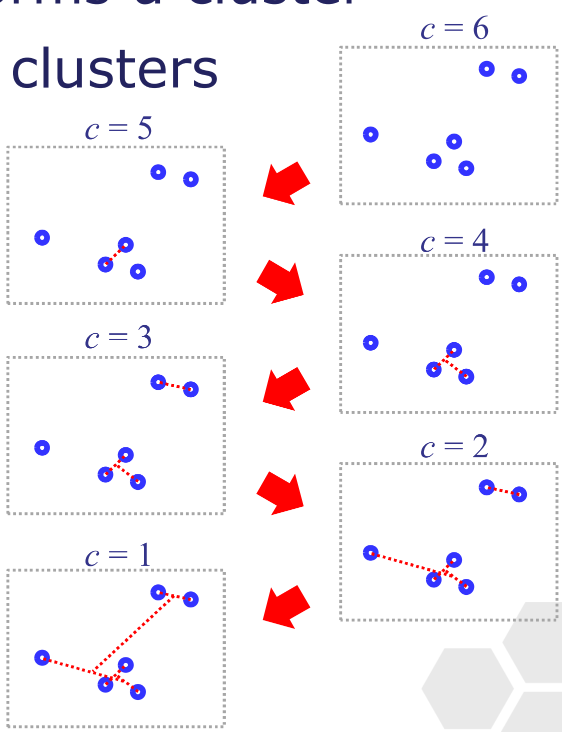


Bottom Up Approach

* Euclidean Distance is used

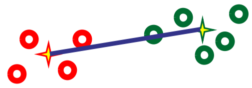
- Initially each sample forms a cluster
- Merge the nearest two clusters until one cluster left



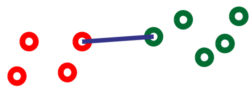


Bottom Up Approach

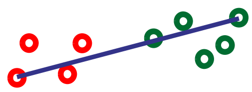
- How to calculate distance between clusters?



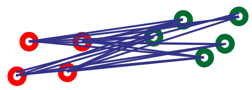
Mean Distance $d_{mean}(D_i, D_j) = \|m_i - m_i\|$



Minimum Distance $d_{min}(D_i, D_j) = \min_{x \in D_i, z \in D_j} \|x - z\|$



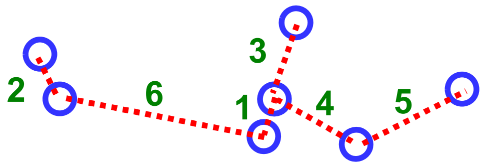
Maximum Distance $d_{max}(D_i, D_j) = \max_{x \in D_i, z \in D_j} \|x - z\|$



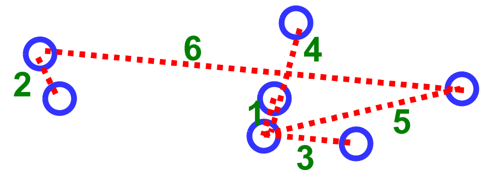
Average Distance $d_{avg}(D_i, D_j) = \frac{1}{n_i n_j} \sum_{x \in D_i} \sum_{x \in D_j} \|x - z\|$

Bottom Up Approach

- Single Linkage (Nearest-Neighbor)
  - Minimum Distance is used
  - Encourage growth of elongated clusters
- Complete Linkage (Farthest Neighbor)
  - Maximum Distance is used
  - Encourages compact clusters



Single Linkage
Min distance between points of each cluster

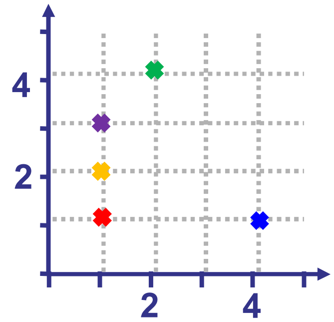


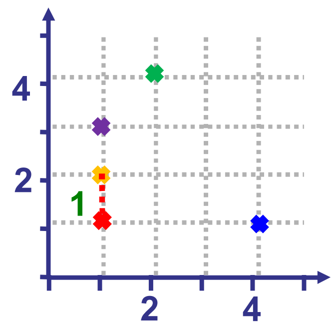
Complete Linkage
Max distance between points of each cluster

Single Linkage: Example 1/4

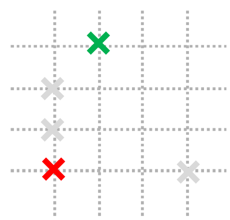
* Euclidean Distance is used

$c = 5$





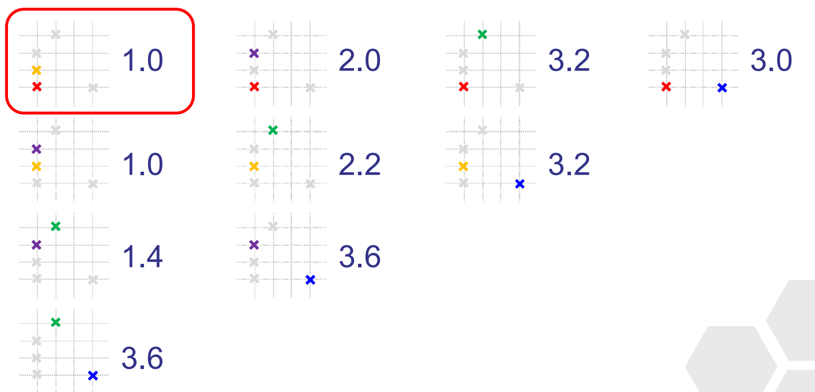
Example:



$$d\big((1,1),(2,4)\big) = \sqrt{(1-2)^2 + (1-4)^2}$$
$$= 3.2$$

$$\min(3.2) = 3.2$$

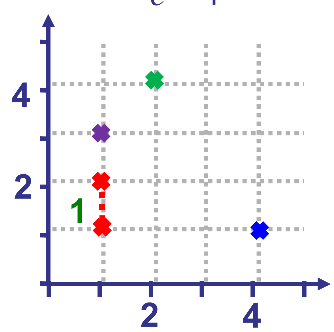
More than one choices

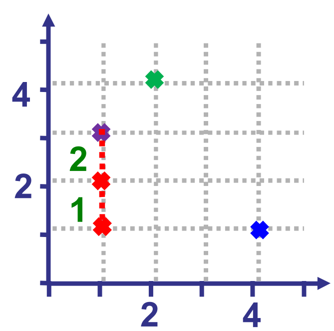


Single Linkage: Example 2/4

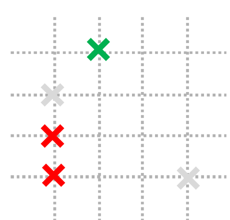
* Euclidean Distance is used

$c = 4$





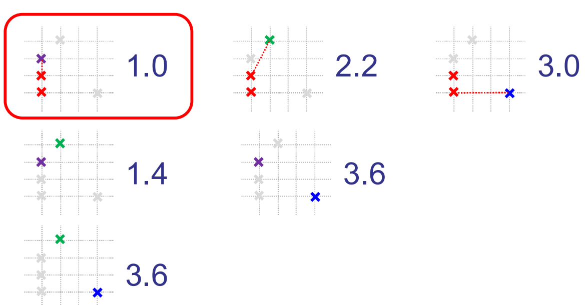
Example:



$$d\big((1,1),(2,4)\big) = \sqrt{(1-2)^2 + (1-4)^2} = 3.2$$

$$d\big((1,2),(2,4)\big) = \sqrt{(1-2)^2 + (2-4)^2} = 2.2$$

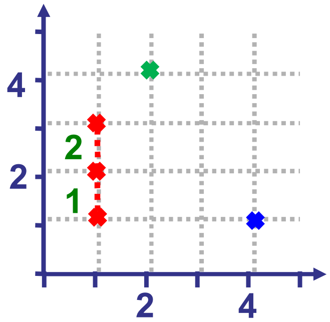
$$\min(3.2, 2.2) = 2.2$$

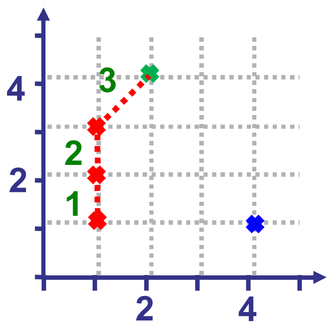


Single Linkage: Example 3/4

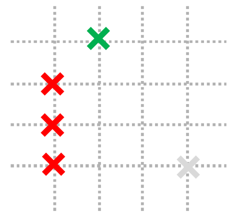
* Euclidean Distance is used

$c = 3$





Example:



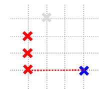
$$d\big((1,1),(2,4)\big) = \sqrt{(1-2)^2 + (1-4)^2} = 3.2$$

$$d\big((1,2),(2,4)\big) = \sqrt{(1-2)^2 + (2-4)^2} = 2.2$$

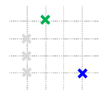
$$d\big((1,3),(2,4)\big) = \sqrt{(1-2)^2 + (3-4)^2} = 1.4$$

$$\min(3.2, 2.2, 1.4) = 1.4$$

1.4



3.0

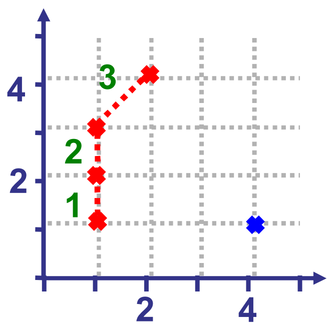


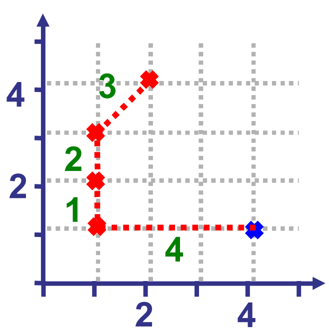
3.6

Single Linkage: Example 4/4

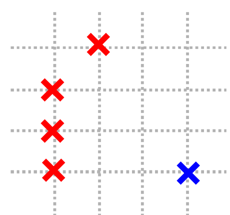
* Euclidean Distance is used

$c = 2$





Example:



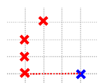
$$d\big((1,1),(4,1)\big) = \sqrt{(1-4)^2 + (1-1)^2} = 3.0$$

$$d\big((1,2),(4,1)\big) = \sqrt{(1-4)^2 + (2-1)^2} = 3.2$$

$$d\big((1,3),(4,1)\big) = \sqrt{(1-4)^2 + (3-1)^2} = 3.6$$

$$d\big((2,4),(4,1)\big) = \sqrt{(2-4)^2 + (4-1)^2} = 3.6$$

$$\min(3.2, 2.2, 1.4) = 3.0$$



3.0

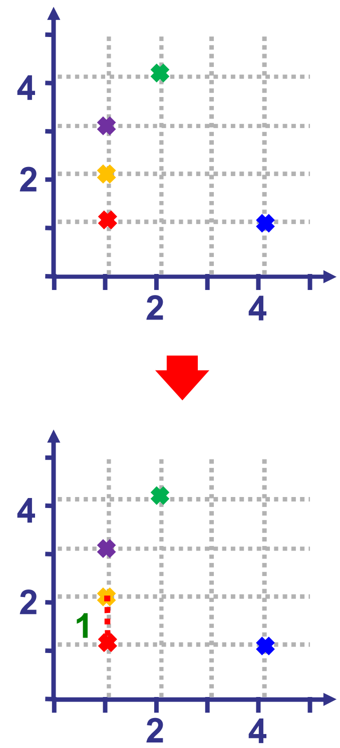
* This step is unnecessary as only one candidate

Clustering: Hierarchical Clustering: Bottom Up Approach

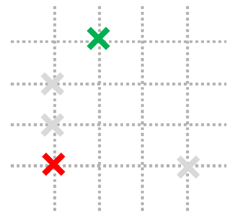
Complete Linkage: Example 1/4

* Euclidean Distance is used

$c = 5$



Example:

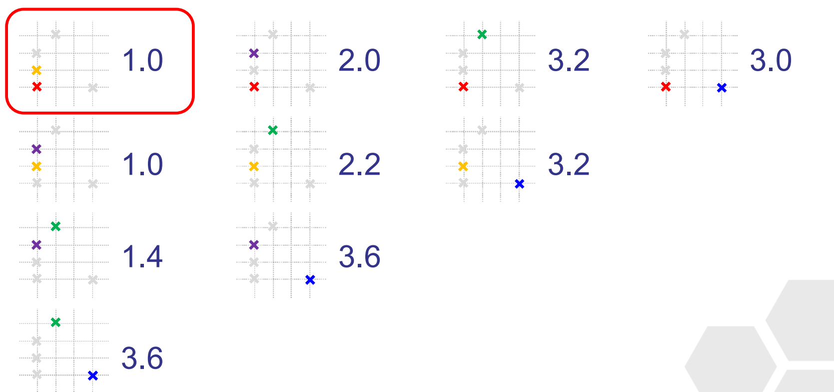


$$d\big((1,1),(2,4)\big) = \sqrt{(1-2)^2 + (1-4)^2} = 3.2$$

$$\max(3.2) = 3.2$$

* This step is the same as Single Linkage since distance measure of clusters with one point is the same

More than one choices

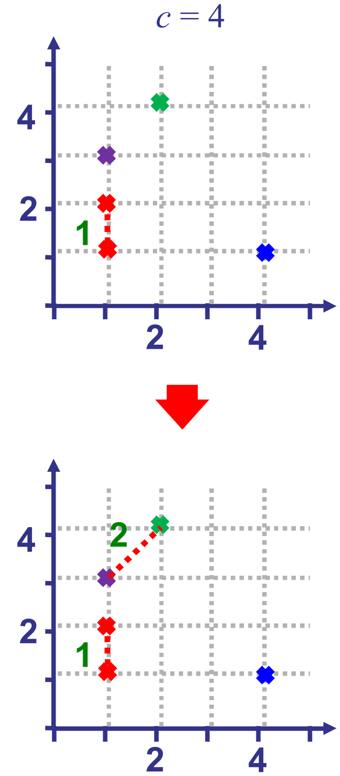


Clustering: Hierarchical Clustering: Bottom Up Approach

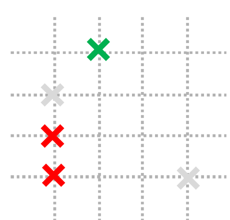
Complete Linkage: Example 2/4

* Euclidean Distance is used

$c = 4$



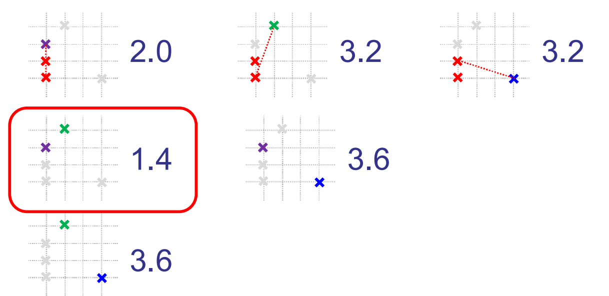
Example:



$$d\big((1,1),(2,4)\big) = \sqrt{(1-2)^2 + (1-4)^2} = 3.2$$

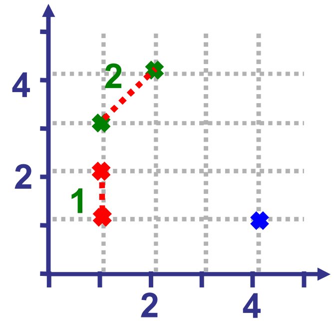
$$d\big((1,2),(2,4)\big) = \sqrt{(1-2)^2 + (2-4)^2} = 2.2$$

$$\max(3.2, 2.2) = 3.2$$



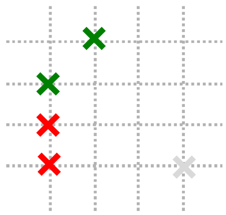
Complete Linkage: Example 3/4

$c = 3$

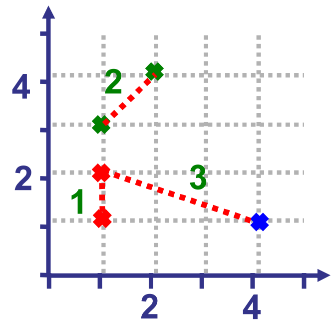


* Euclidean Distance is used

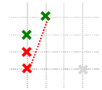
Example:

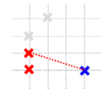


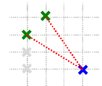
$$d((1,1),(1,3)) = \sqrt{(1-1)^2 + (1-3)^2} = 2.0$$
$$d((1,1),(2,4)) = \sqrt{(1-2)^2 + (1-4)^2} = 3.2$$
$$d((1,2),(1,3)) = \sqrt{(1-1)^2 + (2-3)^2} = 1.0$$
$$d((1,2),(2,4)) = \sqrt{(1-2)^2 + (2-4)^2} = 2.2$$
$$\max(2.0, 3.2, 1.0, 2.2) = 3.2$$



More than one choices

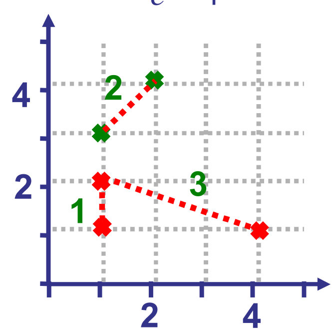

3.2


3.2


3.6

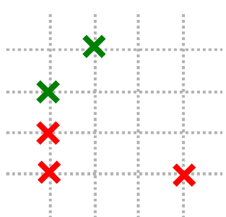
Complete Linkage: Example 4/4

$c = 4$

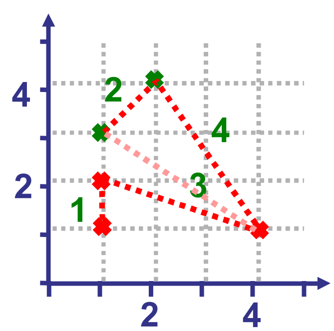


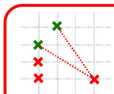
* Euclidean Distance is used

Example:



$$d((1,1),(1,3)) = \sqrt{(1-1)^2 + (1-3)^2} = 2.0$$
$$d((1,1),(2,4)) = \sqrt{(1-2)^2 + (1-4)^2} = 3.2$$
$$d((1,2),(1,3)) = \sqrt{(1-1)^2 + (2-3)^2} = 1.0$$
$$d((1,2),(2,4)) = \sqrt{(1-2)^2 + (2-4)^2} = 2.2$$
$$d((4,1),(1,3)) = \sqrt{(4-1)^2 + (1-3)^2} = 3.6$$
$$d((4,1),(2,4)) = \sqrt{(4-2)^2 + (1-4)^2} = 3.6$$
$$\max(2.0, 3.2, 1.0, 2.2, 3.6, 3.6) = 3.6$$

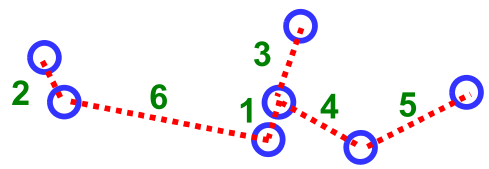



3.6

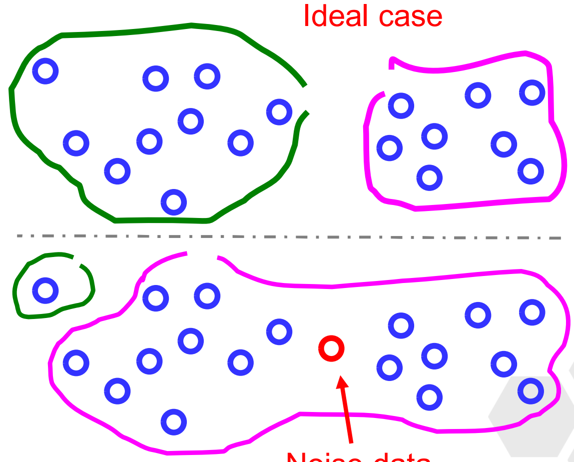
* This step is unnecessary as only one candidate

Bottom Up Approach

- Single Linkage (Nearest-Neighbor)
  - Minimum Distance is used
  - Encourage growth of elongated clusters
  - Disadvantage: Sensitive to noise

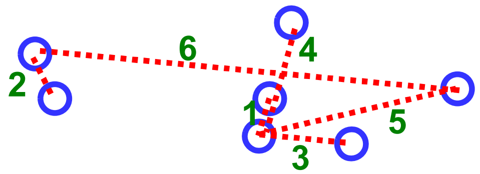


Min distance between points of each cluster



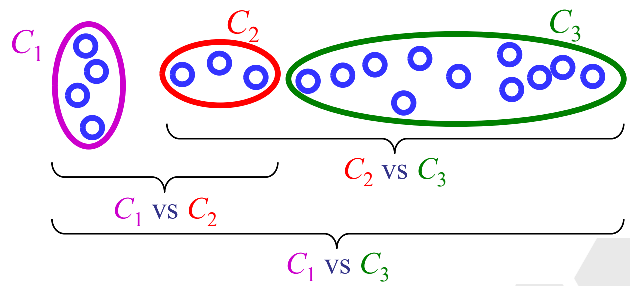
Bottom Up Approach

- Complete Linkage (Farthest Neighbor)
  - Maximum Distance is used
  - Encourages compact clusters
  - Disadvantage: Does not work well if elongated clusters present



Max distance between points of each cluster

Ideally, C_2 and C_3 should be merged



However, C_1 and C_2 will be merged

Bottom Up Approach

- Minimum and maximum distance are noise sensitive (especially, minimum)
- More robust result to outlier when average or mean are used

$$d_{avg}(D_i, D_j) = \frac{1}{n_i n_j} \sum_{x \in D_i} \sum_{x \in D_j} \|x - z\|$$

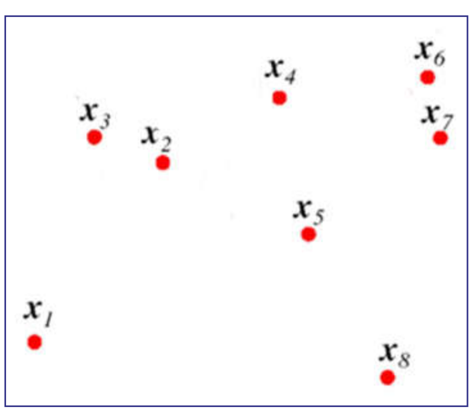
$$d_{mean}(D_i, D_j) = \|m_i - m_i\|$$

- Mean is less time consumed than Average distance

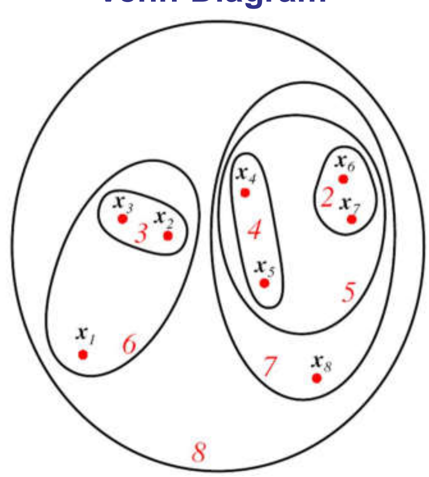
Venn

- Venn diagram can show hierarchical clustering
- No quantitative information is provided

Sample points



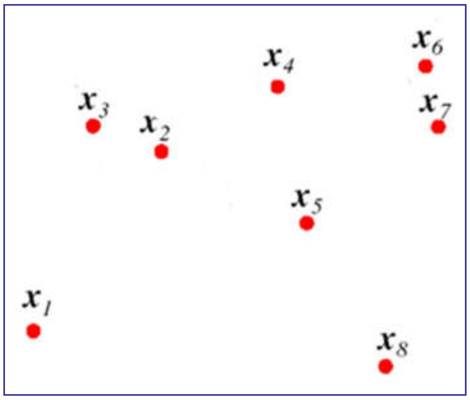
Venn Diagram



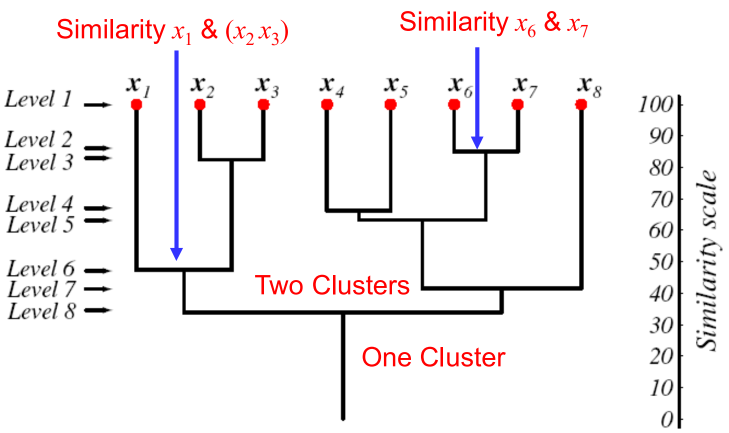
Clustering: Hierarchical Clustering
Dendogram

- Dendogram is another way to represent a hierarchical clustering
- Able to indicate the similarity value

Sample points

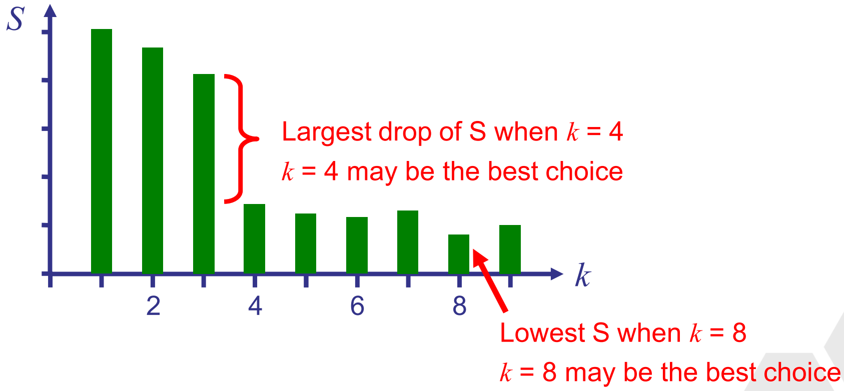


Dendogram



Clustering
Number of Clusters

- How to decide **the number of clusters**?
- Possible solution:
  - Try a range of k and see which one has the lowest or largest drop criterion value (S)
  - Example:



Curse of Dimensionality

- Real data usually have plenty of features
  - E.g., documents, images…
- Huge number of features causes problems
  - Sparsity
  - Complexity (storage and process)

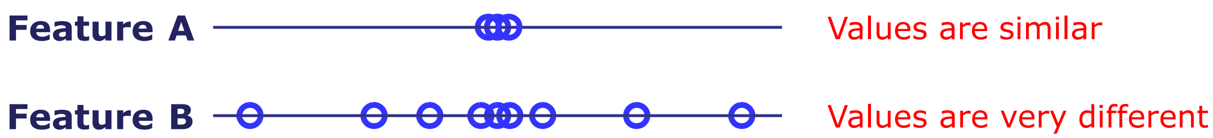
Curse of Dimensionality

- Can the data be described with fewer dimensions, without losing much of its original meaning?

- **Dimensionality Reduction**
  - Not just reduce the amount of data
  - Often brings out the useful part of the data

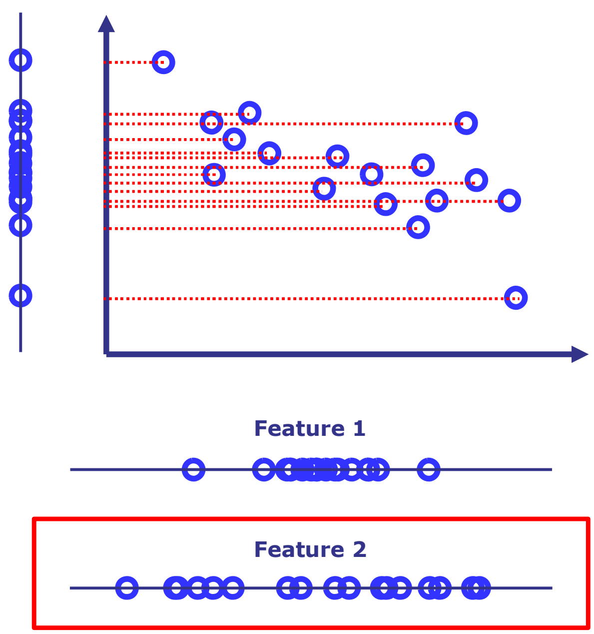
Feature Reduction

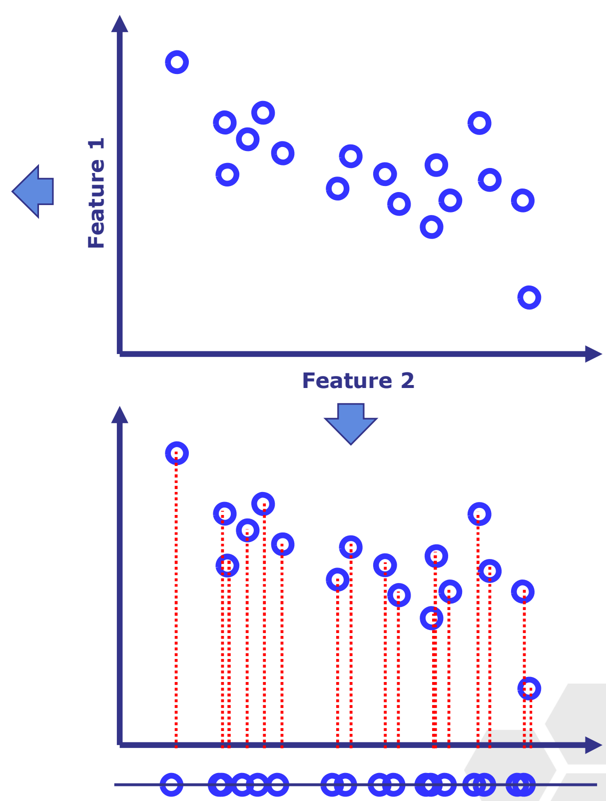
- For unsupervised learning, which feature is more useful to represent a dataset?



- A feature with different values provides more information
  - Variance is one of measures

Feature Reduction

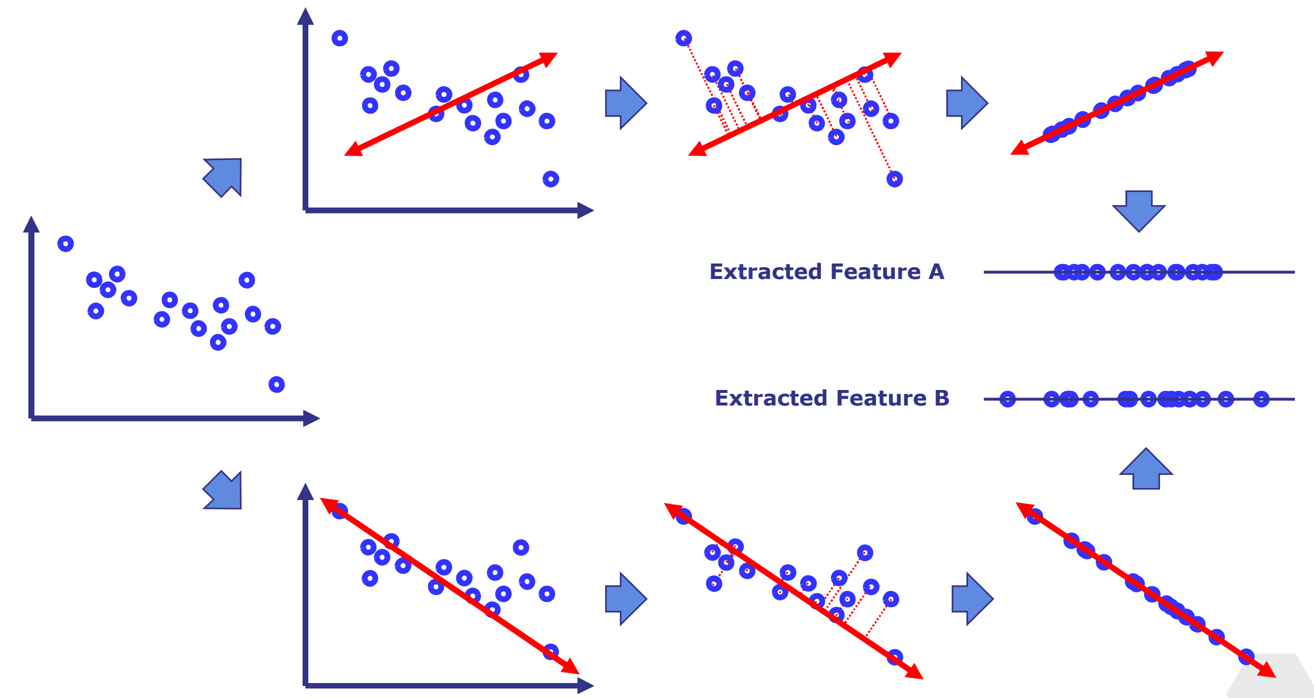




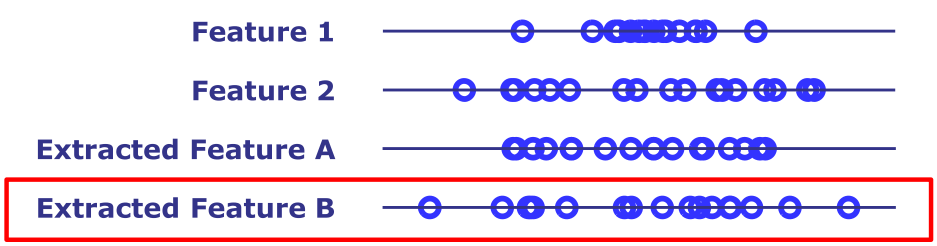
Feature 2 is better (large variance)

Any better choice?

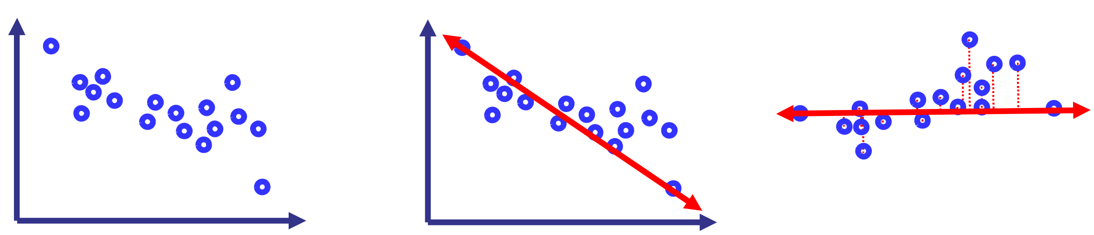
Feature Reduction



Feature Reduction



- Extracted Feature B is the best for representing the data

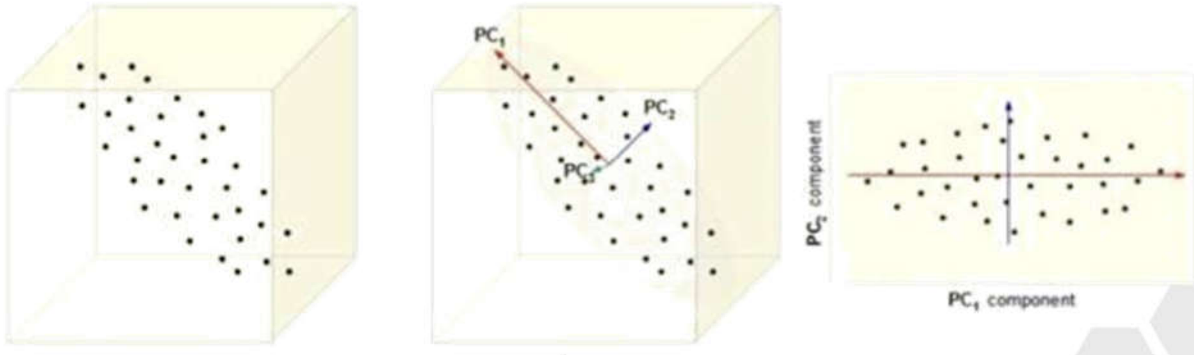


Principal Components Analysis

- PCA reduces data by geometrically projecting them onto lower dimensions called principal components (PCs)
- Project a dataset into a new set of features such that:
  - The features have zero covariance to each other (they are orthogonal)
  - Each feature captures the most remaining variance in the data, while orthogonal to the existing feature

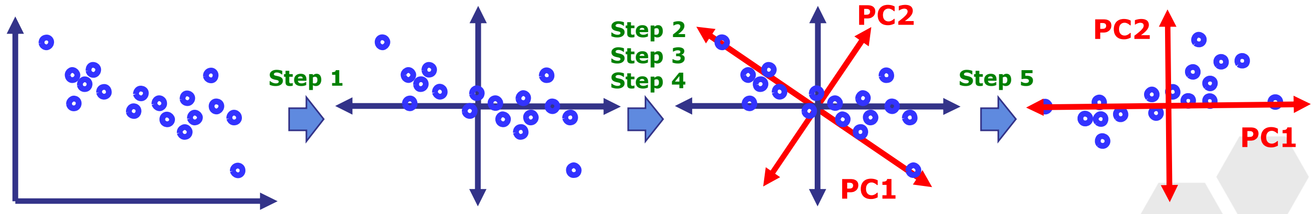
Principal Components Analysis

- **First** principal component (PC) is the direction of greatest variability (variance) in the data
- **Second** PC is the next orthogonal (uncorrelated) direction of greatest variability
- And so on..



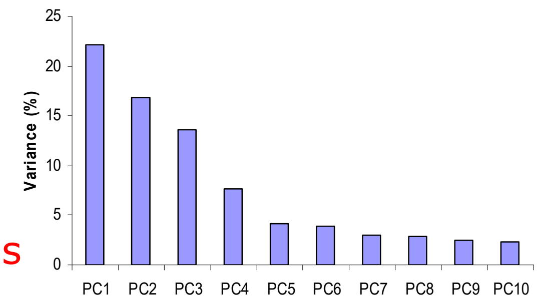
Principal Components Analysis

1. **Standardize** the dataset
2. Calculate the **covariance matrix** for the features in the dataset
3. Calculate the **eigenvalues** and **eigenvectors** for the covariance matrix.
4. **Pick top k eigenvalues** and form their eigenvectors
5. Transform the original matrix



PC number Determination

- A feature with small eigenvalue contains small information
  - n dimensions in original data
  - Choose only the first p eigenvectors, based on their eigenvalues ($p < n$)
  - Final data has only p dimensions



- How to determine k?

Principal Components Analysis
PC number Determination

- Determine by **projection error**

$$\frac{\frac{1}{m}\sum_{i=1}^{n}\left(x^{(i)}-z_k\left(x^{(i)}\right)\right)^2}{\frac{1}{m}\sum_{i=1}^{n}\left(x^{(i)}\right)^2}\le\varepsilon$$

where:

- $x^{(i)}$: the ith sample
- $z_k\left(x^{(i)}\right)$: the ith sample with k PCs
- ε : allowed error

- Determine by **variation ratio**

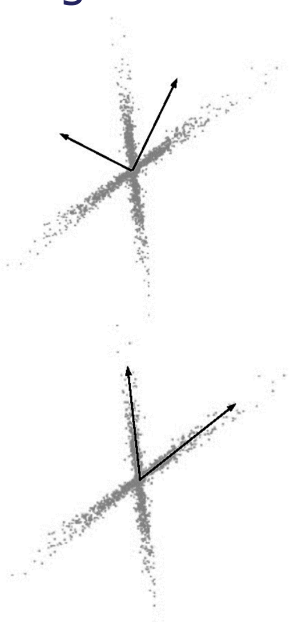
$$\frac{\sum_{j=1}^{k}\sigma_j^2}{\sum_{j=1}^{n}\sigma_j^2}\approx r$$

where:

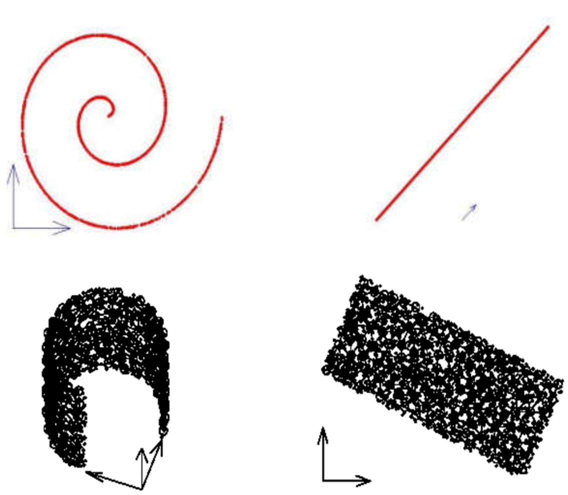
- σ_j : the jth variance in descending order
- r : expected ratio (e.g. 85%)

Principal Components Analysis
Limitation

- Orthogonal Feature

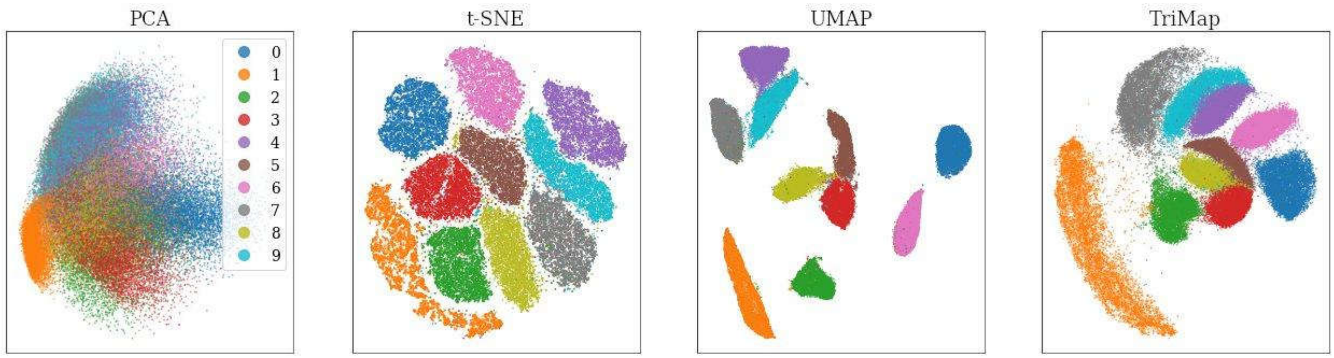


- Non-linear projection



Principal Components Analysis

- Other feature extraction methods



References

- http://users.umiacs.umd.edu/~jbg/teaching/INST_414/