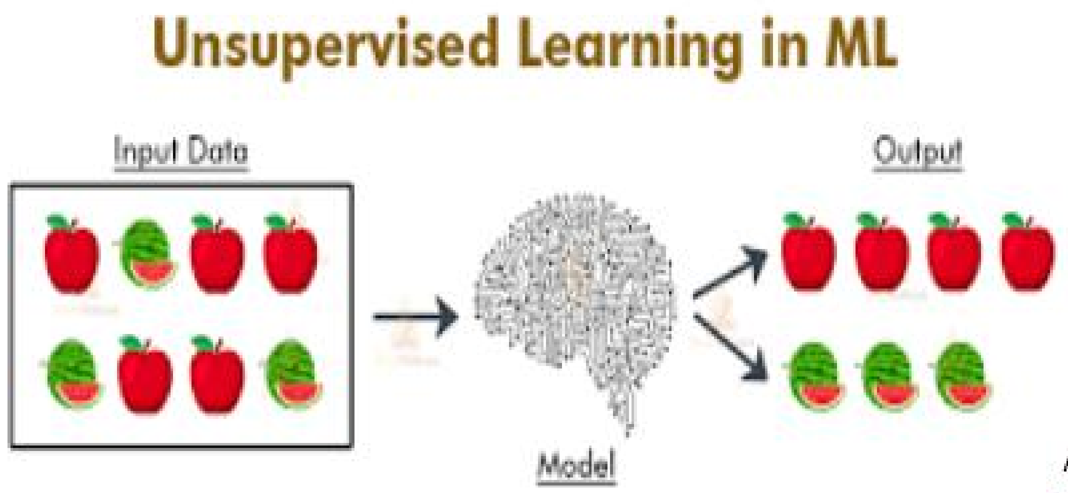
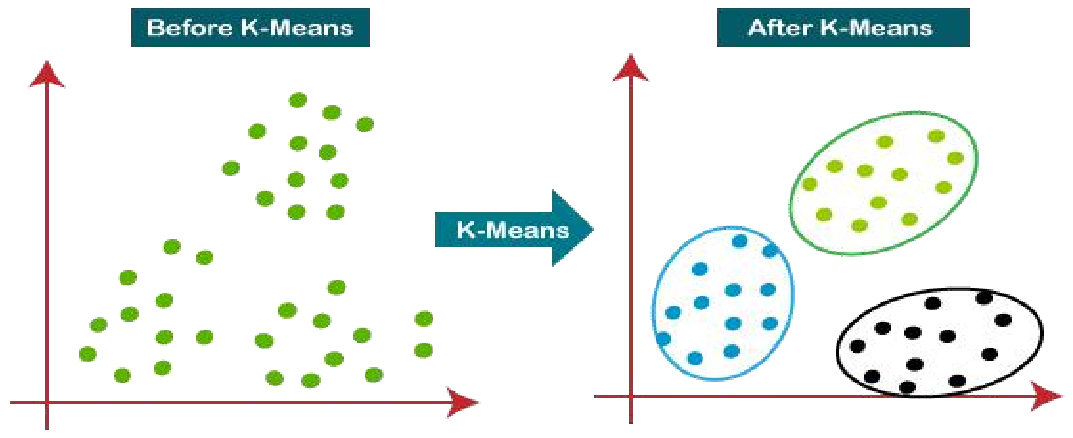
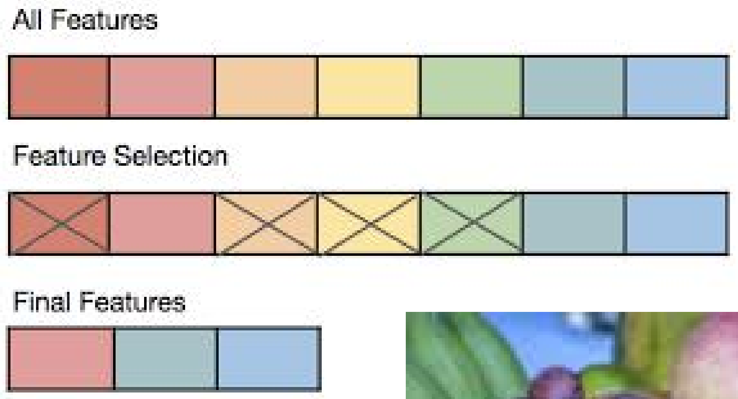
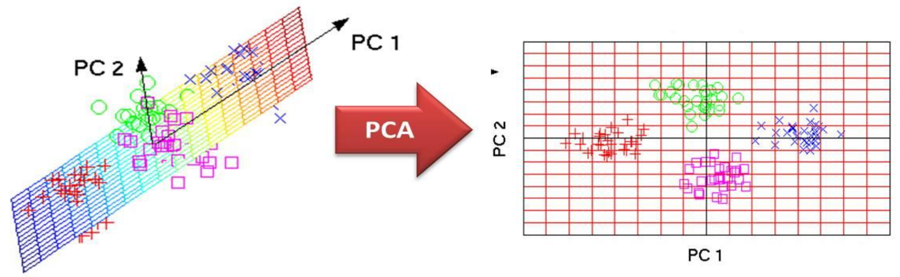
Chapter-6
Unsupervised Learning

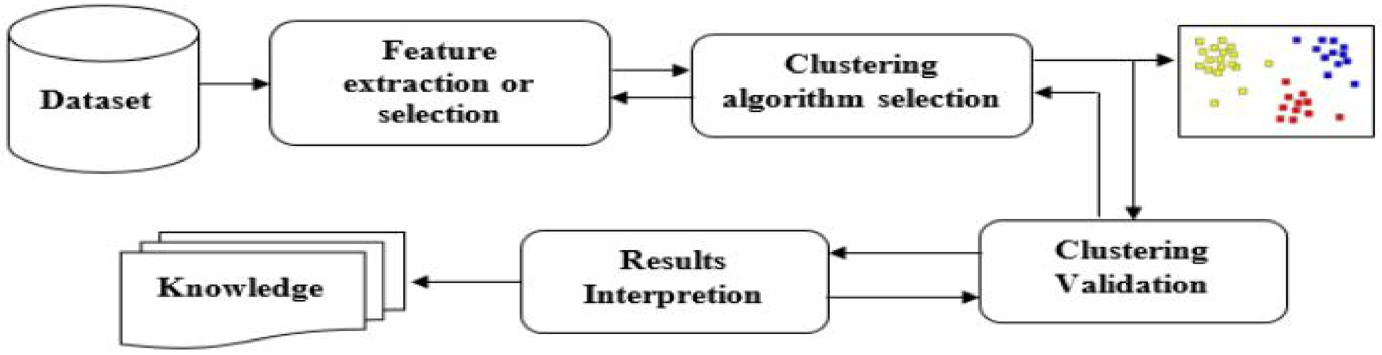




Dimensionality Reduction & Principal Component Analysis

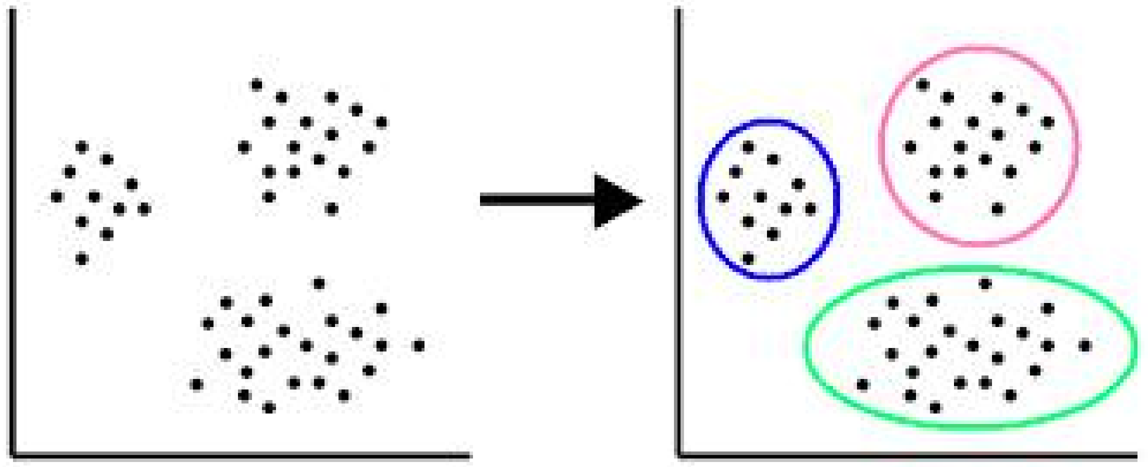





Dataset
Feature extraction or selection
Clustering algorithm selection
Clustering Validation
Results Interpretion
Knowledge

Unsupervised learning

- Unsupervised learning aims to find the underlying structure or the distribution of data. We want to explore the data to find some intrinsic structures in them.



- Learn from inputs $x_1, \ldots, x_n \in R^d$ without any labels $y_1, \ldots, y_n$.

*No predefined classes - Clustering

What is Cluster Analysis?

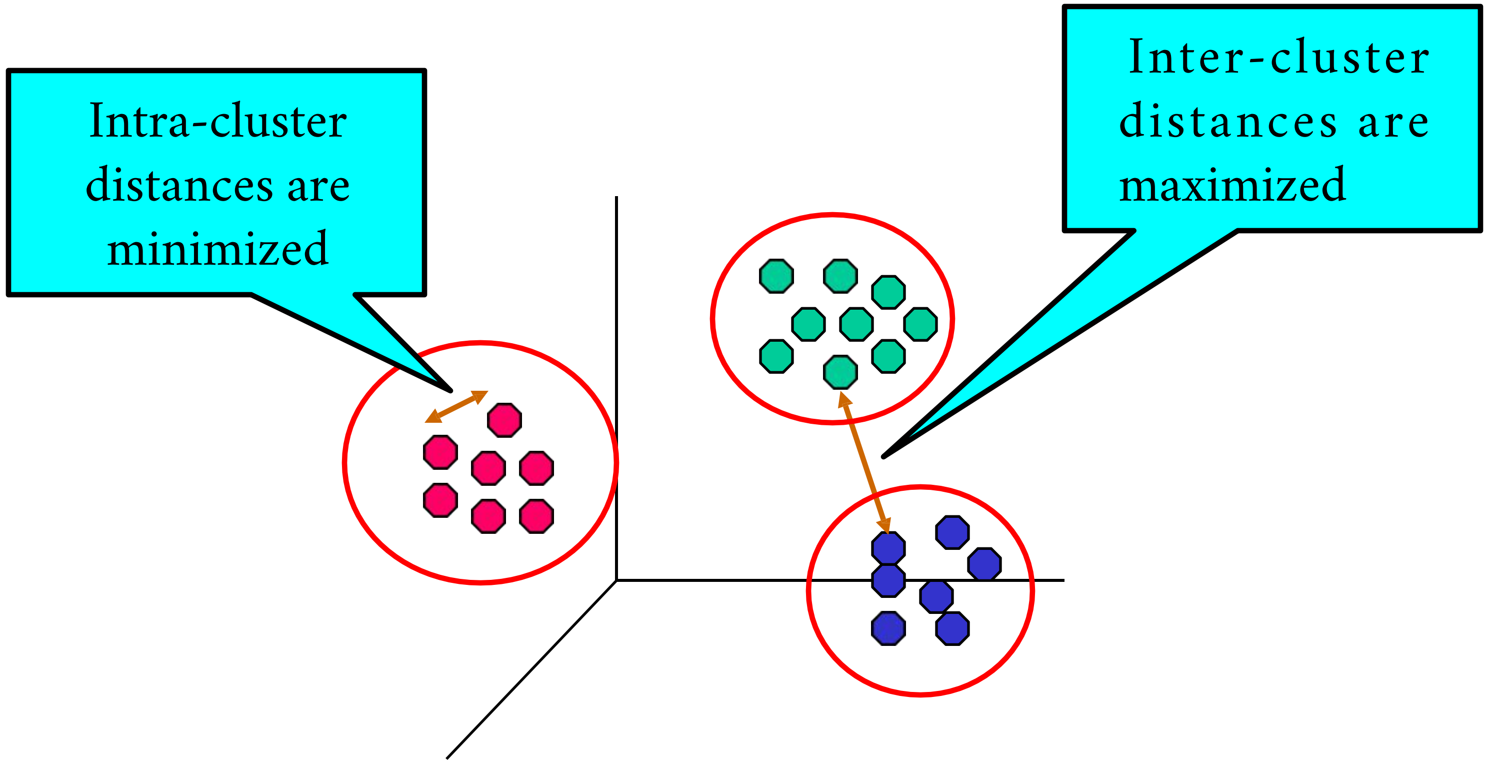
- **Cluster: a collection of data objects**
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- **Cluster analysis**

  Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters

Definition. **Given a database D =$\{t_1,t_2,\ldots,t_n\}$ of tuples and an integer value *k*, the clustering problem is to define mapping f:D$\rightarrow\{1,\ldots,k\}$ where each t_i is assigned to one cluster K_j, $1 \leq j \leq$ k. A cluster, K_j, containing precisely those tuples mapped to it; that is, $K_j = \{t_i \mid f(t_i) = K_j, 1 \leq i \leq n$, and $t_i \in D\}$**

What is Cluster Analysis?

Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups


Intra-cluster distances are minimized
Inter-cluster distances are maximized

Clustering-Applications

- **Recommender systems**: organizing products and customers into groups that are similar

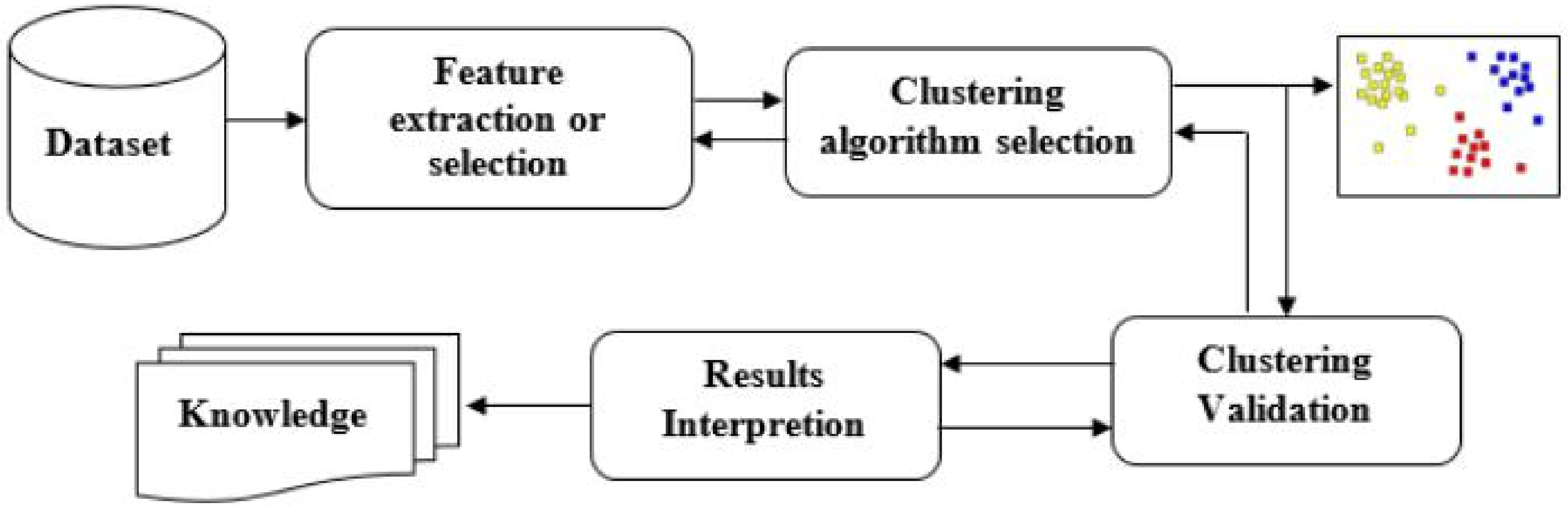
- **Social networks**: cluster users into groups that have similar interests/preferences

- **WWW document classification**: organize webpages (e.g., news articles) into clusters with similar content (sports, ecnomonics, ...)

Das niu Update ist da!
Alle für Sie relevanten News in einer iPad App!
Wählen Sie Ihre individuellen Inhalte aus über 40 verschiedenen Zeitungen und Magazinen.

Procedures of Cluster Analysis

- A typical cluster analysis consists of four steps


Dataset
Feature extraction or selection
Clustering algorithm selection
Clustering Validation
Results Interpretion
Knowledge

Procedures of Cluster Analysis

Feature extraction

- Feature extraction is the process of using one or more transformations of the input features to generate new principal features.
- Feature extraction can be elaborated in the context of dimensionality reduction and data visualization.

Procedures of Cluster Analysis

Clustering algorithm design or selection

- The clustering step is usually combined with the selection of a corresponding proximity (i.e, the closeness or distance) measure and
- the construction of a clustering criterion function (i.e, finding the optimal partitioning of a data set according to some criterion function or agorithm).
- **Proximity measures** refer to the measures of Similarity and Dissimilarity.
- **Clustering criterion**: once a proximity measure is chosen, the construction of a clustering criterion function makes the partition of clusters as an optimization problem.

Procedures of Cluster Analysis

Cluster validation (Assessment of results)

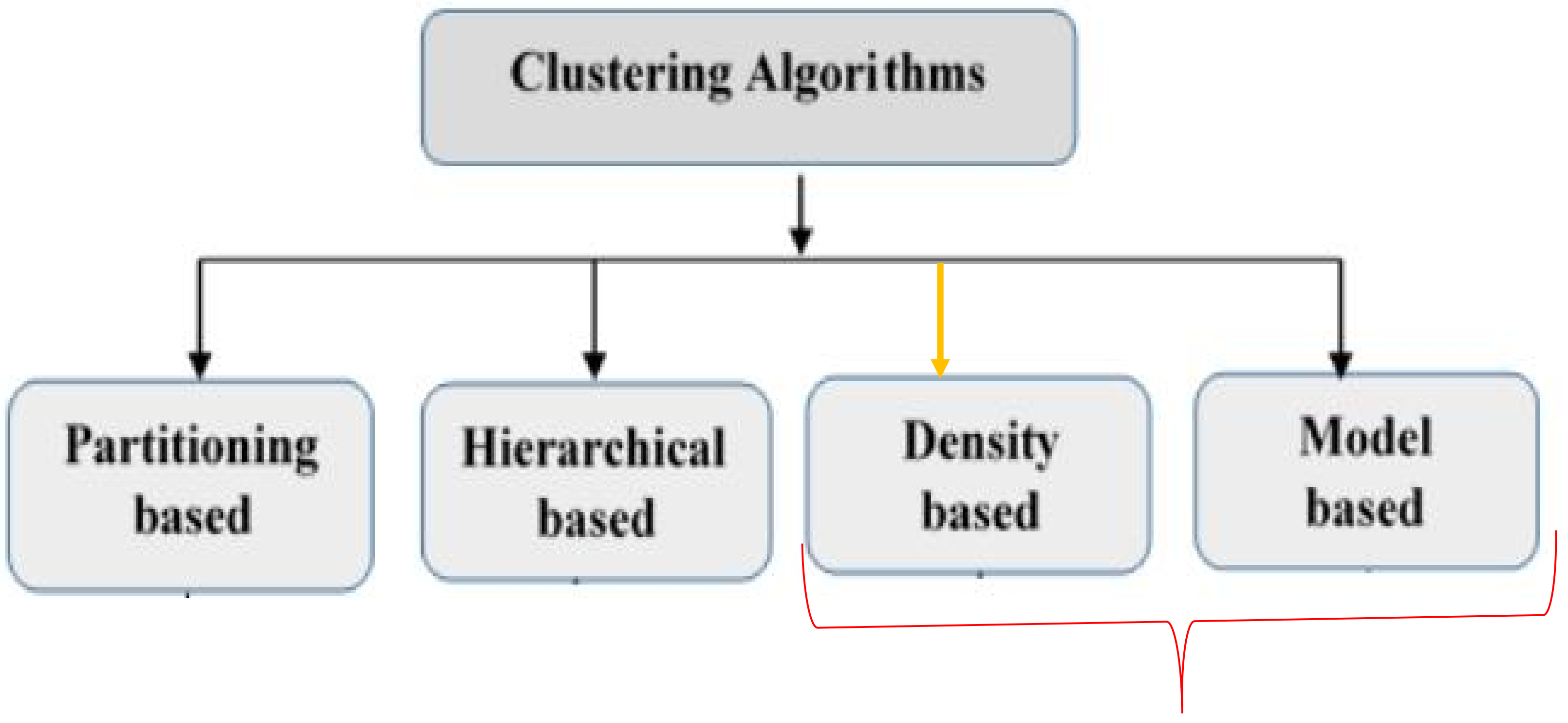
- Given a dataset, any clustering algorithm can usually generate clusters, no matter whether the structure exists or not.
- Thus, effective evaluation standards and criteria are essential to provide the users with a degree of confidence for the clustering results.
- These evaluation methods should be objective and have no preference for any algorithm.

Procedures of Cluster Analysis

Results Interpretation

- The final target of clustering is to supply users with meaningful perceptions from the original dataset, with the aim that they can effectively solve the problems faced.
- Experts in different domains interpret the data groupings.
- Further analyses, even experiments, may be required to assure the reliability of extracted knowledge.

Categories of clustering Algorithms



will not be discussed in this chapter

Partitioning method

Partitioning method: Construct a partition of a database **D** of **n** objects into a set of **k** clusters

Various approaches have been proposed and some of them are:

K-mean Approach

K-medoid Approach, CLARA (Custering LARge Application) and CLARANS (CLustering Algorithm based on RANdomized Search) [will not be coverd here]

K-means clustering

- K-means is a partitional clustering algorithm
- Let the set of data points (or instances) D be

  $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$,

  where $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ir})$ is a vector in a real-valued space $X \in R^r$, and r is the number of attributes (dimensions) in the data.
- The k-means algorithm partitions the given data into k clusters.
  - Each cluster has a cluster **center**, called **centroid**.
  - k is specified by the user

K-means algorithm

Given *k*, the *k-means* algorithm is implemented in 4 steps:

- Select K centroid (Can be K values randomly, or K data points randomly)
- Partition objects into *k* subsets. An object will be clustered into class J if it has the smallest distance with this class mean compared to the distance with the other class mean
- Compute the new centroids of the clusters of the current partition. The centroid of the jth cluster is the center (mean point) of the data point whose cluster index is found to be the center of class j in the above step.
- Go back to Step 3, stop when the process converge.

K-means algorithm

```
1:  function KMEANS(parameter $k$, inputs $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$)
2:      initialize cluster centers $\mathbf{c}_1, \ldots, \mathbf{c}_k$
3:      repeat
4:          for $i = 1 : n$ do
5:              label the input $\mathbf{x}_i$ as belonging to the nearest cluster,
                    $y_i := \arg\min_{j=1,\ldots,k} \left\|\mathbf{x}_i - \mathbf{c}_j\right\|^2$
6:          end for
7:          for $j = 1 : k$ do
8:              compute cluster center $\mathbf{c}_j$ as the mean of all inputs of the $j$th cluster,
                    $\mathbf{c}_j := \mathrm{mean}(\{\mathbf{x}_i : y_i = j\})$
9:          end for
10:     until convergence criterion is met
11:     return cluster centers $c_1, \ldots, c_k$
12: end function
```

K-Means Clustering – Simple Example

Example: Suppose we are given the following items to cluster: {2, 4, 10, 12, 3, 20,30,11,25} and suppose *k*=2.

Initially assign the means to the first two values: m_1=2 and m_2 =4.

Using Euclidean distance, find initial clusters K_1 = {2, 3} and K_2={4, 10, 12, 20, 30, 11, 25}

- For instance, dist(2,10) = $\sqrt{(10-2)^2}$ = 8; dist(4,10) = $\sqrt{(10-4)^2}$ = 6, hence item 10 will be assigned to cluster K_2
- But, note that, the item 3 is equally close to both means, so we arbitrarily choose K_1.

Now compute new means m_1= 2.5 and m_2=16

Assign items closer to m_1 and m_2, to get clusters K_1 = {2,3,4} and K_2 = { 10, 12, 20, 30, 11, 25}

K-Means Clustering – Simple Example

- Continue in this fashion till the values of K_1 and K_2 do not change as shown in the table.
- Since there is no change in K_1 and K_2 (last two rows of the table), the answer is : K_1= { 2, 3, 4, 10, 11, 12} and K_2 ={20, 30, 25}

m_1	m_2	K_1	K_2
3	18	{2,3,4,10}	{12,20,30,11,25
4.75	19.6	{2,3,4,10,11, 12}	20,30,25}
7	25	{2,3,4,10,11,12	20,30,25}

*No (or minimum) re-assignments of data points to different clusters, or no change of centroids

Strengths of k-means

- Strengths:
  - Simple: easy to understand and to implement
  - Efficient: Time complexity: $O(tkn)$, where n is the number of data points, k is the number of clusters, and t is the number of iterations.
  - Since both k and t are small. k-means is considered a linear algorithm.
- K-means is the most popular clustering algorithm.

Weaknesses of k-means

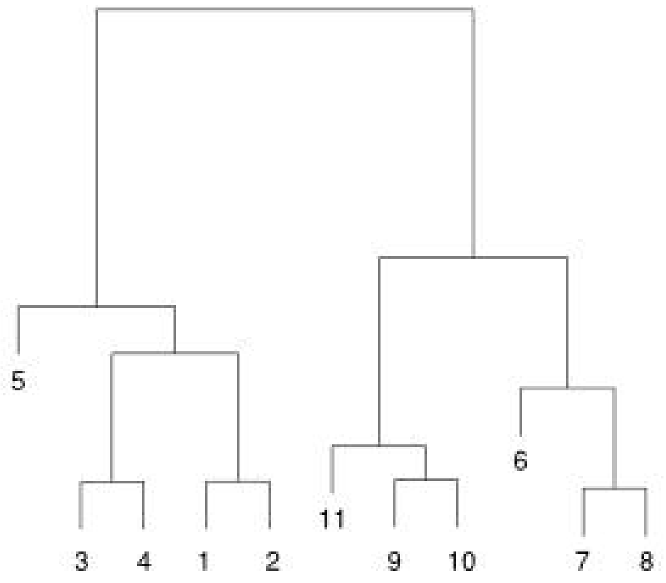
- The algorithm is only applicable if the mean is defined.
  - For categorical data, ***k*-mode** - the centroid is represented by most frequent values.
- The user needs to specify *k*.
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are very far away from other data points.
  - Outliers could be errors in the data recording or some special data points with very different values.

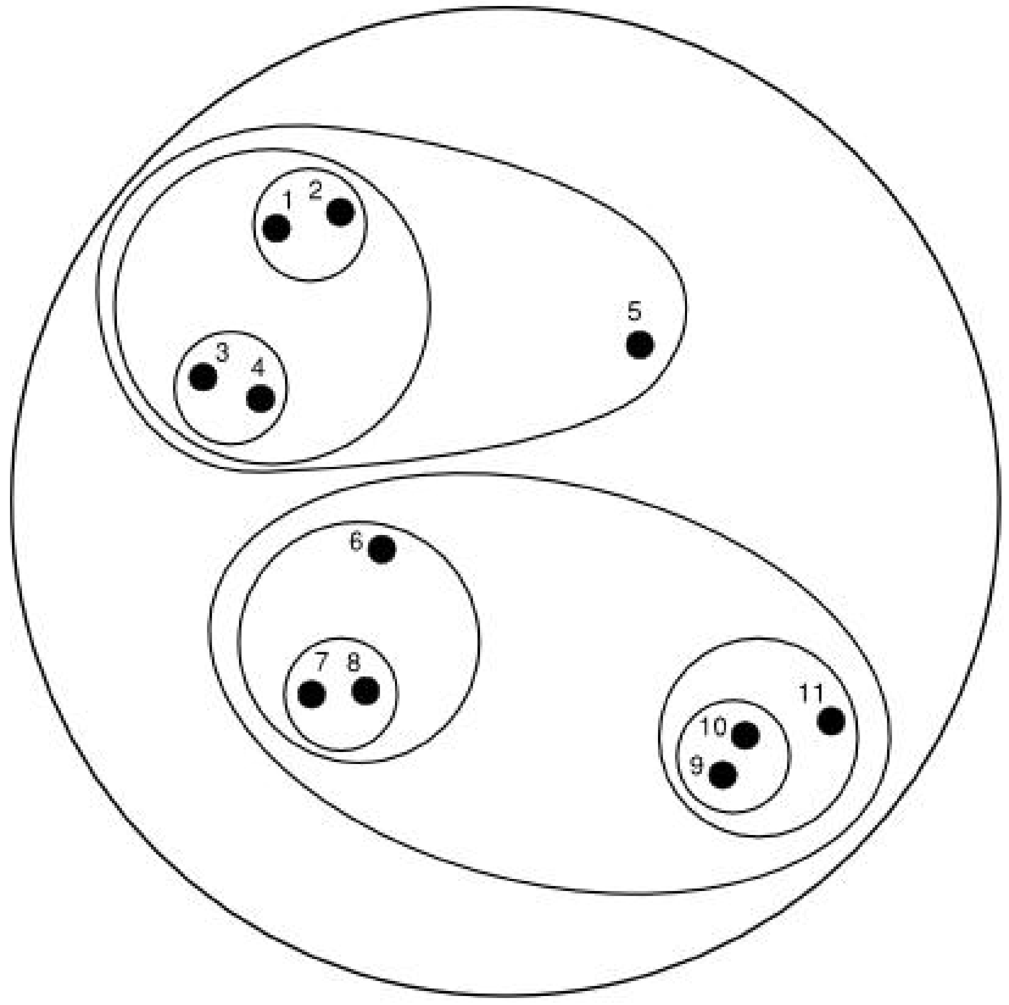
K-means summary

- Despite weaknesses, *k*-means is still the most popular algorithm due to its simplicity, efficiency and
  - other clustering algorithms have their own lists of weaknesses.
- No clear evidence that any other clustering algorithm performs better in general
  - although they may be more suitable for some specific types of data or applications.
- Comparing different clustering algorithms is a difficult task. No one knows the correct clusters!

Hierarchical Clustering

- Produce a nested sequence of clusters, a tree, also called Dendrogram.

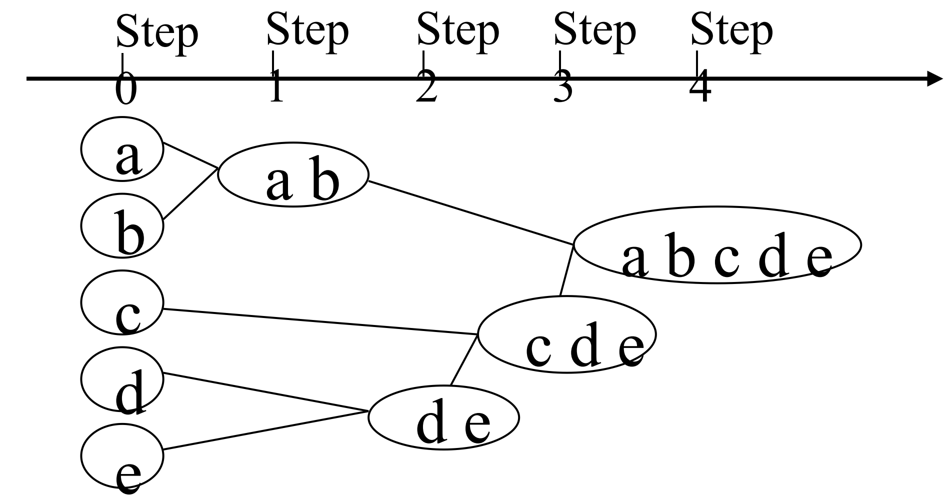




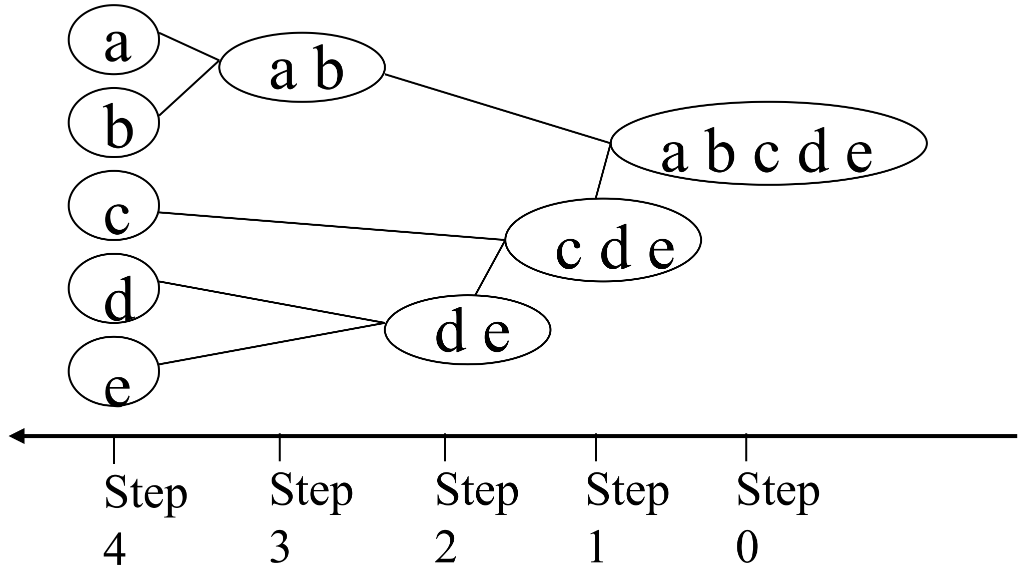
Types of hierarchical clustering

- Agglomerative (bottom up) clustering: It builds the dendrogram (tree) from the bottom level, and
  - merges the most similar (or nearest) pair of clusters
  - stops when all the data points are merged into a single cluster (i.e., the root cluster).
- Divisive (top down) clustering: It starts with all data points in one cluster, the root.
  - Splits the root into a set of child clusters. Each child cluster is recursively divided further
  - stops when only singleton clusters of individual data points remain, i.e., each cluster with only a single point

Types of hierarchical clustering



Agglomerative Nesting (AGNES)



Divisive Analysis (DIANA)

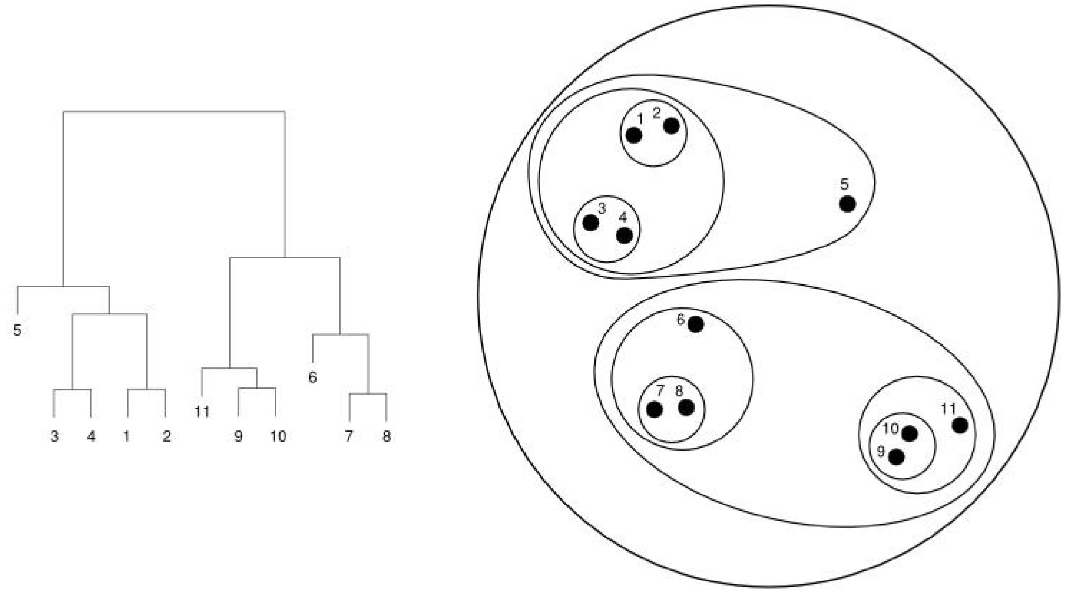
Agglomerative clustering

It is more popular than divisive methods.

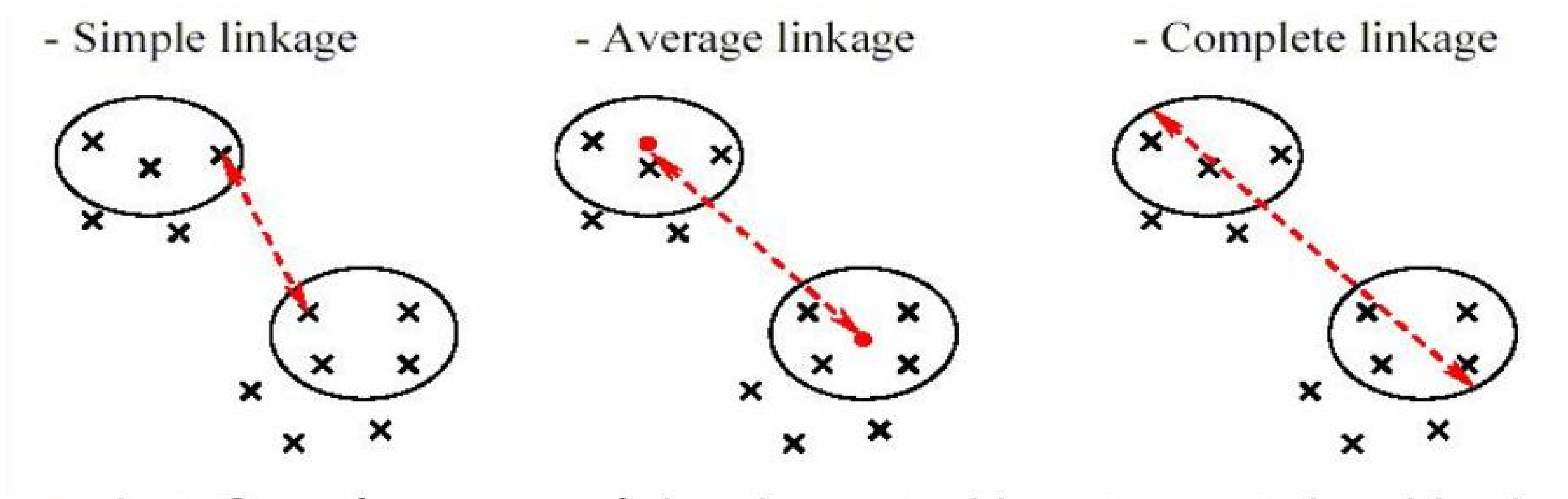
- At the beginning, each data point forms a cluster (also called a node).
- Merge nodes/clusters that have the least distance.
- Go on merging
- Eventually all nodes belong to one cluster

Agglomerative clustering algorithm

```
1: function HIERARCHICALCLUSTERING(inputs $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$)
2:     assign each input to a cluster
3:     repeat
4:         link the two clusters with minimal distance
5:     until only a single root cluster left
6:     return tree of cluster linkages
7: end function
```



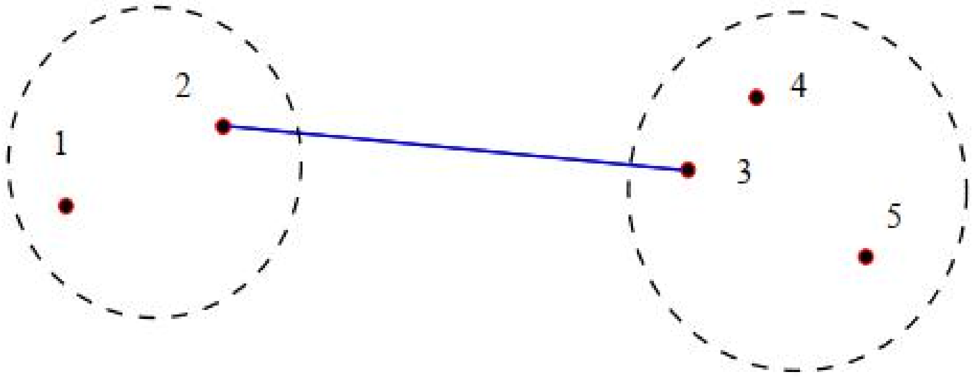
Measuring the distance of two clusters



- Let $S_j \subseteq \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be the set of inputs contained in the jth cluster
- **Simple linkage**: $d(i,j) := \min_{\mathbf{x} \in S_i, \tilde{\mathbf{x}} \in S_j} \|\mathbf{x} - \tilde{\mathbf{x}}\|$
- **Average linkage**: $d(i,j) := \text{mean}_{\mathbf{x} \in S_i, \tilde{\mathbf{x}} \in S_j} \|\mathbf{x} - \tilde{\mathbf{x}}\|$
- **Complete linkage**: $d(i,j) := \max_{\mathbf{x} \in S_i, \tilde{\mathbf{x}} \in S_j} \|\mathbf{x} - \tilde{\mathbf{x}}\|$

Single link method

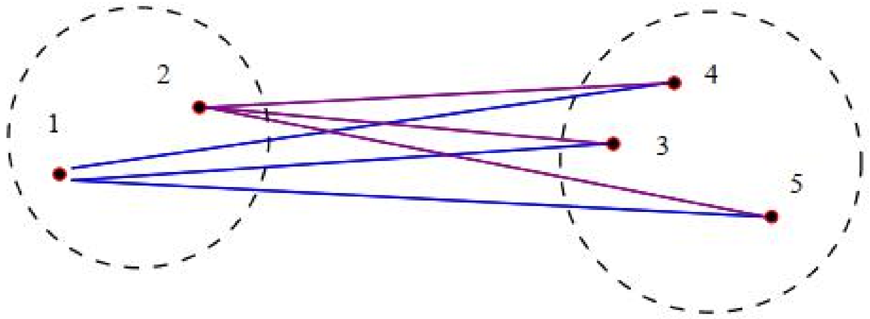
- The distance between two clusters is the distance between two closest data points in the two clusters, one data point from each cluster.
- It can find arbitrarily shaped clusters, but
  - It may cause the undesirable "chain effect" by noisy points



Distance between clusters = $d_{2,3}$

Average link and centroid methods

- Average link: the distance between two clusters is the average distance of all pair-wise distances between the data points in two clusters.

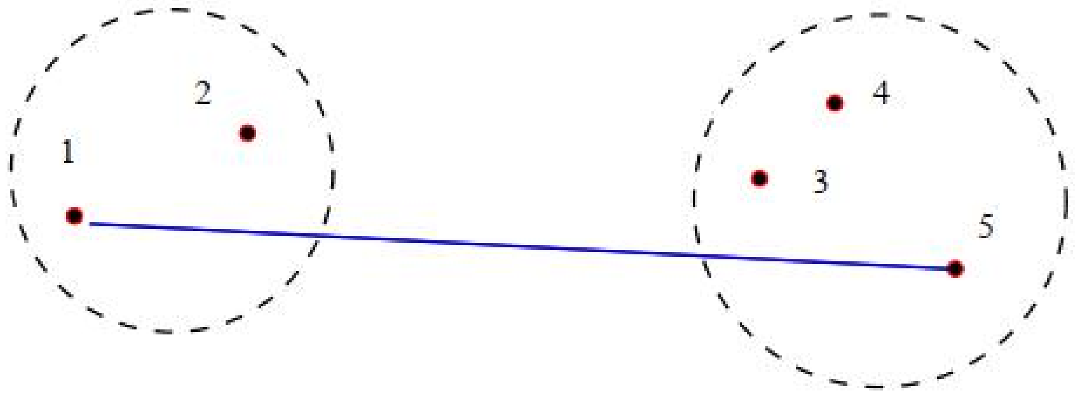


$$\text{Distance between clusters} = \frac{d_{1,3} + d_{1,4} + d_{1,5} + d_{2,3} + d_{2,4} + d_{2,5}}{6}$$

- Centroid method: In this method, the distance between two clusters is the distance between their centroids

Complete link method

- The distance between two clusters is the distance of two furthest data points in the two clusters.
- It is sensitive to outliers because they are far away



Distance between clusters = $d_{1,5}$

Hierarchical Clustering: the complexity

- All the algorithms are at least O(n^2). n is the number of data points.
- Single link can be done in O(n^2).
- Complete and average links can be done in O(n^2logn).
- Due the complexity, hard to use for large data sets.

Hierarchical Clustering: Problems and Limitations

Once a decision is made to combine two clusters, it cannot be undone

No objective function is directly minimized

Different schemes have problems with one or more of the following:

- Sensitivity to noise and outliers
- Difficulty handling different sized clusters and convex shapes
- Breaking large clusters

How to choose a clustering algorithm

- Clustering research has a long history. A vast collection of algorithms are available.
  - We only introduced several main algorithms.
- Choosing the "best" algorithm is a challenge.
  - Every algorithm has limitations and works well with certain data distributions.
  - It is very hard, if not impossible, to know what distribution the application data follow. The data may not fully follow any "ideal" structure or distribution required by the algorithms.
  - One also needs to decide how to standardize the data, to choose a suitable distance function and to select other parameter values.

Choose a clustering algorithm (cont …)

- Due to these complexities, the common practice is to
  - run several algorithms using different distance functions and parameter settings, and
  - then carefully analyze and compare the results.
- The interpretation of the results must be based on insight into the meaning of the original data together with knowledge of the algorithms used.
- Clustering is highly application dependent and to certain extent subjective (personal preferences).

Cluster Evaluation

- For cluster analysis, the question is how to evaluate the “goodness” of the resulting clusters?
- But “clusters are in the eye of the beholder”!
- Then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters
  - Measuring Cluster Quality:
    - Ground truth is the ideal clustering that is often built using human experts.
    - **Extrinsic Method**: If ground truth is available, compares the clustering against the ground truth and measure.
    - **Intrinsic Method**: If ground truth is not available, evaluate goodness of a clustering by considering how well the clusters are separated.

Summary

- The validation of clustering structures is the most difficult and frustrating part of cluster analysis.
- Clustering is has along history and still active
  - There are a huge number of clustering algorithms
  - More are still coming every year.
- We only introduced several main algorithms. There are many others, e.g.,
  - sub-space clustering, scale-up methods, neural networks based methods, fuzzy clustering, co-clustering, etc.
- Clustering is hard to evaluate, but very useful in practice. This partially explains why there are still a large number of clustering algorithms being devised every year.
- Clustering is highly application dependent and to some extent subjective.

Dimensionality reduction techniques

- Another unsupervised learning setting e.g PCA
- Dimensionality reduction is simply, the process of reducing the dimension of your feature set.
- Your feature set could be a dataset with a hundred columns (i.e features).

- High-Dimensions = Lot of Features

Document classification

Features per document =

thousands of words/unigrams

millions of bigrams, contextual information

Dimensionality reduction techniques

Dimensionality reduction

- Represent the data $x_1, \ldots, x_n \in R^d$ in a subspace of lower dimension with as little loss of information as possible

- **Advantages:**
  - Visualiaztion
  - Lower computation and time complexity
  - Avoid overfiting and reduce noise

Dimensionality reduction techniques

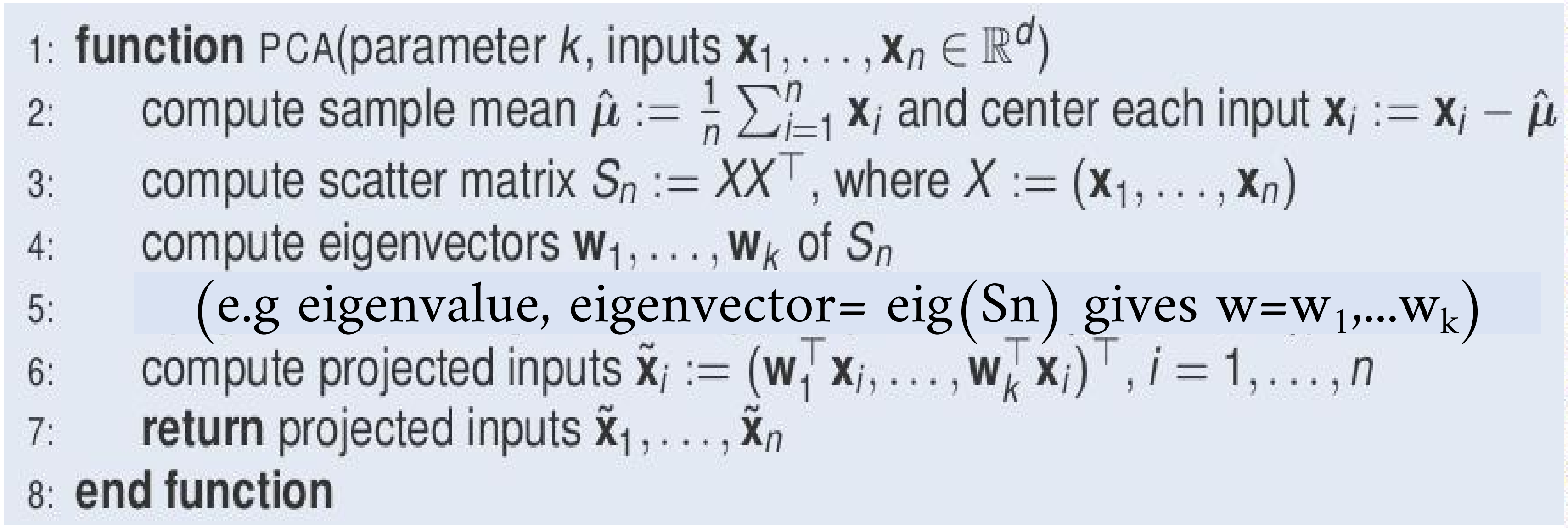
- Problem settings:
  - Given $x_1, \ldots, x_n \in R^d$
  - find a k -dimensional subspace
  - such that the data projected onto that space
  - is as close to the original data as possible

  - Common techniques:
    - **Principal Component Analysis(PCA)**
    - **Factor Analysis**
    - **Linear Discriminant Analysis(LDA)**

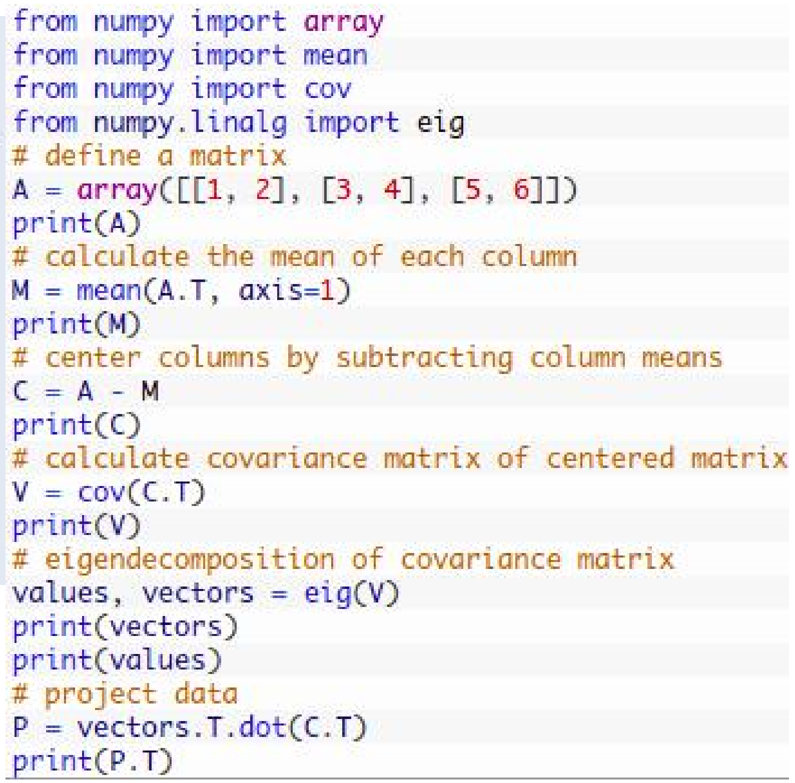
please read FA and LDA

Dimensionality reduction techniques

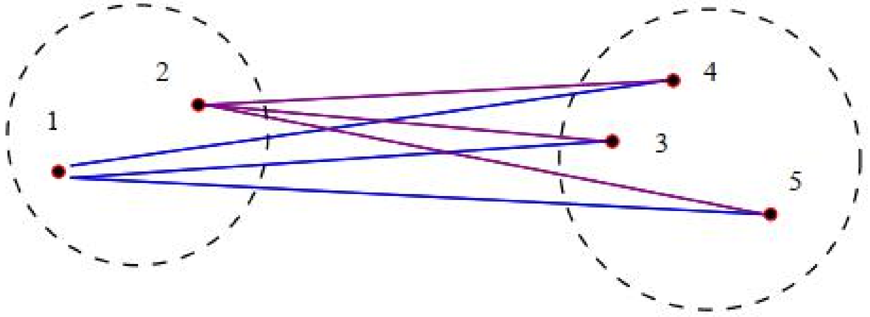
- The most common and well known dimensionality reduction (PCA, LDA, FA)
  - **Principal Component Analysis(PCA)**: PCA rotates and projects data along the direction of increasing variance. The features with the maximum variance are the principal components.



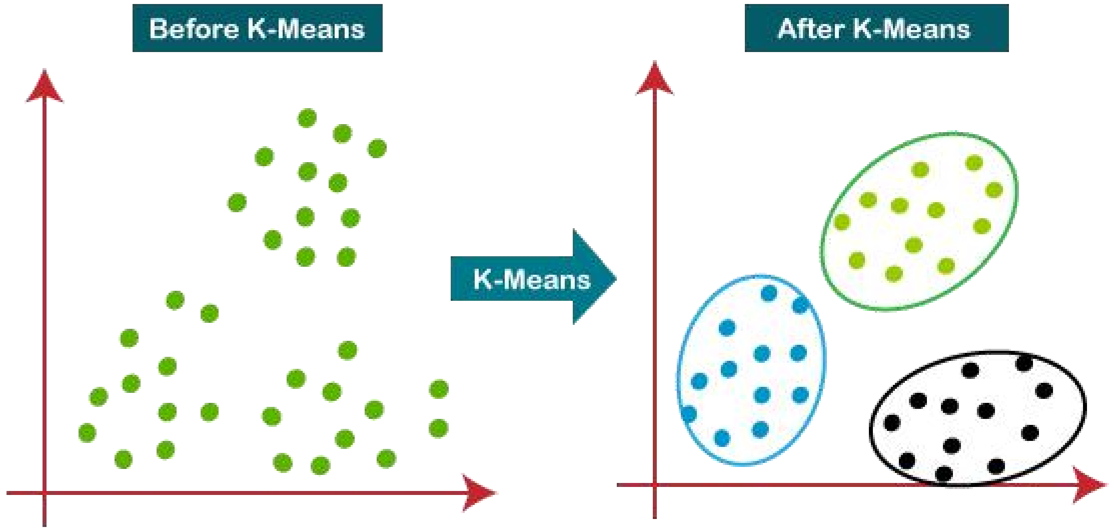
```
1: function PCA(parameter $k$, inputs $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$)
2:     compute sample mean $\hat{\mu} := \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ and center each input $\mathbf{x}_i := \mathbf{x}_i - \hat{\mu}$
3:     compute scatter matrix $S_n := XX^\top$, where $X := (\mathbf{x}_1, \ldots, \mathbf{x}_n)$
4:     compute eigenvectors $\mathbf{w}_1, \ldots, \mathbf{w}_k$ of $S_n$
5:         (e.g eigenvalue, eigenvector= eig(Sn) gives $w = w_1, \ldots w_k$)
6:     compute projected inputs $\tilde{\mathbf{x}}_i := (\mathbf{w}_1^\top \mathbf{x}_i, \ldots, \mathbf{w}_k^\top \mathbf{x}_i)^\top, i = 1, \ldots, n$
7:     return projected inputs $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n$
8: end function
```



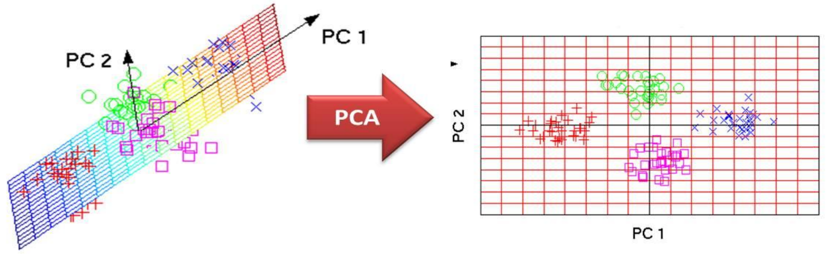
```python
from numpy import array
from numpy import mean
from numpy import cov
from numpy.linalg import eig
# define a matrix
A = array([[1, 2], [3, 4], [5, 6]])
print(A)
# calculate the mean of each column
M = mean(A.T, axis=1)
print(M)
# center columns by subtracting column means
C = A - M
print(C)
# calculate covariance matrix of centered matrix
V = cov(C.T)
print(V)
# eigendecomposition of covariance matrix
values, vectors = eig(V)
print(vectors)
print(values)
# project data
P = vectors.T.dot(C.T)
print(P.T)
```



$$\text{Distance between clusters} = \frac{d_{1,3} + d_{1,4} + d_{1,5} + d_{2,3} + d_{2,4} + d_{2,5}}{6}$$



Dimensionality Reduction & Principal Component Analysis



```
1: function PCA(parameter $k$, inputs $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$)
2:     compute sample mean $\hat{\mu} := \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ and center each input $\mathbf{x}_i := \mathbf{x}_i - \hat{\mu}$
3:     compute scatter matrix $S_n := XX^\top$, where $X := (\mathbf{x}_1, \ldots, \mathbf{x}_n)$
4:     compute eigenvectors $\mathbf{w}_1, \ldots, \mathbf{w}_k$ of $S_n$
5:       (e.g., in MATLAB: [foo,W] = eig($S_n$) gives $W = (\mathbf{w}_1, \ldots, \mathbf{w}_k)$)
6:     compute projected inputs $\tilde{\mathbf{x}}_i := (\mathbf{w}_1^\top \mathbf{x}_i, \ldots, \mathbf{w}_k^\top \mathbf{x}_i)^\top$, $i = 1, \ldots, n$
7:     return projected inputs $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n$
8: end function
```