IST407/707 Applied Machine Learning

Clustering Techniques

Syracuse University
School of Information Studies

Helpful textbook chapters

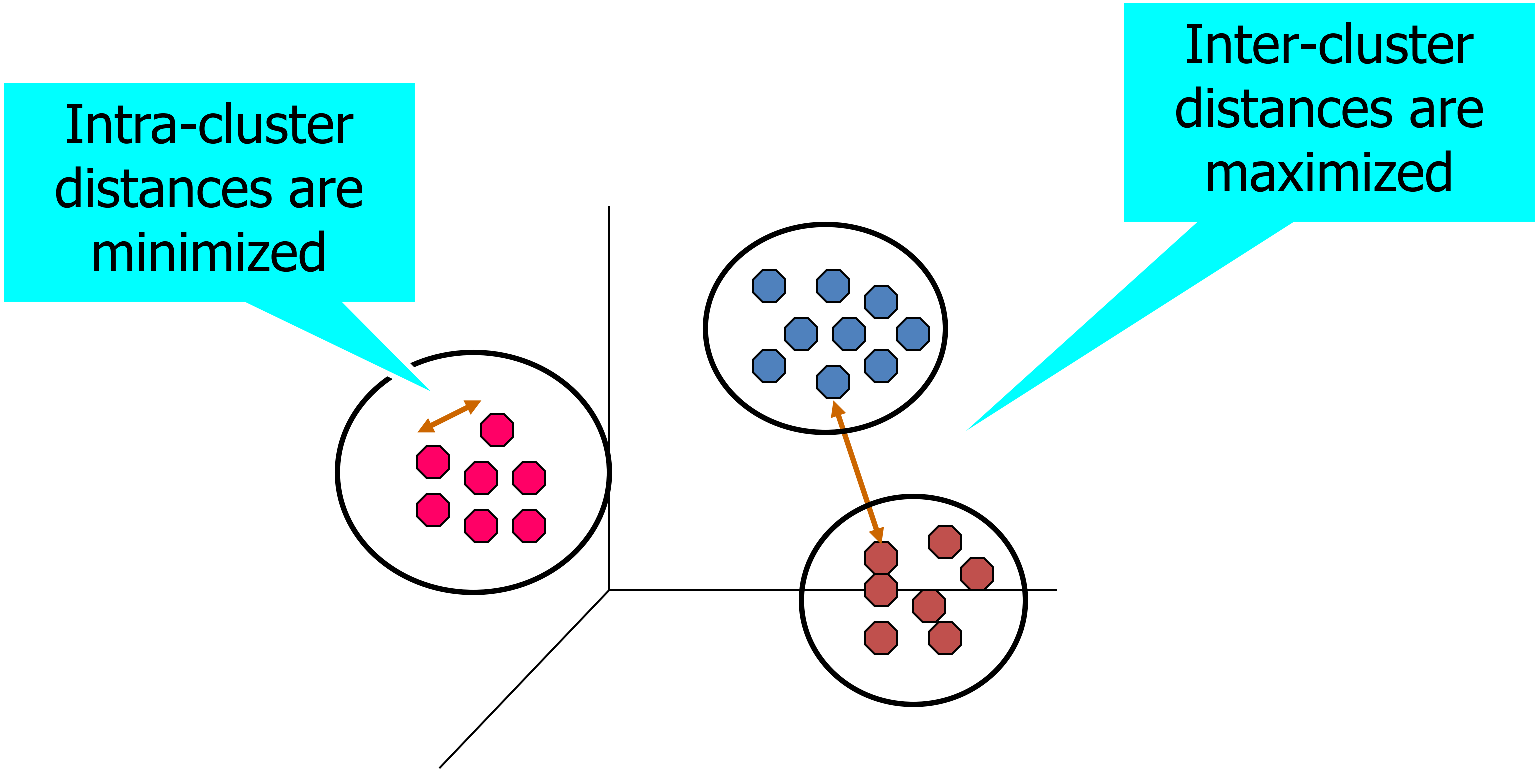
In Tan et al;

- Chapter 2.4 Measures of similarity and dissimilarity
- Chapter 8.1 cluster analysis overview
- Chapter 8.2 kMeans clustering algorithm
- Chapter 8.3 HAC algorithm

CLUSTER ANALYSIS

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

Why is Cluster Analysis Used ?

- When analyzing data, we may have millions of items. How can we understand the variance across items?
- We might hypothesize that items may only belong to a few groups or classes: items are similar within each segment but different across segments.
- Clustering is one way to induce (not deduce!) important groupings that inform subsequent analyses.
- *E.g. customer segmentation, event analysis, resource utilization, pathogen detection, etc.*

What is Cluster Analysis?

Unsupervised learning: no predefined classes

Typical applications

- Explore a large data set without prior knowledge about it.
    - Customer segmentation, document clustering, etc.
- Classification without training data
    - Usually less accurate than supervised learning methods.
- Outlier detection
    - E.g. Identify plagiarism cases

When is Clustering Appropriate?

- Automatic cluster detection is a tool for undirected data mining, because the automatic cluster detection techniques find patterns in the data with no target variable.
- Automatic cluster detection is an undirected data mining technique that can be used to learn about the structure of complex data. Clustering does not answer any question directly, but studying clusters can lead to valuable insights.
- One important application of clustering is customer segmentation. Clusters of customers form naturally occurring customer segments of people whose similarities may include similar needs and interests.
- Automatic cluster detection is a form of modeling. Clusters are detected in training data and the rules governing the clusters are captured in a model, which can be used to score previously unclassified data.
- After cluster labels have been assigned, they often become input to directed data mining models. They may also serve as reporting dimensions.

Requirements of Clustering in Data Mining

Scalability

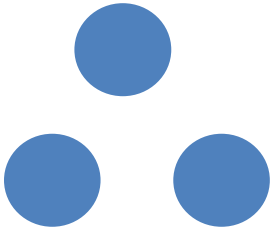
- Ability to explore large data set

Ability to deal with different types of attributes

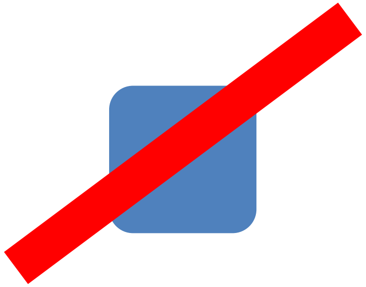
- Nominal, ordinal, numeric
- When we calculate dist. measure between variables, can algorithm handle all different variable types?

Discovery of clusters with arbitrary shape

- Spherical vs. other shapes



Easy to cluster using distance-based



Difficult to cluster because the two clusters are overlapped

Requirements for effective cluster analysis

- Some domain knowledge very helpful for determining input parameters and clustering technique
  - The number of desired clusters
- Strategy for noise and outliers
- Insensitive to order of input records
- High dimensionality (though depends on technique!)
  - Sparse data
- Interpretability and usability
  - Able to explain the patterns observed

These are the main aspects needed for effective clustering

Types of Clusterings

A clustering is a set of clusters

Important distinction between hierarchical and partitional sets of clusters

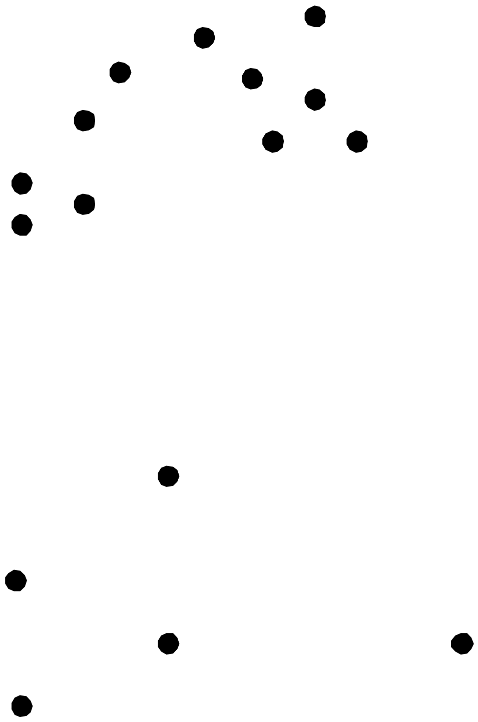
Partitional (flat) Clustering

- A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset

Hierarchical clustering

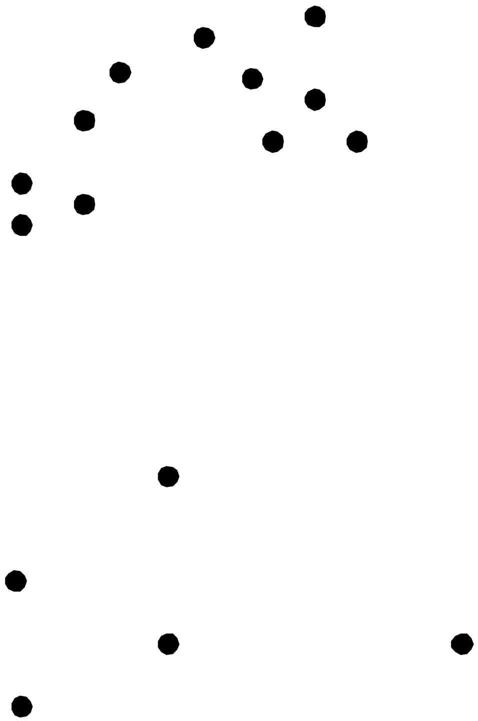
- A set of nested clusters organized as a hierarchical tree

Partitional Clustering

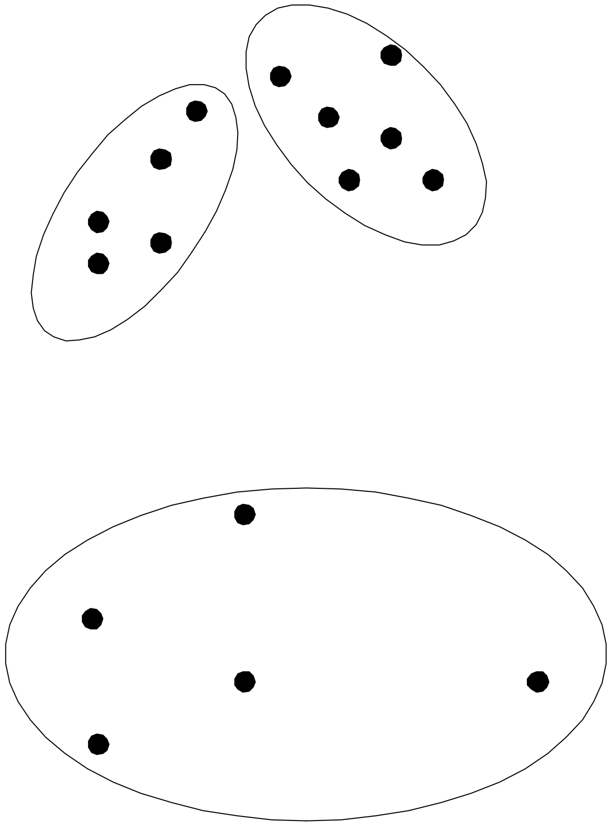


Original Points

Partitional Clustering

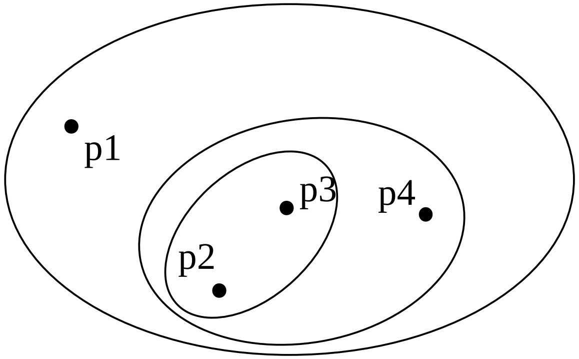


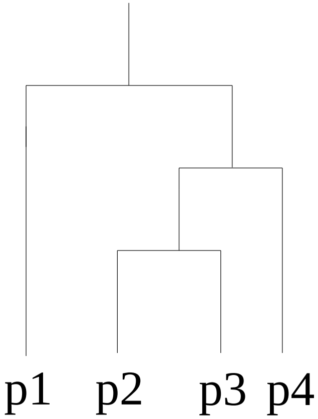
Original Points



A Partitional Clustering

Hierarchical Clustering

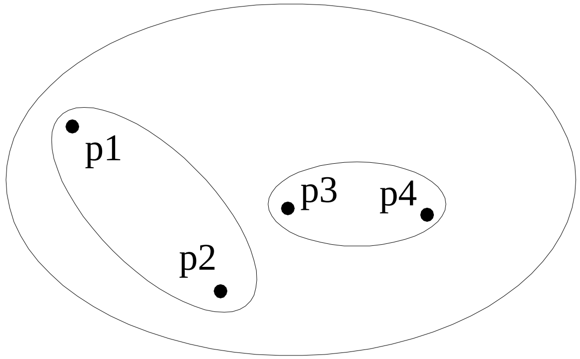


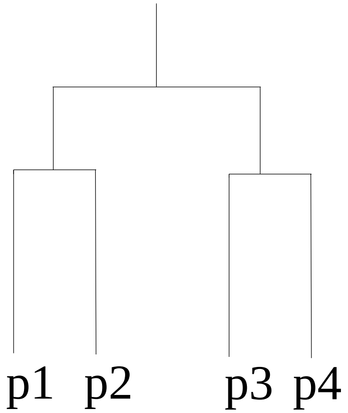


Steps

1. Start with two nearest data examples p2 and p3 and merge into cluster.
2. Merge with p4 to get larger cluster
3. Continue process until all data examples are included.

Hierarchical Clustering





We can also use parallel processing to allow examples to be clustered at the same

The Dendrogram shows the Process of Hierarchical Clustering .

Major Clustering Approaches

Partitioning approach:

- Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
- Typical methods: k-means, k-medoids, CLARANS, EM

Hierarchical approach:

- Create a hierarchical decomposition of the set of data (or objects) using some criterion
- Typical methods: Diana, Agnes, BIRCH, ROCK, CAMELEON

There many Algorithms Available for Clustering

Where Can I Apply K-Means?

Here is a list of nine interesting use cases for k-means.

Document Classification

- Cluster documents in multiple categories based on tags, topics, and the content of the document. This is a very standard classification problem and k-means is a highly suitable algorithm for this purpose. The initial processing of the documents is needed to represent each document as a vector and uses term frequency to identify commonly used terms that help classify the document. The document vectors are then clustered to help identify similarity in document groups. Here is a sample implementation of the k-means for document clustering.

Delivery Store Optimization

- Optimize the process of good delivery using truck drones by using a combination of k-means to find the optimal number of launch locations and a genetic algorithm to solve the truck route as a traveling salesman problem. Here is a whitepaper on the same topic.

Identifying Crime Localities

- With data related to crimes available in specific localities in a city, the category of crime, the area of the crime, and the association between the two can give quality insight into crime-prone areas within a city or a locality. Here is an interesting paper based on crime data from Delhi FIRs.

https://dzone.com/articles/10-interesting-use-cases-for-the-k-means-algorithm

Where Can I Apply K-Means?

Here is a list of nine interesting use cases for k-means.

Customer Segmentation

- Clustering helps marketers improve their customer base, work on target areas, and segment customers based on purchase history, interests, or activity monitoring. Here is a white paper on how telecom providers can cluster pre-paid customers to identify patterns in terms of money spent in recharging, sending SMS, and browsing the internet. The classification would help the company target specific clusters of customers for specific campaigns.

Insurance Fraud Detection

- Machine learning has a critical role to play in fraud detection and has numerous applications in automobile, healthcare, and insurance fraud detection. Utilizing past historical data on fraudulent claims, it is possible to isolate new claims based on its proximity to clusters that indicate fraudulent patterns. Since insurance fraud can potentially have a multi-million dollar impact on a company, the ability to detect frauds is crucial. Check out this white paper on using clustering in automobile insurance to detect frauds.

Automatic Clustering of IT Alerts

- Large enterprise IT infrastructure technology components such as network, storage, or database generate large volumes of alert messages. Because alert messages potentially point to operational issues, they must be manually screened for prioritization for downstream processes. Clustering of data can provide insight into categories of alerts and mean time to repair, and help in failure predictions.

https://dzone.com/articles/10-interesting-use-cases-for-the-k-means-algorithm

Where Can I Apply K-Means?

Here is a list of ten interesting use cases for k-means.

Rideshare Data Analysis

- The publicly available Uber ride information dataset provides a large amount of valuable data around traffic, transit time, peak pickup localities, and more. Analyzing this data is useful not just in the context of Uber but also in providing insight into urban traffic patterns and helping us plan for the cities of the future. Here is an article with links to a sample dataset and a process for analyzing Uber data.

Cyber-Profiling Criminals

- Cyber-profiling is the process of collecting data from individuals and groups to identify significant co-relations. The idea of cyber profiling is derived from criminal profiles, which provide information on the investigation division to classify the types of criminals who were at the crime scene. Here is an interesting white paper on how to cyber-profile users in an academic environment based on user data preferences.

Call Record Detail Analysis

- A call detail record (CDR) is the information captured by telecom companies during the call, SMS, and internet activity of a customer. This information provides greater insights about the customer’s needs when used with customer demographics.

https://dzone.com/articles/10-interesting-use-cases-for-the-k-means-algorithm

Aside: Heilmeier’s catechsim

George H. Heilmeier, a former DARPA director (1975-1977), crafted a set of questions known as the "Heilmeier Catechism" to help Agency officials think through and evaluate proposed research programs.

1. What are you trying to do? Articulate your objectives using absolutely no jargon.
2. How is it done today, and what are the limits of current practice?
3. What is new in your approach and why do you think it will be successful?
4. Who cares? If you are successful, what difference will it make?
5. What are the risks?
6. How much will it cost?
7. How long will it take?
8. What are the mid-term and final “exams” to check for success?

Class Discussion

Your management team wants to implement a Cluster based solution. They would like to understand the risks and benefits of such a solution, and how it could be incorporated into everyday workflows.

Identify a use case and develop the outline of a proposal. Please include:

- **Context and domain:** Provide a sense of the domain and what stakeholders' needs are.
- **Objectives (H1):** What are the specific objectives you hope to address address?
- **State of the art (H2):** How are these needs currently being met (or aren't they)?
- **Method fit (H3):** Why is clustering the right method?
- **Value added (H4):** What is the benefit of applying this technique given stakeholder needs?
- **Data Needed (H5 & H6):** What data is necessary and how will we collect it? Are there cost / benefit tradeoffs to consider?
- **Metrics (H8):** How will we know if we've been successful?

DISTANCE MEASURES

Distance Measures (chapter 2.4)

Similarity and distance: two opposite concepts:

- Similarity measures how close/similar two examples are.
- Distance measures how far/different two examples are.

The definitions of distance functions are dependent on variable types: numeric, nominal. Many data sets contain mixed types of attributes.

- Example: how similar are these two people?
  i = (Refund = No, Married, Income = 120K)
  j = (Refund = Yes, Married, Income = 90K)

Three variables in example: (One Numeric, Two Nominal)

How can we compare the similarities or distances between these two examples i and j ?

Numeric Attributes

If the data has all numeric attributes, distance measures can compare the numeric values of the attributes:

Some popular ones include: *Minkowski distance*:

$$d(i,j)=\sqrt[q]{(|x_{i_1}-x_{j_1}|^q+|x_{i_2}-x_{j_2}|^q+\ldots+|x_{i_p}-x_{j_p}|^q)}$$

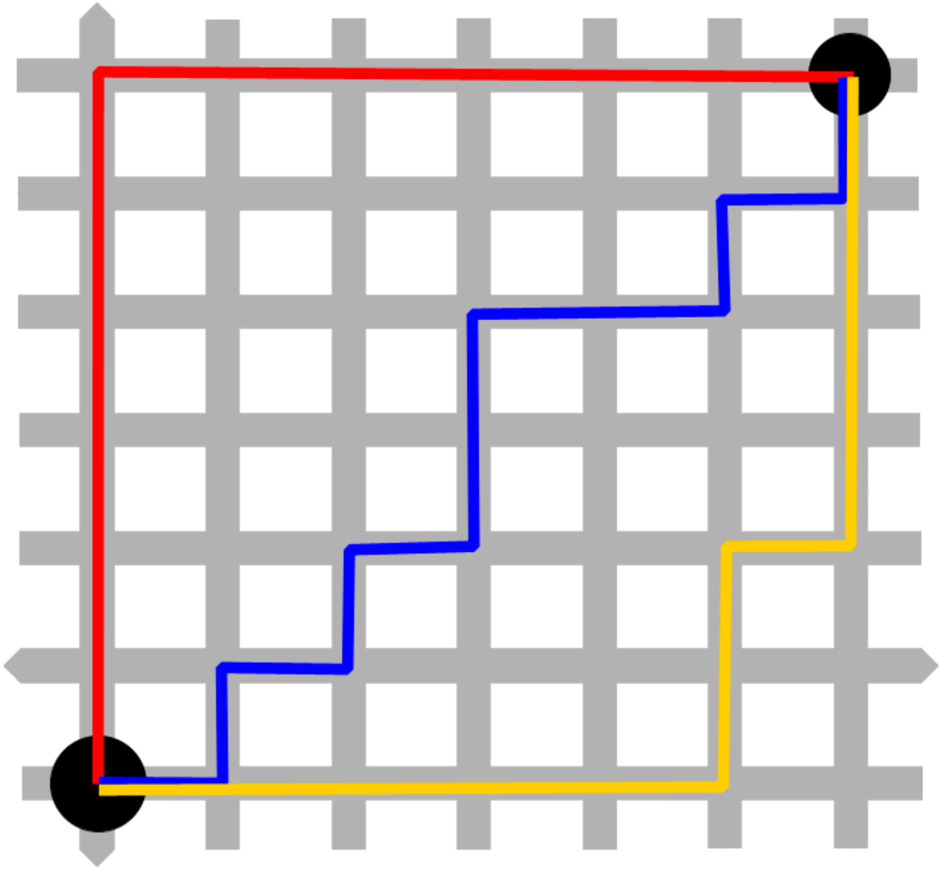
where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two *p*-dimensional data instances, and *q* is a positive integer

If *q = 1, d* is Manhattan distance

- Taking the absolute value of the differences between attribute values

$$d(i,j)=|x_{i_1}-x_{j_1}|+|x_{i_2}-x_{j_2}|+\ldots+|x_{i_p}-x_{j_p}|$$

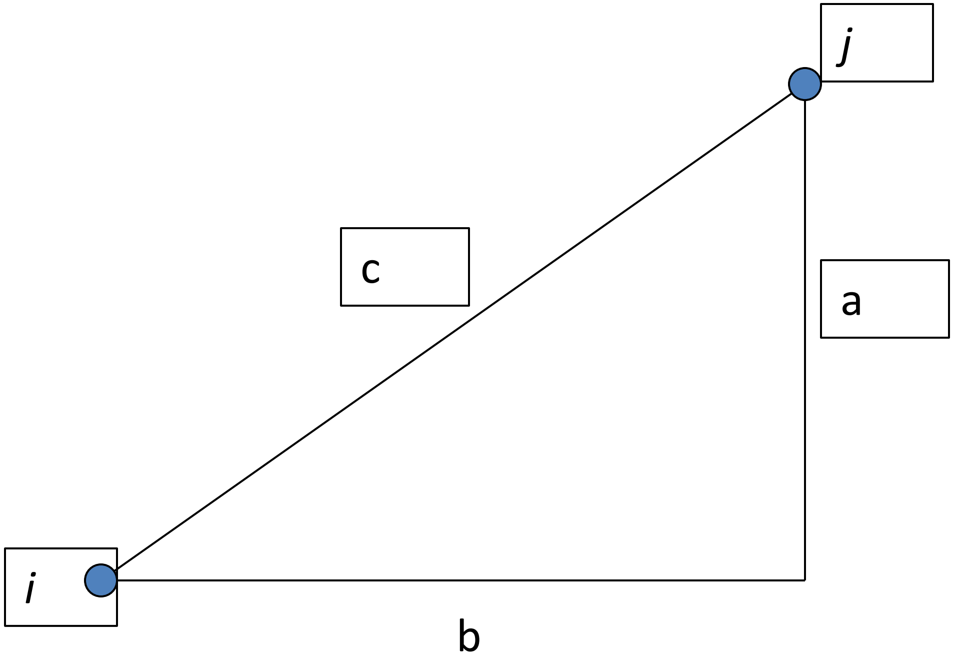
Manhattan Distance



Euclidean distance

When *q = 2, d* is Euclidean distance:

$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+\ldots+|x_{i_p}-x_{j_p}|^2)}$$



$$d_1(i,j)=a+b$$

$$d_2(i,j)=\sqrt{a^2+b^2}=c$$

Properties of distance measure

A distance measure should satisfy the following requirements:

- $d(i,j) \geq 0$ (non-negative value)
- $d(i,i) = 0$ (zero distance to itself)
- $d(i,j) = d(j,i)$ (symmetric measure)
- $d(i,j) \leq d(i,k) + d(k,j)$ (shortest distance between two points) -> triangle inequality

Distance between nominal values

Example: how similar are these two people?

i = (Refund = Yes, Married, Income = 120K)

j = (Refund = No, Divorced, Income = 90K)

Taxpayer	Refund	Marital Status	Income in thousands
i	Yes	Married	120
j	No	Divorced	90

Method 1: simple matching

Taxpayer	Refund	Marital Status	Income in thousands
i	Yes	Married	120
j	No	Divorced	90

m: # of matches, *p*: total # of nominal variables

$$d(i,j) = \frac{p-m}{p}$$

Method 2: convert nominal to binary variables

Taxpayer	Refund	Marital Status	Income in thousands
i	Yes	Married	120
j	No	Divorced	90

Convert a nominal attribute to multiple binary attributes, and treat binary attributes as numeric (0 or 1)

Taxpayer	Refund	Married?	Divorced?	Single?	Income
1	1	1	0	0	120
2	0	0	1	0	90

Binary variables: symmetric or asymmetric

All patients run through many tests.

How different are their test results?

Patient	Test1	Test2	Test3	Test4	Test5	test6
Jack	1	0	1	0	0	0
Mary	1	0	1	0	1	0

Binary variables: symmetric or asymmetric

A contingency table for binary data

- Gives the number of attributes of each pair of values

$$\begin{array}{cccc} & 1 & 0 & sum \\ 1 & a & b & a+b \\ 0 & c & d & c+d \\ sum & a+c & b+d & p \end{array}$$

Patient	Test1	Test2	Test3	Test4	Test5	test6
Jack	1	0	1	0	0	0
Mary	1	0	1	0	1	0

Symmetric binary attributes

Distance measure for symmetric binary attributes:

$$\begin{array}{ccccc} & 1 & 0 & sum \\ 1 & a & b & a+b \\ 0 & c & d & c+d \\ sum & a+c & b+d & p \end{array}$$

$$d(i, j) = \frac{b+c}{a+b+c+d}$$

Asymmetric binary attributes

If most test results are negative,

d will be much greater than a, b, and c . Sharing many negative test results is not that informative to doctors.

	1	0	sum
1	a	b	$a+b$
0	c	d	$c+d$
sum	$a+c$	$b+d$	p

Distance measure for asymmetric binary attributes:

$$d(i,j)=\frac{b+c}{a+b+c}$$

Distance between ordinal values

Method 1: treat as nominal

Method 2: treat as numeric

Attributes of Mixed Types

A database may contain different types of attributes

- symmetric binary, asymmetric binary, nominal, ordinal, numerical

How to compute the distance between examples with heterogeneous attributes?

- Calculate distance for each type of attribute and aggregate

Similarity measure

If defining a distance measure d in [0,1] range, similarity can be defined as 1-d.

Other similarity measures

- Cosine similarity measure

Note – beware Euclidean distance in high dimensions!

- https://stats.stackexchange.com/questions/99171/why-is-euclidean-distance-not-a-good-metric-in-high-dimensions

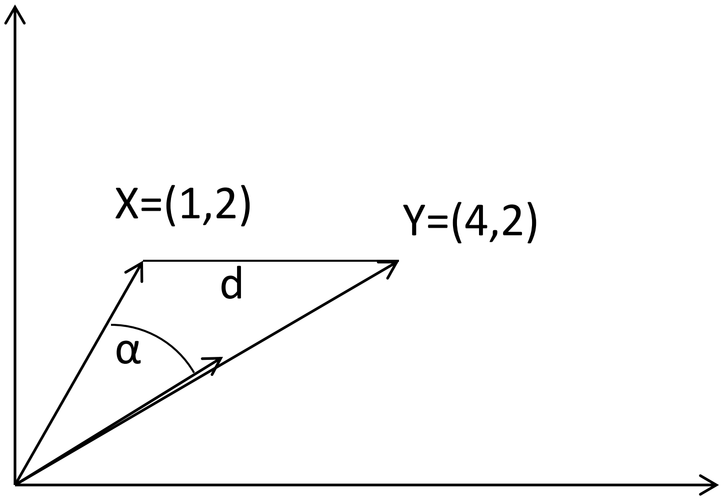
Vector space representation and Cosine similarity

Distance/similarity measures

- Euclidean distance

$$d = \vec{x} - \vec{y} = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

$$= \sqrt{(1-4)^2 + (2-2)^2} = 3$$



- Cosine similarity

$$\cos(\alpha) = \frac{x \cdot y}{|x||y|} = \frac{x_1 y_1 + x_2 y_2}{\sqrt{x_1{}^2 + x_2{}^2} \cdot \sqrt{y_1{}^2 + y_2{}^2}}$$

$$= \frac{1 \times 4 + 2 \times 2}{\sqrt{1^2 + 2^2} \cdot \sqrt{4^2 + 2^2}} = \frac{8}{\sqrt{5} \cdot \sqrt{20}} = 0.8$$

Cosine similarity

In the range of [0, 1]

- "0" means two vectors are perpendicular to each other
- "1" means same vector direction and length

Commonly used in information retrieval and text mining to compare document similarity

- High-dimensional space
    - Each word in the vocabulary is a dimension

Importance of Normalization

Sometimes we need to normalize the distance measure so that we don’t cause problems

Potential Problem Type 1

Different variables might use different scales

- Age: [0, 120]
- Income: [0, 2M]
- If averaging the difference on these two variables, income would weigh much more than age
- Solution: normalize both variables to the same scale, e.g. [0, 1]

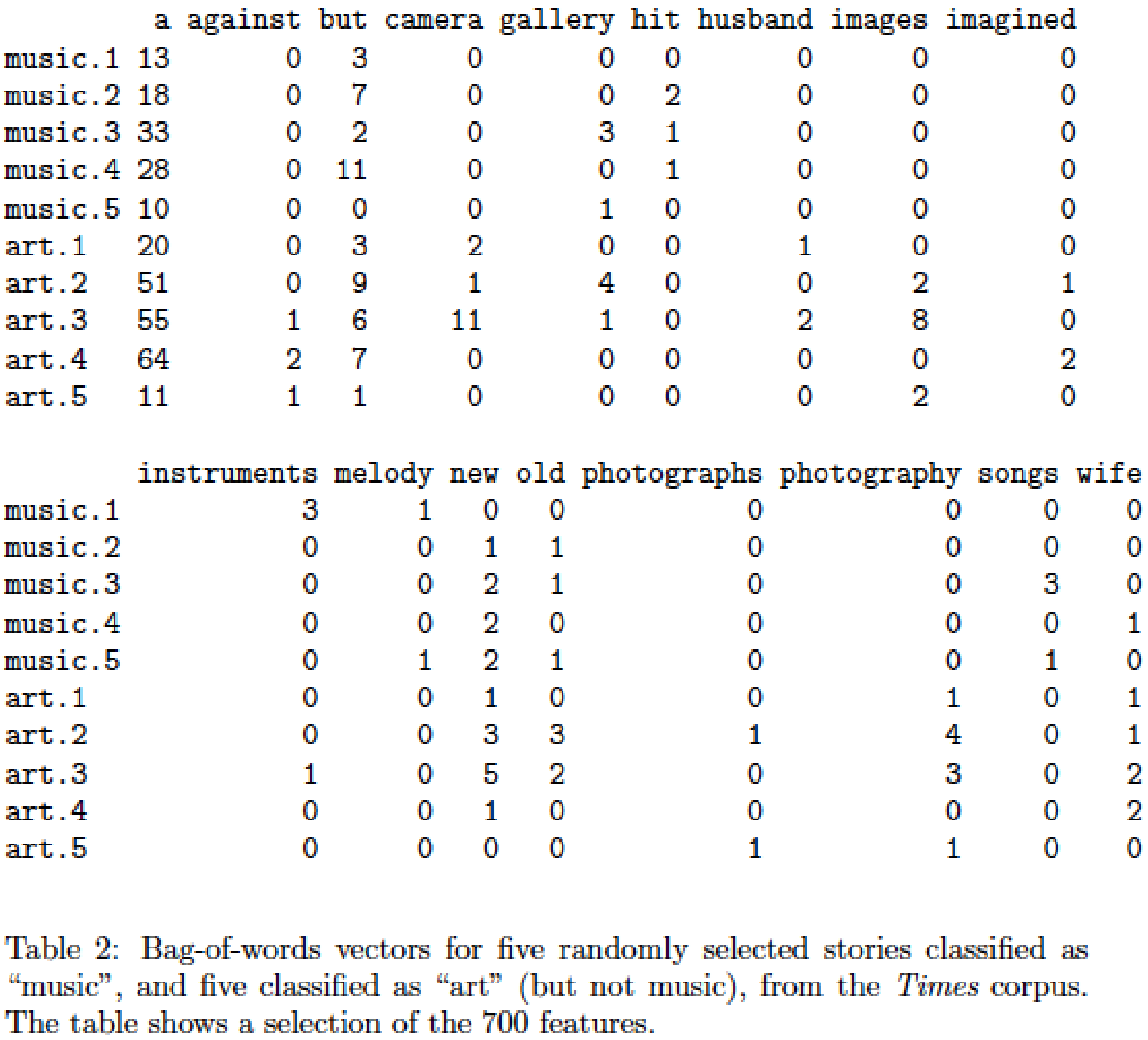
The dimensions with larger scales can dominate the distance measures while the dimension with smaller numbers will not be as impactful as they should be

Potential Problem Type 2

Lets say we have a collection of documents we want to analyze. Some documents are long, while others are short.

- Different examples(documents)/vectors might differ greatly in length
- E.g. for text documents, the vector lengths of long documents are much greater than the vector lengths short documents
- The longer documents tend to be far away from the short documents bases on the raw Euclidian distance.

An example of normalization in Information Retrieval

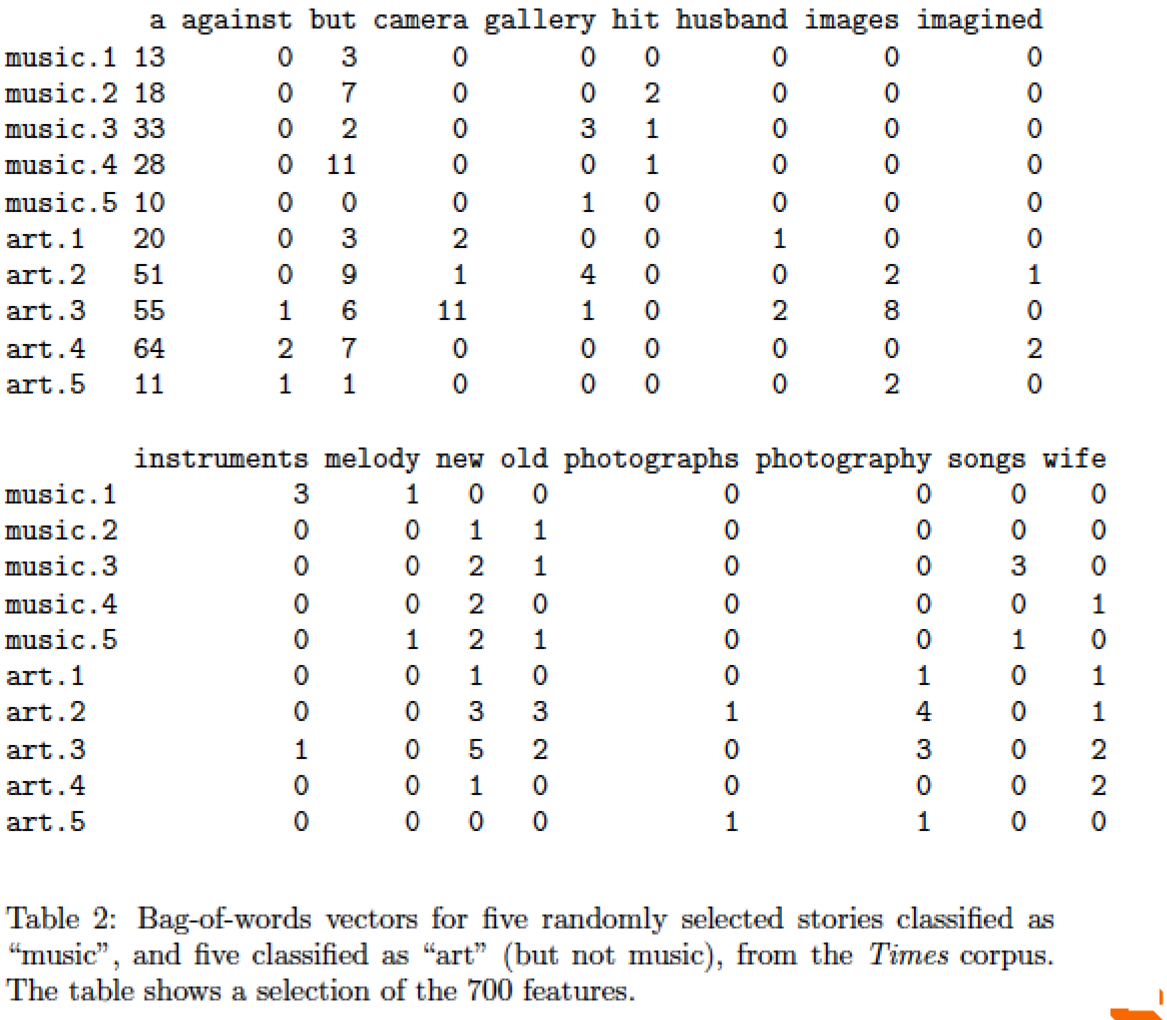


	a	against	but	camera	gallery	hit	husband	images	imagined
music.1	13	0	3	0	0	0	0	0	0
music.2	18	0	7	0	0	2	0	0	0
music.3	33	0	2	0	3	1	0	0	0
music.4	28	0	11	0	0	1	0	0	0
music.5	10	0	0	0	1	0	0	0	0
art.1	20	0	3	2	0	0	1	0	0
art.2	51	0	9	1	4	0	0	2	1
art.3	55	1	6	11	1	0	2	8	0
art.4	64	2	7	0	0	0	0	0	2
art.5	11	1	1	0	0	0	0	2	0

	instruments	melody	new	old	photographs	photography	songs	wife
music.1	3	1	0	0	0	0	0	0
music.2	0	0	1	1	0	0	0	0
music.3	0	0	2	1	0	0	3	0
music.4	0	0	2	0	0	0	0	1
music.5	0	1	2	1	0	0	1	0
art.1	0	0	1	0	0	1	0	1
art.2	0	0	3	3	1	4	0	1
art.3	1	0	5	2	0	3	0	2
art.4	0	0	1	0	0	0	0	2
art.5	0	0	0	0	1	1	0	0

Table 2: Bag-of-words vectors for five randomly selected stories classified as "music", and five classified as "art" (but not music), from the *Times* corpus. The table shows a selection of the 700 features.

Longer docs tend to be far away from short ones based on raw Euclidean distance

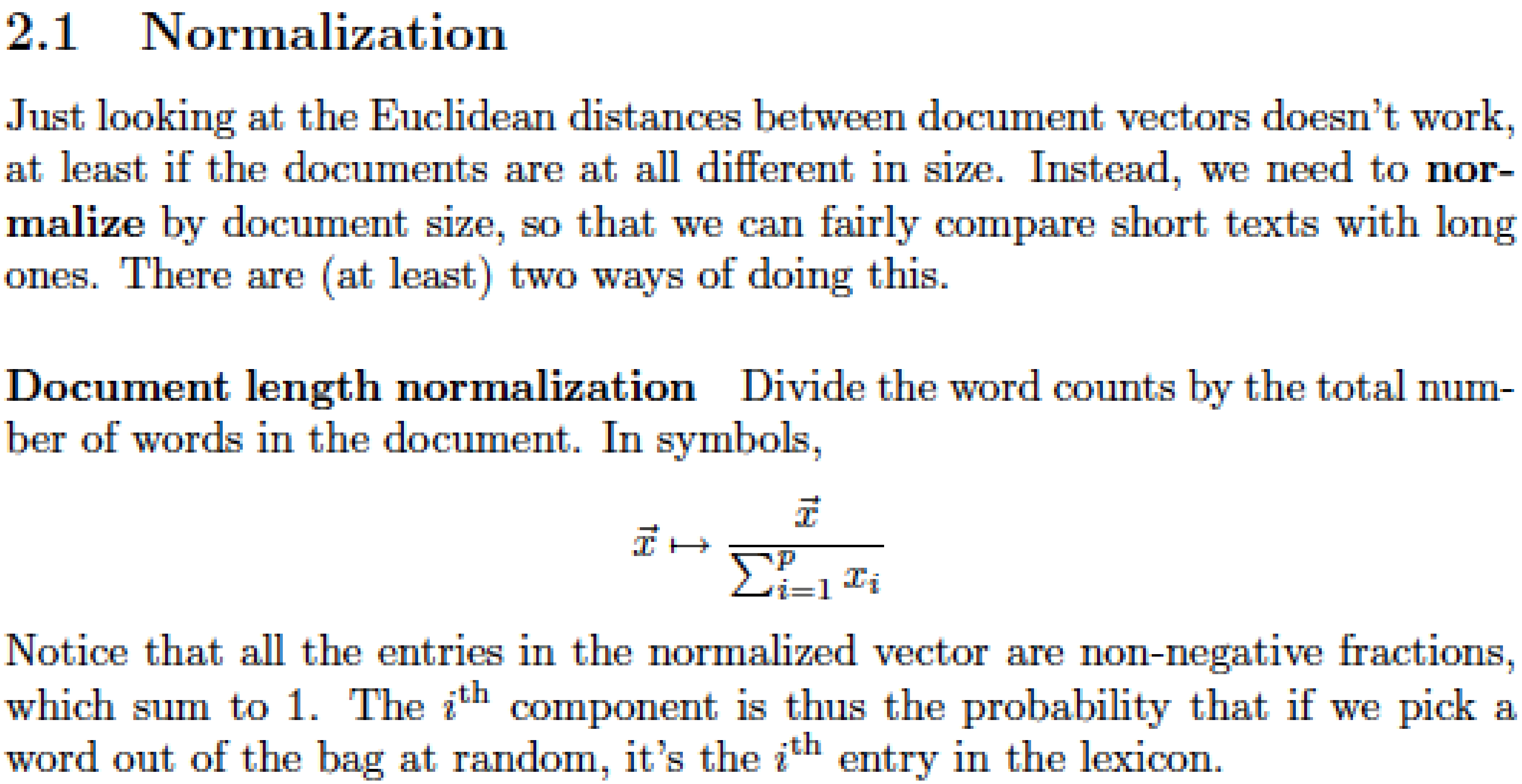


	a	against	but	camera	gallery	hit	husband	images	imagined
music.1	13	0	3	0	0	0	0	0	0
music.2	18	0	7	0	0	2	0	0	0
music.3	33	0	2	0	3	1	0	0	0
music.4	28	0	11	0	0	1	0	0	0
music.5	10	0	0	0	1	0	0	0	0
art.1	20	0	3	2	0	0	1	0	0
art.2	51	0	9	1	4	0	0	2	1
art.3	55	1	6	11	1	0	2	8	0
art.4	64	2	7	0	0	0	0	0	2
art.5	11	1	1	0	0	0	0	2	0

	instruments	melody	new	old	photographs	photography	songs	wife
music.1	3	1	0	0	0	0	0	0
music.2	0	0	1	1	0	0	0	0
music.3	0	0	2	1	0	0	3	0
music.4	0	0	2	0	0	0	0	1
music.5	0	1	2	1	0	0	1	0
art.1	0	0	1	0	0	1	0	1
art.2	0	0	3	3	1	4	0	1
art.3	1	0	5	2	0	3	0	2
art.4	0	0	1	0	0	0	0	2
art.5	0	0	0	0	1	1	0	0

Table 2: Bag-of-words vectors for five randomly selected stories classified as "music", and five classified as "art" (but not music), from the *Times* corpus. The table shows a selection of the 700 features.

Normalization by doc length (L-1)



2.1 Normalization

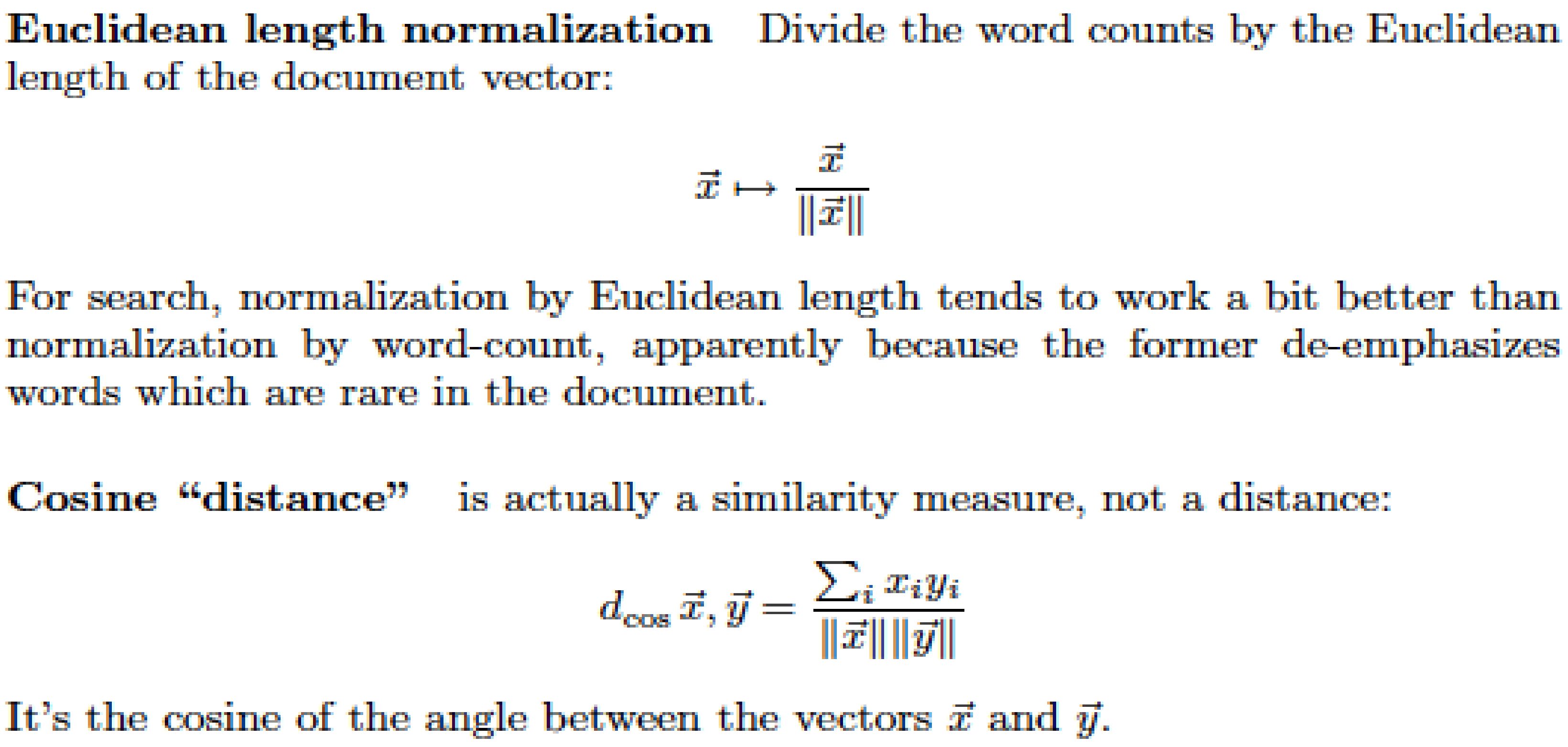
Just looking at the Euclidean distances between document vectors doesn't work, at least if the documents are at all different in size. Instead, we need to **normalize** by document size, so that we can fairly compare short texts with long ones. There are (at least) two ways of doing this.

Document length normalization Divide the word counts by the total number of words in the document. In symbols,

$$\vec{x} \mapsto \frac{\vec{x}}{\sum_{i=1}^{p} x_i}$$

Notice that all the entries in the normalized vector are non-negative fractions, which sum to 1. The i^{th} component is thus the probability that if we pick a word out of the bag at random, it's the i^{th} entry in the lexicon.

Normalization by Euclidean length (L-2)



Euclidean length normalization Divide the word counts by the Euclidean length of the document vector:

$$\vec{x} \mapsto \frac{\vec{x}}{||\vec{x}||}$$

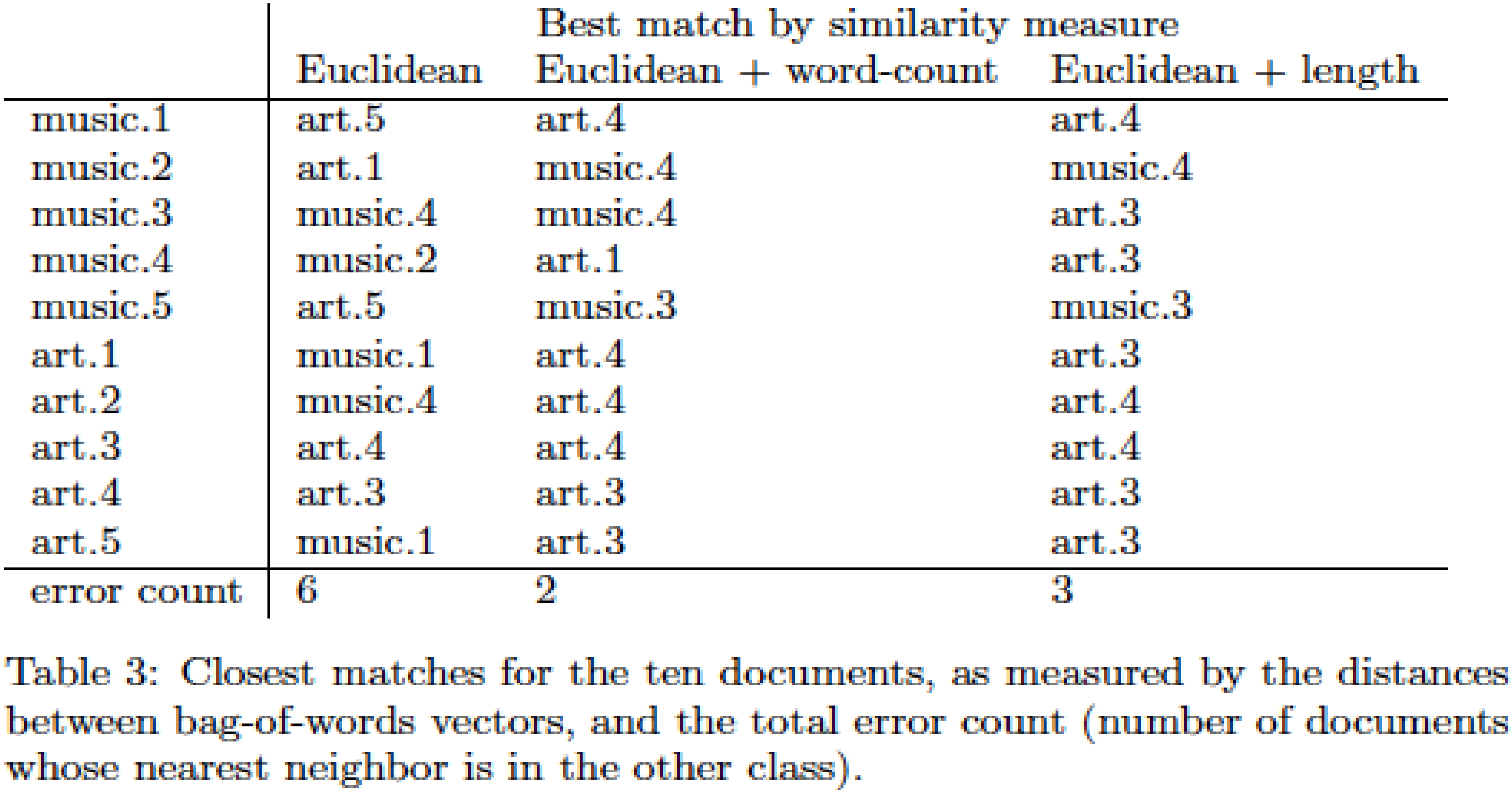
For search, normalization by Euclidean length tends to work a bit better than normalization by word-count, apparently because the former de-emphasizes words which are rare in the document.

Cosine "distance" is actually a similarity measure, not a distance:

$$d_{\cos} \vec{x}, \vec{y} = \frac{\sum_i x_i y_i}{||\vec{x}|| ||\vec{y}||}$$

It's the cosine of the angle between the vectors $\vec{x}$ and $\vec{y}$.

Compare results with/without normalization



	Best match by similarity measure		
	Euclidean	Euclidean + word-count	Euclidean + length
music.1	art.5	art.4	art.4
music.2	art.1	music.4	music.4
music.3	music.4	music.4	art.3
music.4	music.2	art.1	art.3
music.5	art.5	music.3	music.3
art.1	music.1	art.4	art.3
art.2	music.4	art.4	art.4
art.3	art.4	art.4	art.4
art.4	art.3	art.3	art.3
art.5	music.1	art.3	art.3
error count	6	2	3

Table 3: Closest matches for the ten documents, as measured by the distances between bag-of-words vectors, and the total error count (number of documents whose nearest neighbor is in the other class).

Weighted distance/similarity

For some data, some dimensions are more important than others, and thus their similarity/distance carries more weight. In these cases, we can assign different weights to individual dimensions/attributes.

$$d(i, j) = 2 \times |x_{i_1} - x_{j_1}| + |x_{i_2} - x_{j_2}|$$

$$similarity(p, q) = \frac{\sum_{k=1}^{n} w_k \delta_k s_k}{\sum_{k=1}^{n} \delta_k}$$

$$distance(p, q) = \left(\sum_{k=1}^{n} w_k |p_k - q_k|^r \right)^{1/r}.$$

Summary of distance/similarity

- Manhattan and Euclidean distance for numeric variables
- Properties of distance measure
- Distance for nominal variables: count matches or convert to binary
- Symmetric vs. asymmetric binary variables
- Convert ordinal to either numeric or nominal
- Calculate distance/similarity on each attribute or attribute group and then average over all
- Cosine similarity
- Importance of normalization

K-MEANS ALGORITHM

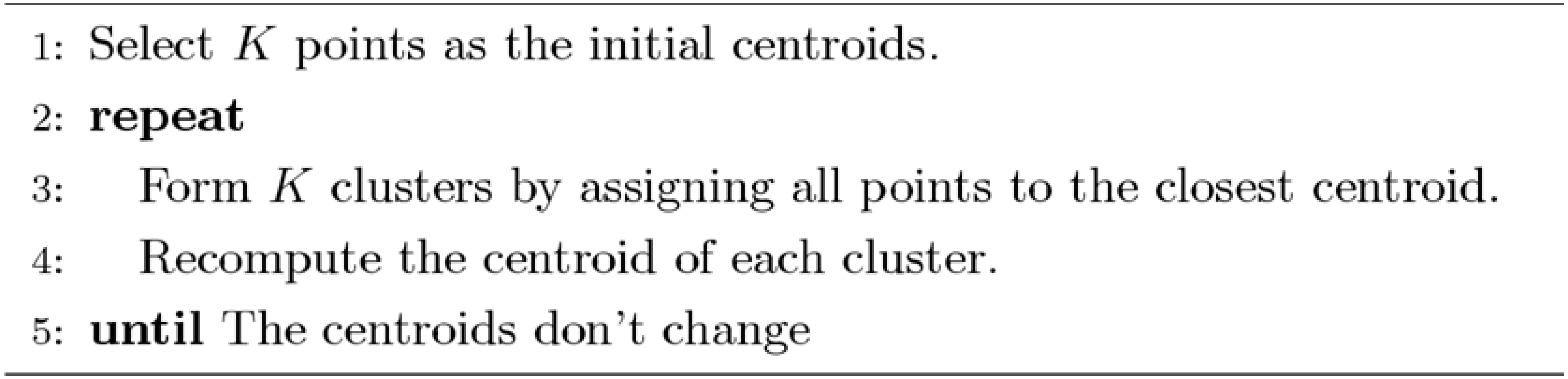
Centroid / Medoid of a Cluster

Centroid: the “gravity center” of a cluster, which is calculated as the average of all data examples in a cluster

- For numeric variable, use the mean as the average
- For nominal data, use mode as the average

Medoid: the data item that is closest to the centroid (or whose sum of dissimilarities to other objects is minimized)

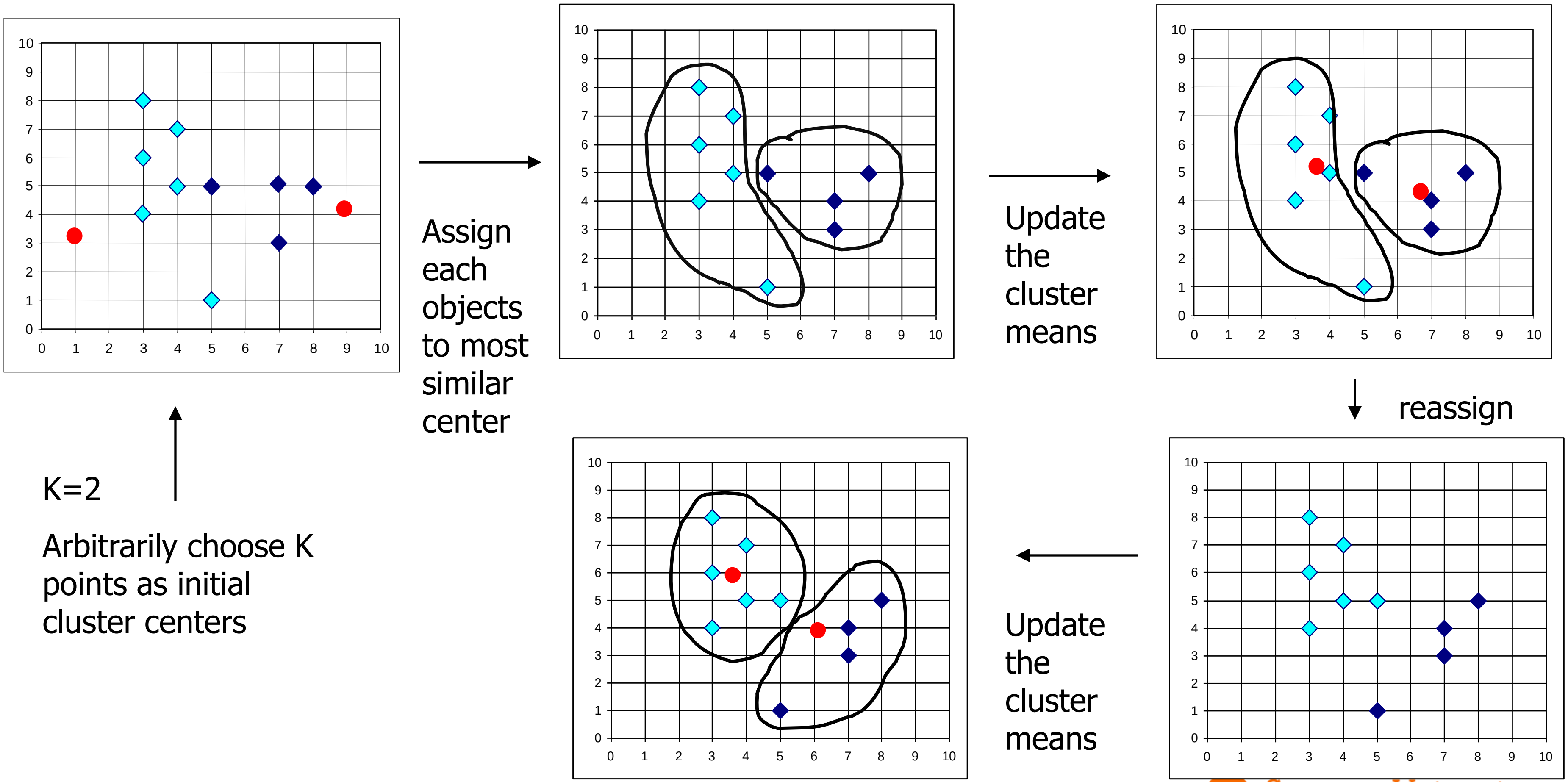
The *K-Means* Clustering Method



```
Select $K$ points as the initial centroids.
repeat
    Form $K$ clusters by assigning all points to the closest centroid.
    Recompute the centroid of each cluster.
until The centroids don't change
```

The *K-Means* Clustering Method

Example



K=2

Arbitrarily choose K points as initial cluster centers

Assign each objects to most similar center

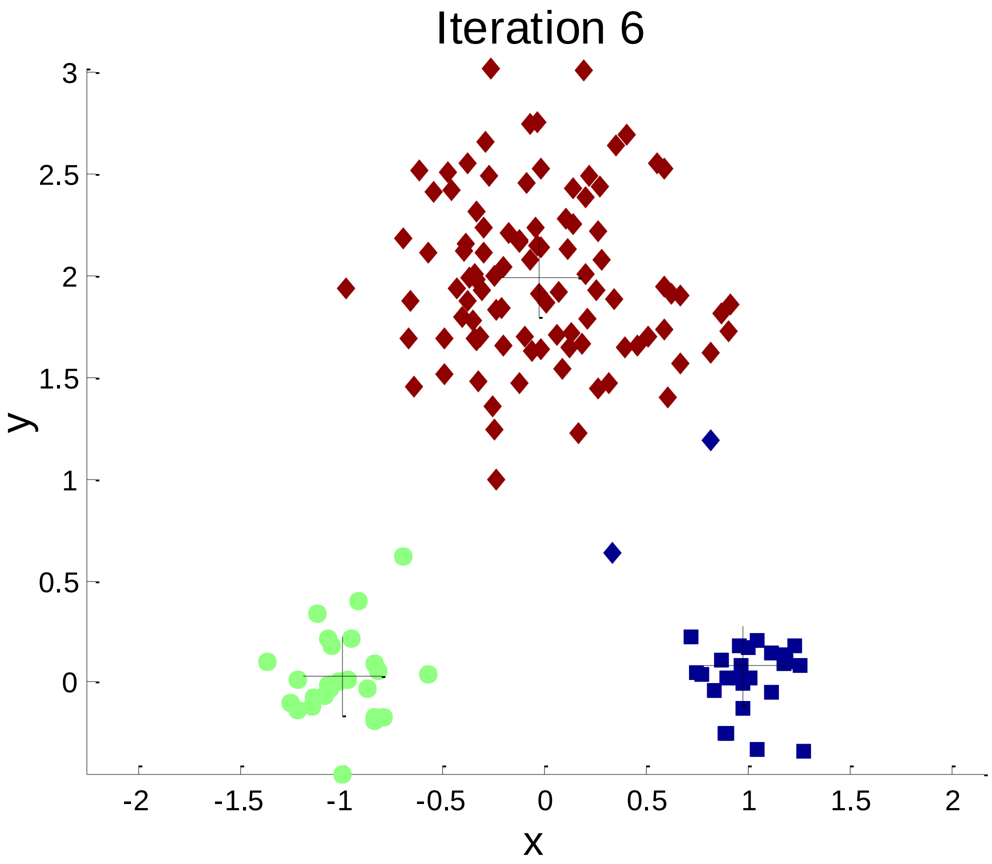
Update the cluster means

reassign

Update the cluster means

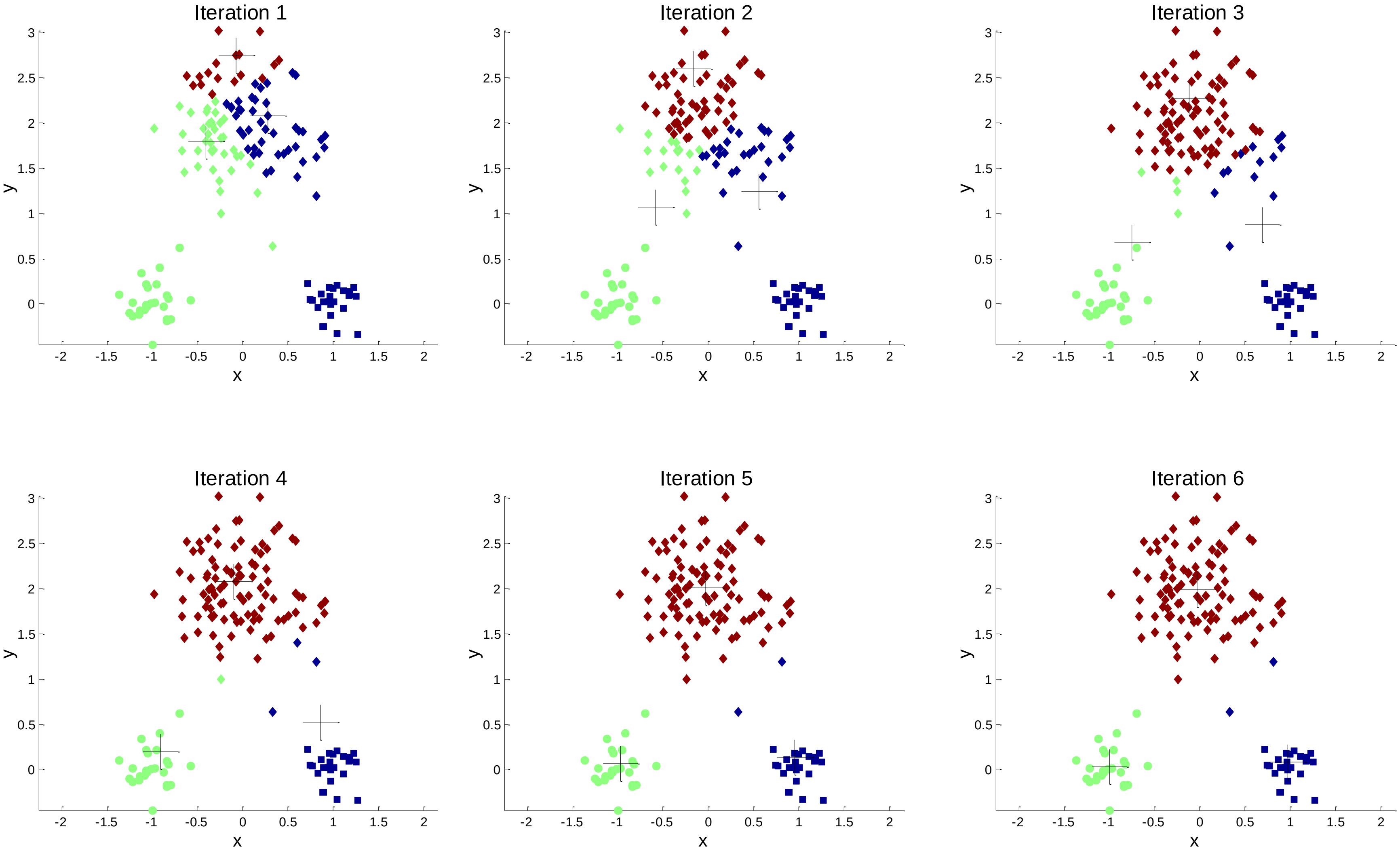
Syracuse University
School of Information Studies

Importance of Choosing Initial Centroids

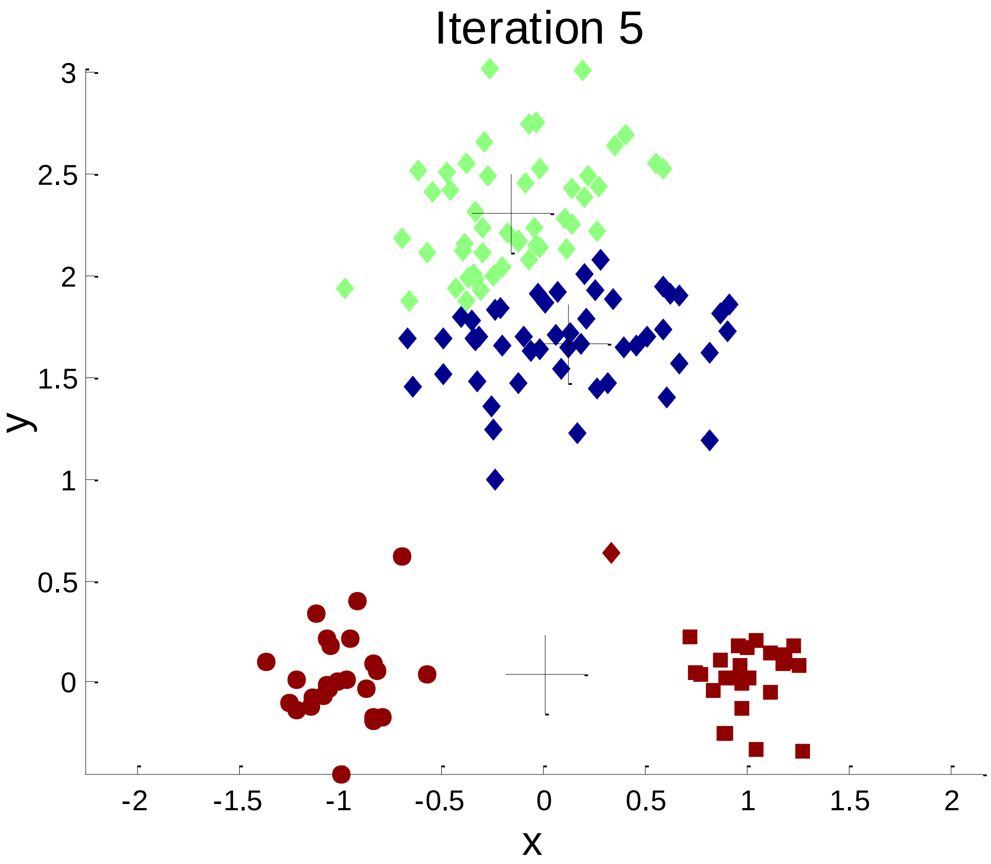


A good clustering result

Importance of Choosing Initial Centroids

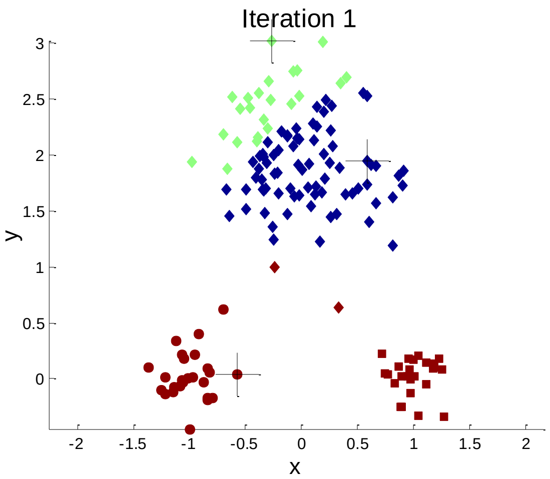


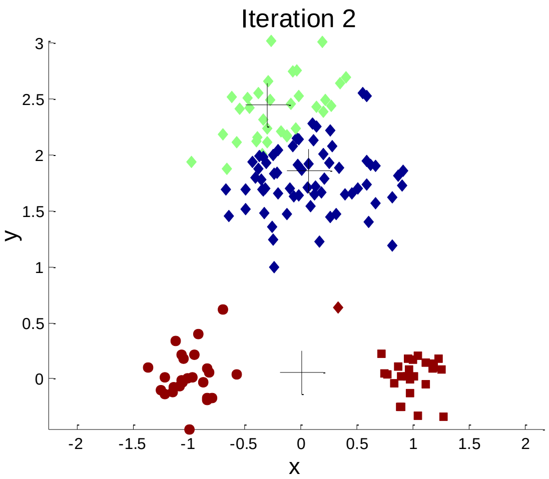
Importance of Choosing Initial Centroids …

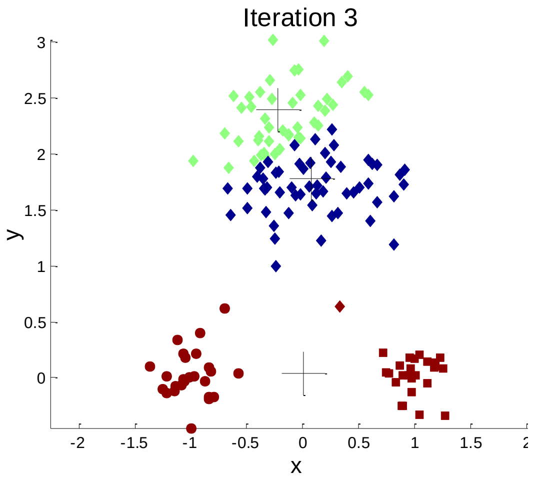


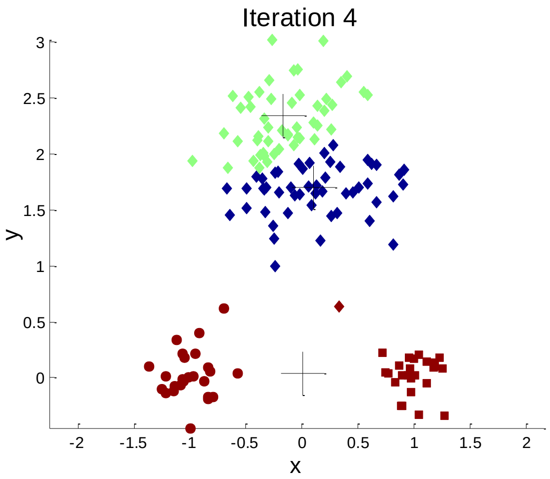
A less meaningful result

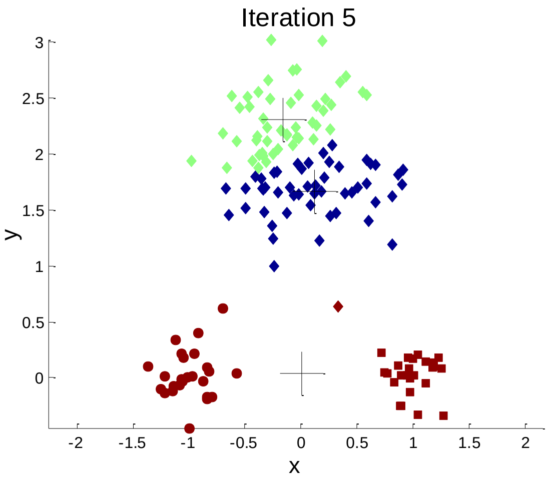
Importance of Choosing Initial Centroids …


Iteration 1


Iteration 2


Iteration 3


Iteration 4


Iteration 5

TUNING K-MEANS

Solutions to Initial Centroids Problem

Choosing the initial Centroid is very important for k-means. If we choose initial bad centroid, we will not get meaningful results

- Multiple runs, changing random seeds every time
  - Helps, but probability is not on your side, esp. for many dimensions
- Sample and use hierarchical clustering to determine initial centroids
- Select more than k initial centroids and then select among these initial centroids
- Select most widely separated

Compare SSE of different initial centroids

Most common measure is Sum of Squared Error (SSE)

- *x* is a data point in cluster C_i and m_i is the centroid/medoid for cluster C_i
- For each point, the error is the distance to the centroid/medoid
- To get SSE, we square these errors and sum them.

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

- Given two clustering results with SAME number of clusters, we can choose the one with the smallest SSE

Be careful with SSE

Attention! One easy way to reduce SSE is to increase K, the number of clusters.

- However, we must be careful. When K equals the data set size(# of clusters = # of data points in data set), each data point will end up as its own cluster, SSE=0.

Don’t simply use K to reduce SSE. K should have a reasonable range of value in real applications.

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

Syracuse University
School of Information Studies

What if the iteration never stops?

It’s possible that the k-Means algorithm could reiterate endlessly. Want can we do to control?

Set max number of iterations

Set min value of SSE change

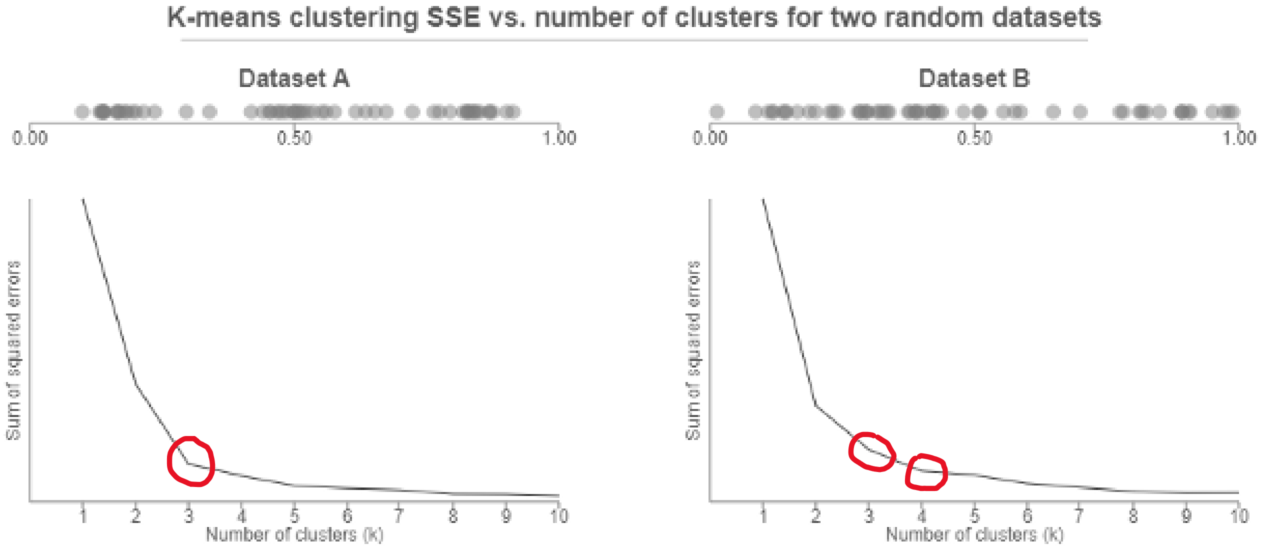
- Set slightly above 0

Choosing K

- Domain knowledge should provide a range
- Several approaches to optimizing from there
  - The Elbow method (Scree plot of SSE)
  - The Silhouette Method

The Elbow Method

1. Calculate SSE across a range of reasonable Ks
2. Plot the results
3. Pick the "elbow"



Here, it's pretty clear

Here, not so much

The Elbow Method: Limitations

Somewhat subjective

Comparing SSEs between different values of k is subject to some pitfalls

Usually a good idea to verify with another method

The Silhouette Method

The Silhouette value measures how similar a point is to its own cluster (cohesion) compared to other cluster (separation).

$$Sil(i) = \begin{cases} \dfrac{b(i) - a(i)}{\max\{a(i), b(i)\}} & |C_i| > 1 \\ 0 & |C_i| = 1 \end{cases}$$

Mean intra-cluster distance

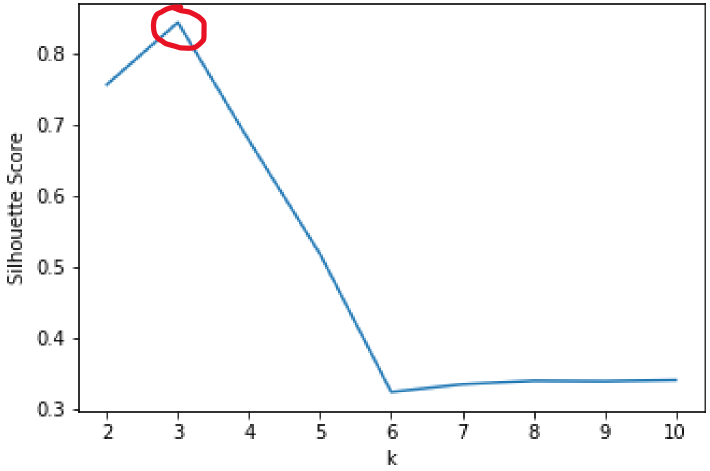
$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j)$$

Mean distance to closest cluster

$$b(i) = \min_{i \neq j} \frac{1}{|C_j|} \sum_{j \in C_j} d(i, j)$$

Using The Silhouette Method

1. Compute the average silhouette value for clusterings at each value of k.
2. Graph
3. Pick the peak



Choosing K - Summary

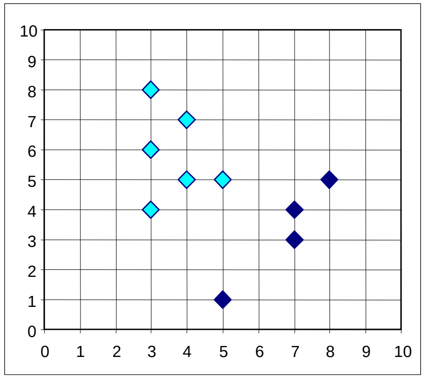
- The Elbow method and Silhouette method are complementary.
- Many other methods are available (e.g., statistical methods), and sometimes all can be applied at once using a simple clustering library.
  - Majority vote wins!
- In general, you should apply at least a couple of methods.
- As always, domain knowledge should be your guide!

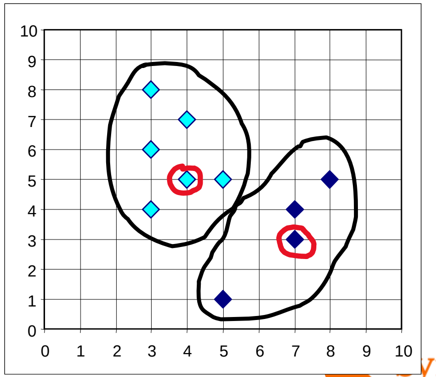
Use Medoids to resist outliers in K-means

The k-means algorithm is sensitive to outliers !

- Since an object with an extremely large value may substantially distort the distribution of the data.

K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located object (data point) in a cluster.**

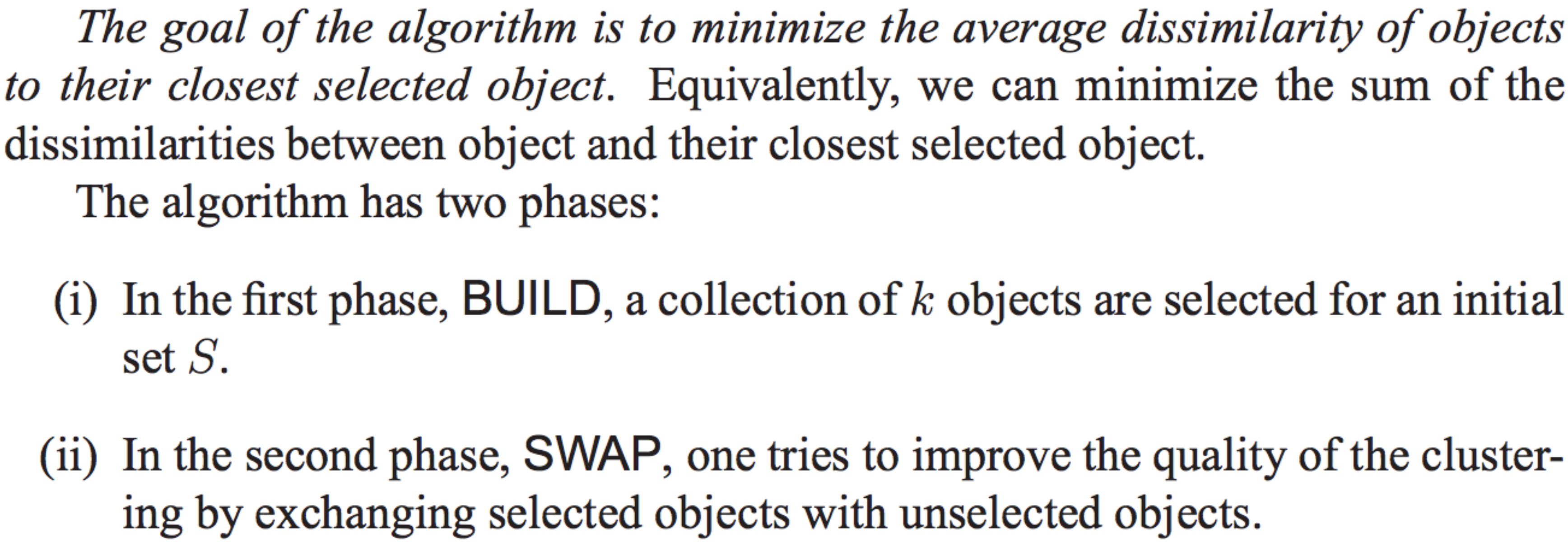




PAM: a k-Medoid algorithm

http://www.cs.umb.edu/cs738/pam1.pdf

PAM: Partition around medoids



The goal of the algorithm is to minimize the average dissimilarity of objects to their closest selected object. Equivalently, we can minimize the sum of the dissimilarities between object and their closest selected object.

The algorithm has two phases:

(i) In the first phase, BUILD, a collection of k objects are selected for an initial set S.

(ii) In the second phase, SWAP, one tries to improve the quality of the clustering by exchanging selected objects with unselected objects.

K-Medoid is an alternative algorithm to handle outliers.

Variations of the *K-Means* Method

K-Means is called a hard cluster algorithm because data points can only belong to one cluster.

One variation are the mixture models (soft clustering)

- Estimates clusters from probability distributions
- Includes the **Expectation Maximization (EM) algorithm**

Cluster Validity

For classification we have training data, so we can evaluate how good the model is. In a cluster, w do not have training so we do not know how good the model is.

For supervised classification we have a variety of measures to evaluate how good our model is

- Accuracy, precision, recall

For cluster analysis, the analogous question is how to evaluate the "goodness" of the resulting clusters?

Some say "clusters are in the eye of the beholder"! But is that our only option ?

Different Methods for Cluster Validation

Cluster Cohesion: Measures how closely related are objects in a cluster

- high intra-class similarity
- SSE as a cohesion measure

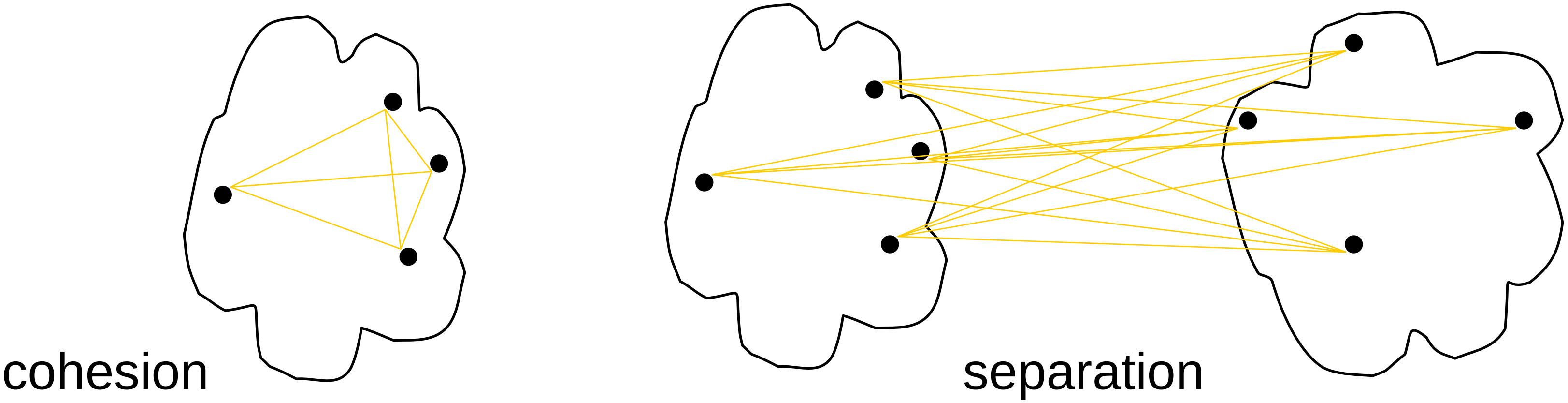
Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters

- low inter-class similarity

The Silhouette Method is one measure that combines these

Validating the Cluster results can be challenging.

- Comparing the results of a cluster analysis to externally known results
- Confirm with domain experts

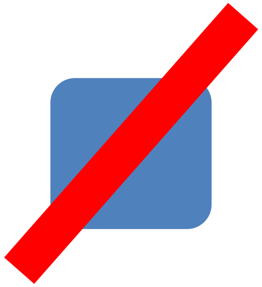


Comments on the *K-Means* Method

Strength: *Relatively efficient*:

Weakness

- Need to specify *k,* the *number* of clusters, in advance. Can be challenging.
- Sensitive noisy data and *outliers*
- Not suitable to discover clusters with *non-convex shapes*



Final Comment on Cluster Validity

"The validation of clustering structures is the most difficult and frustrating part of cluster analysis.

Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage."

Algorithms for Clustering Data, Jain and Dubes

Make sure you spend enough time to inspect clusters. Experiment with different k values to see impact on results.

Not Easy But Important

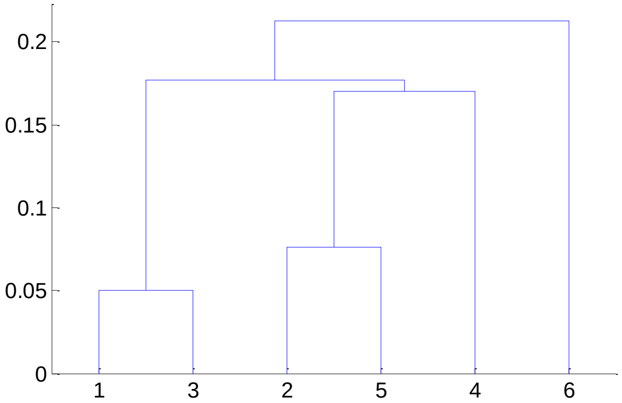
HIERARCHICAL AGGLOMERATIVE CLUSTERING

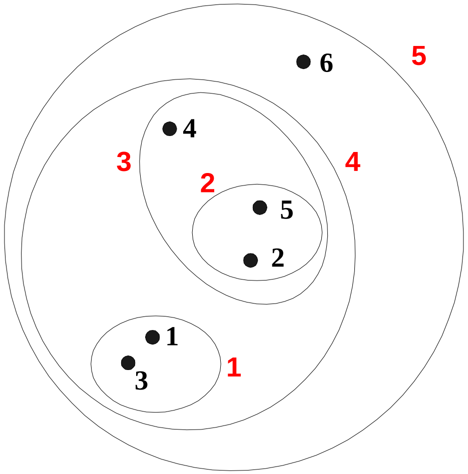
Hierarchical Clustering

Produces a set of nested clusters organized as a hierarchical tree

Can be visualized as a dendrogram

- A tree like diagram that records the sequences of merges or splits

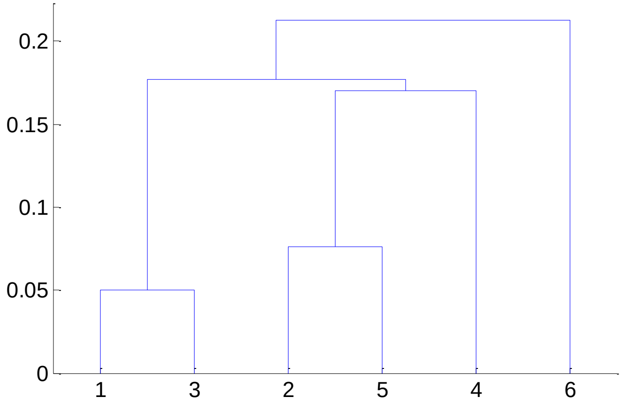




Dendrogram: Shows How the Clusters are Merged

Decompose data objects into a several levels of nested partitioning (tree of clusters), called a dendrogram.

A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.



Strengths of Hierarchical Clustering

Do not have to assume any particular number of clusters

- Any desired number of clusters can be obtained by ‘cutting’ the dendrogram at the proper level

They may correspond to meaningful taxonomies

Agglomerative Clustering Algorithm

More popular hierarchical clustering technique

Basic algorithm is straightforward

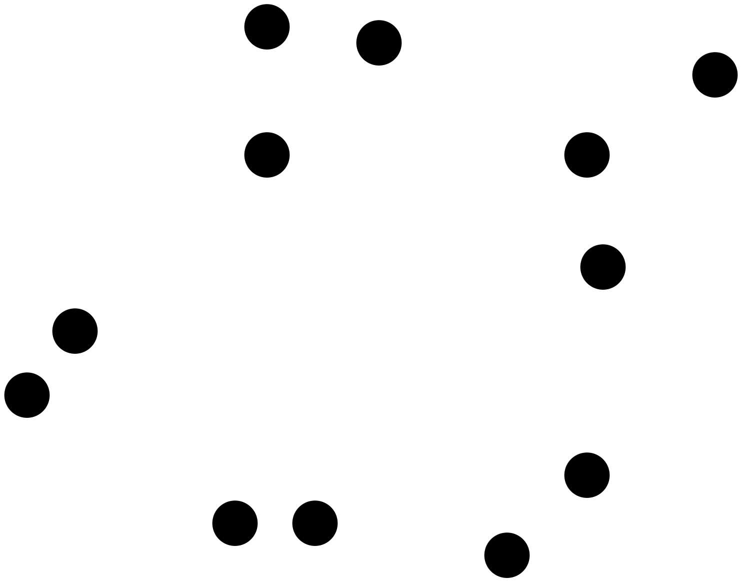
1. Let each data point be a cluster
2. Compute the distance matrix
3. **Repeat**
4. Merge the two closest clusters
5. Update the distance matrix
6. **Until** only a single cluster remains

Key operation is the computation of the distance of two clusters

- Different approaches to defining the distance between clusters distinguish the different algorithms

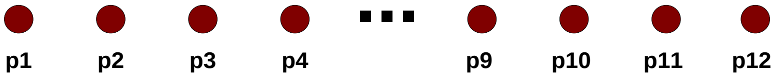
Starting Situation

Start with clusters of individual points and a distance matrix

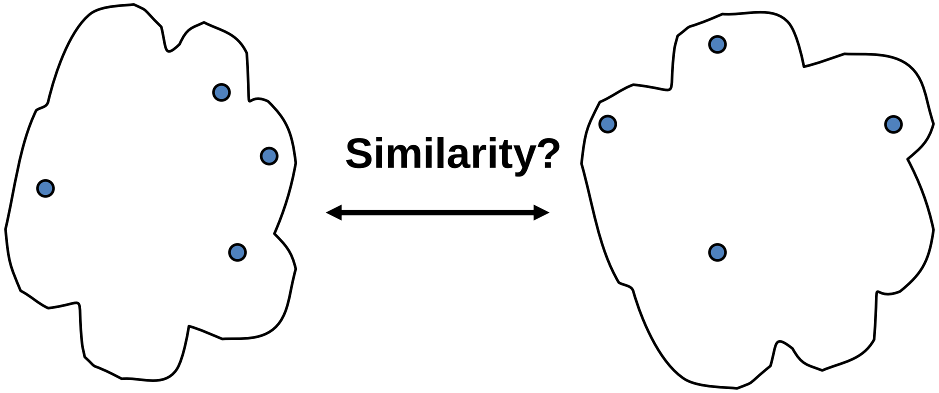


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Distance Matrix

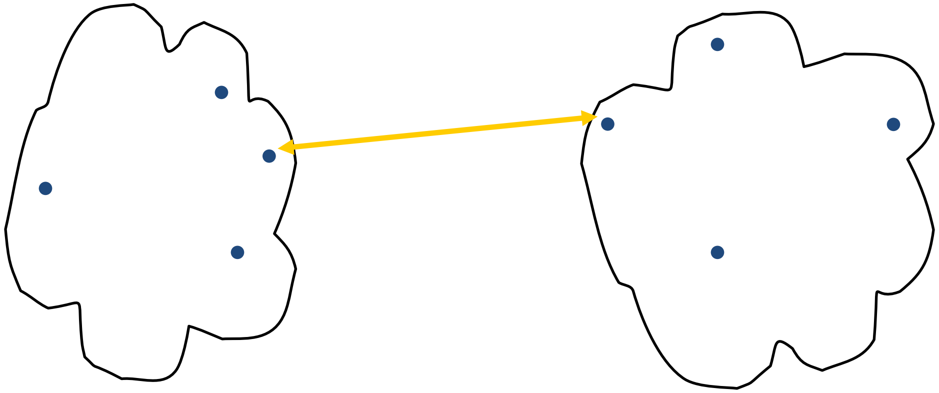


How to Define Inter-Cluster Distance



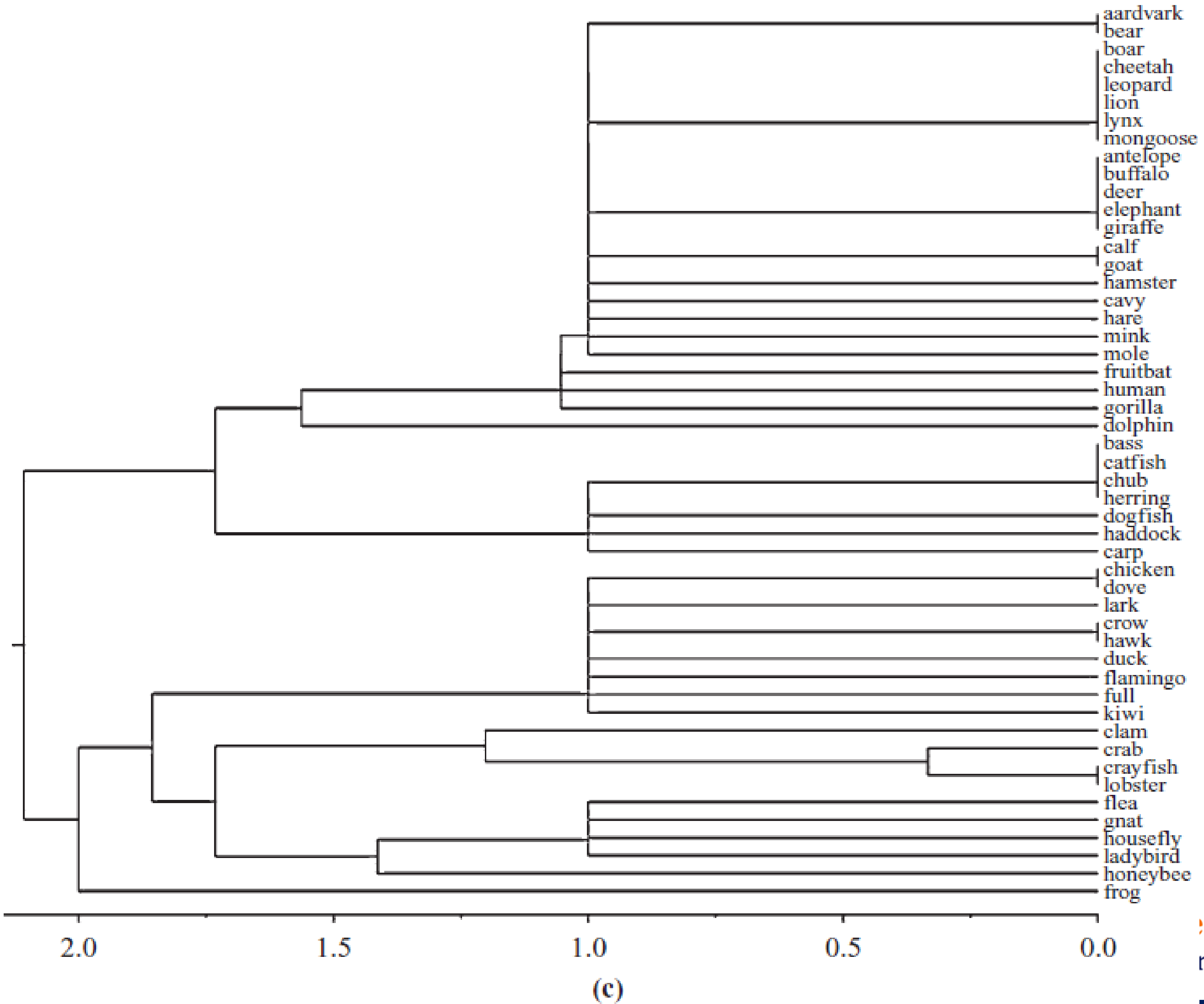
- Single-linkage
- Complete-linkage
- Average-linkage
- Centroid-linkage
- Ward's method

How to Define Inter-Cluster Distance

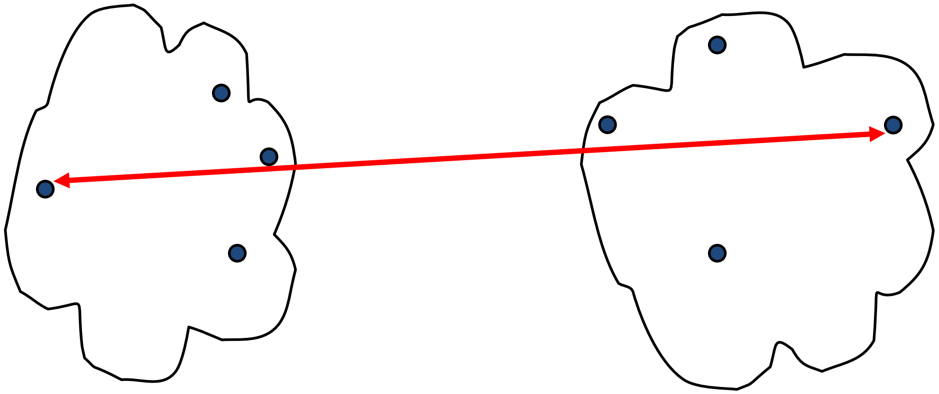


- Single-linkage
  - Minimum distance between two clusters
  - The distance between a pair of closest members
- Pros and cons
  - Depends only on the distance ordering
  - Sensitive to outliers
  - Clusters with large diameters

Single-linkage

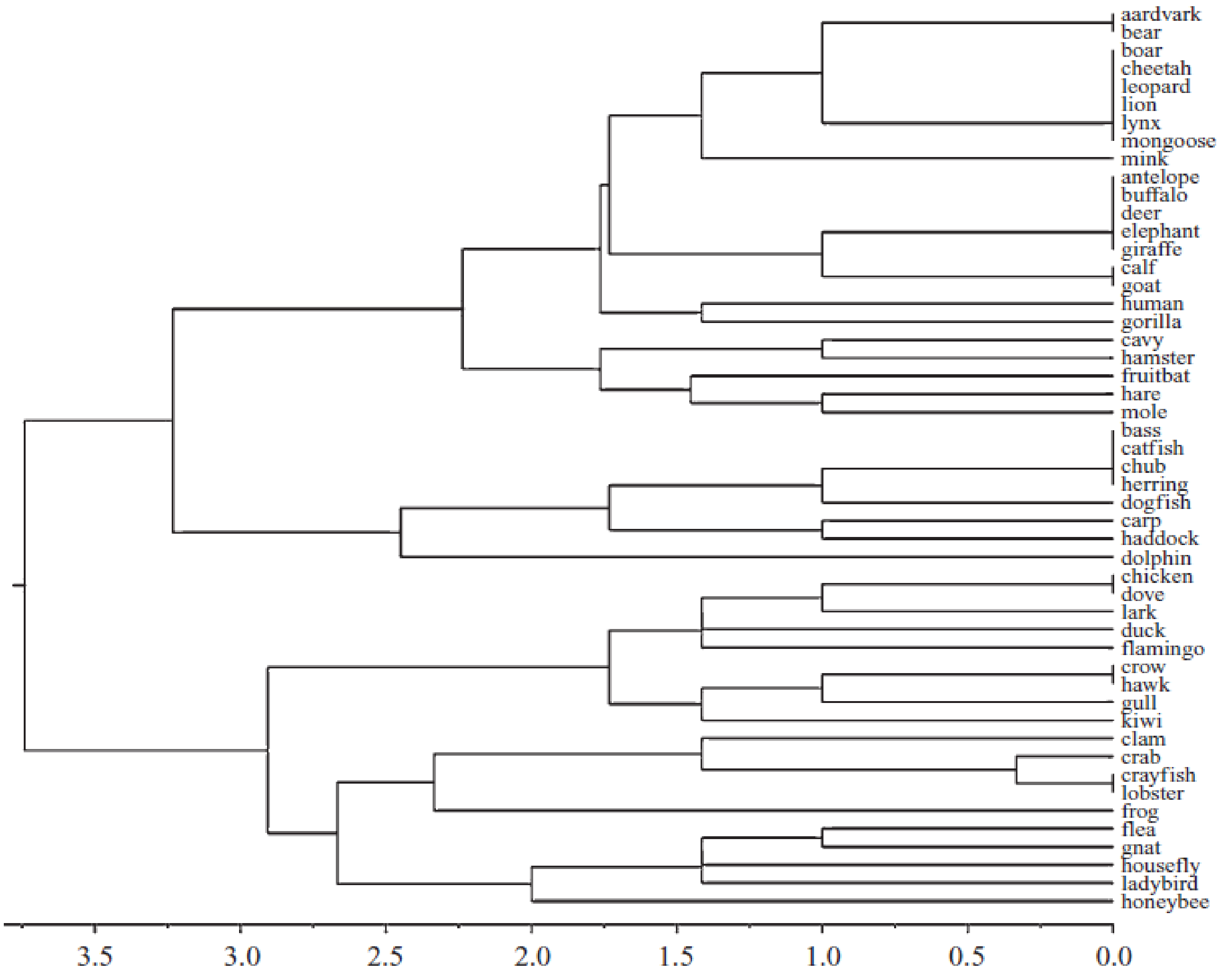


How to Define Inter-Cluster Distance



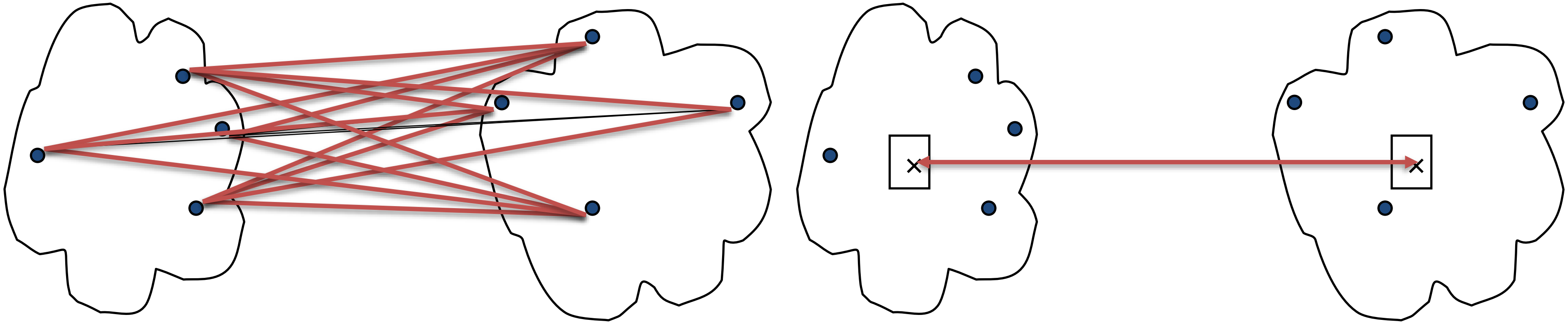
- Complete-linkage
  - Maximum distance between clusters
  - The distance between a pair of farthest members
- Pros and cons
  - Depends only on the distance ordering
  - Sensitive to outliers
  - Clusters with small diameters

Complete-linkage



(a)

How to Define Inter-Cluster Distance



- To overcome sensitivity to outliers:
  - Average-linkage: The average distance between each pair of members of the two clusters
  - Centroid-linkage: Distance between two centroids

How to Define Inter-Cluster Distance

Ward’s method: minimize the increase in SSE for each cluster

Recall: $$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

- Pros and cons
  - Creates compact, evenly sized clusters
  - Presumes clusters of similar density

Hierarchical Clustering: Problems and Limitations

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

No objective function is directly minimized–how to choose a cutoff?

Different linkage calculations 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

Note that Ward’s method and complete linkage are usually preferred

Exercise

In Orange the HierarchicalClusterer is a classic HAC algorithm.

Question 1: using the “classes to clusters evaluation”, run HierarchicalClusterer on the Iris data using single linkage and complete linkage. Which one led to better result? Can you explain why?

Question 2: compare k-Means and HAC performance in clustering animals in the zoo data set. Which algorithm gave better result (based on Orange’s class-to-cluster accuracy measure), in what parameter setting?