Principles of Computer Science II

Working with Data Sets

Marco Zecchini

Sapienza University of Rome

Lecture 8

Analysis of Data

- Viewing and analyzing vast amounts of biological data in its unstructured entirety can be perplexing.
- It is easier to interpret data if it is organized into clusters that combine similar (i.e., related) data points.

Analyzing data from DNA microarray experiments (expression analysis – i.e., determining which genes are switched "on" or "off" under certain conditions of interest).

Building and understanding evolutionary trees based on genomic or other data.

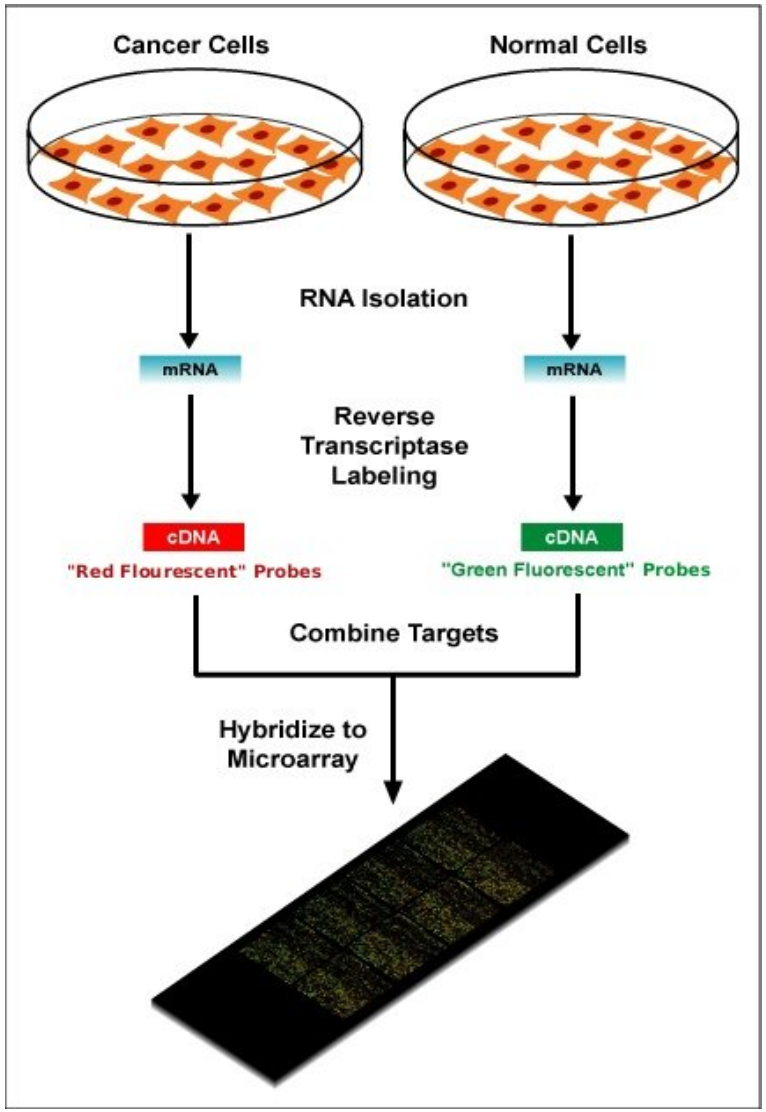
Microarray Analysis

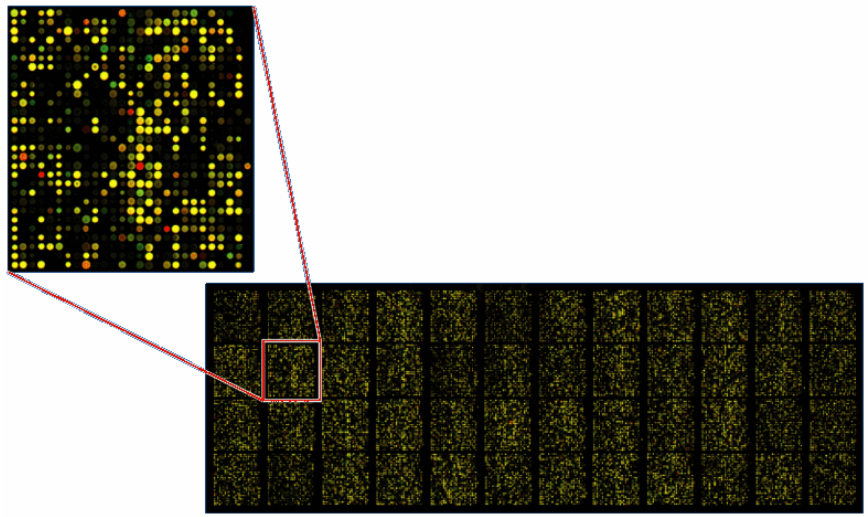
- What do newly sequenced genes do?
- Simply comparing new gene sequences to known DNA sequences often does not reveal the function of a new gene.
- For 40% of sequenced genes, functionality cannot be ascertained by comparing to sequences of other known genes.
- It is easier to interpret data if it is organized into clusters that combine similar (i.e., related) data points.

Microarrays and expression analysis

- Microarrays measure activity (expression level) of genes under varying conditions and/or points in time.
- Expression level is estimated by measuring amount of mRNA for that particular gene:
  - A gene is active if it is being transcribed.
  - More mRNA usually indicates more gene activity.

A Microarray Experiment





- Input: Biological samples (RNA) turned into fluorescent markers (cDNA), experimental conditions.
- Output: Numerical data on gene expression (matrix)

Microarray Data Transformation

- Microarray data are usually transformed into a (relative, normalized) intensity matrix
- Can also be represented as a bit matrix (log_2 of relative intensity)
- The intensity matrix allows biologists to infer correlations between different genes (even if they are dissimilar) and to understand how genes functions might be related
- Care must be taken to normalize the data appropriately, e.g. different time points can come from different arrays.

Microarray Data Intensity Matrix

- The intensity Matrix is composed by n rows one per each analyzed gene and m columns, one per each condition/time instant

Time	1 hr	2 hr	3 hr
g_1	10.0	8.0	10.0
g_2	10.0	0.0	9.0
g_3	4.0	8.5	3.0
g_4	9.5	0.5	8.5
g_5	4.5	8.5	2.5
g_6	10.5	9.0	12.0
g_7	5.0	8.5	11.0
g_8	2.7	8.7	2.0
g_9	9.7	2.0	9.0
g_{10}	10.2	1.0	9.2

(a) Intensity matrix, **I**

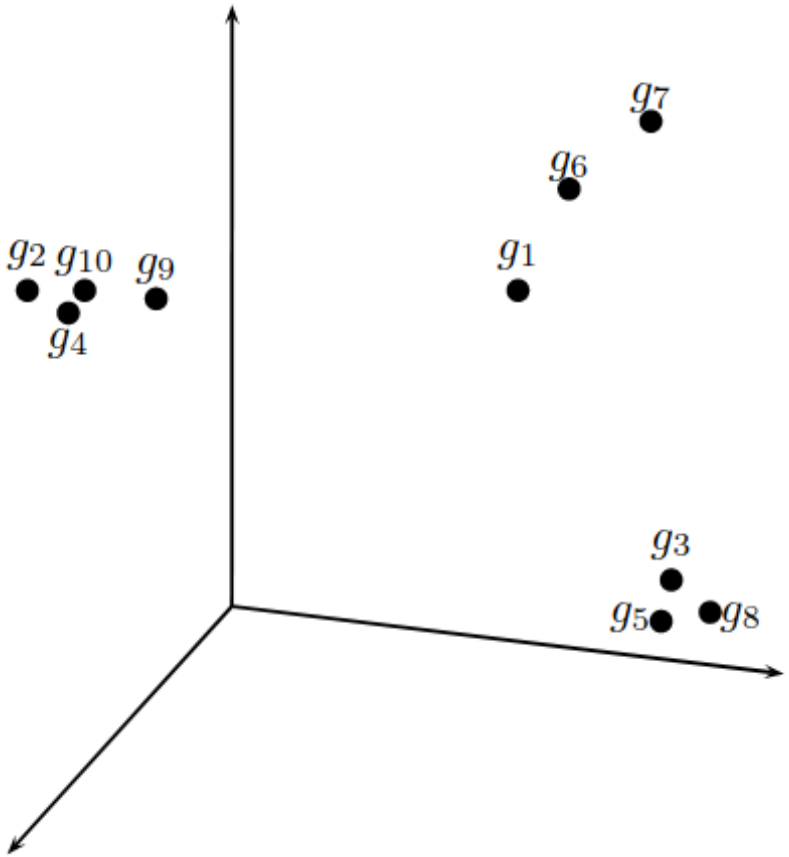
Microarray Data Intensity Matrix

- The intensity Matrix is composed by n rows one per each analyzed gene and m columns, one per each condition/time instant
- Which genes are similar?
- How to measure the distance/similarity?

Time	1 hr	2 hr	3 hr
g_1	10.0	8.0	10.0
g_2	10.0	0.0	9.0
g_3	4.0	8.5	3.0
g_4	9.5	0.5	8.5
g_5	4.5	8.5	2.5
g_6	10.5	9.0	12.0
g_7	5.0	8.5	11.0
g_8	2.7	8.7	2.0
g_9	9.7	2.0	9.0
g_{10}	10.2	1.0	9.2

(a) Intensity matrix, **I**

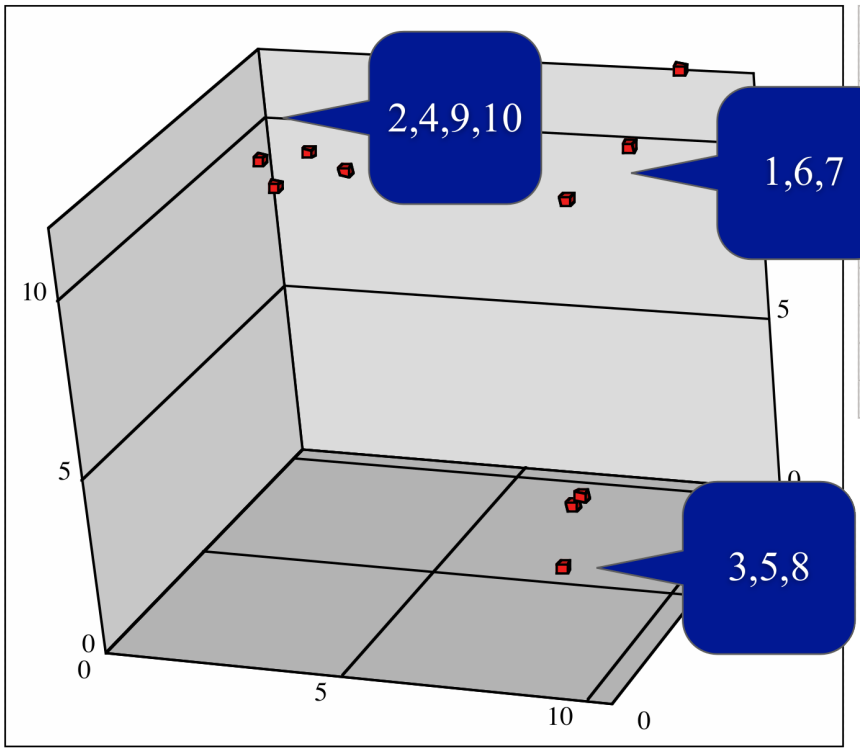
Points in a three-dimensional space



Euclidean Distance in D-dimensions

$$D(x, y) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$$

Finding Similar Genes



	1	2	3	4	5	6	7	8	9	10
1		8.1	9.2	7.7	8.9	2.3	5.1	10.9	6.1	7.0
2	8.1		12.0	0.9	11.8	9.5	10.1	13.3	2.0	1.0
3	9.2	12.0		11.2	0.5	11.1	8.1	1.7	10.5	11.5
4	7.7	0.9	11.2		10.9	9.2	9.5	12.5	1.6	1.1
5	8.9	11.8	0.5	10.9		10.8	8.0	2.1	10.3	11.3
6	2.3	9.5	11.1	9.2	10.8		5.6	12.7	7.7	8.5
7	5.1	10.1	8.1	9.5	8.0	5.6		9.3	8.3	9.3
8	10.9	13.3	1.7	12.5	2.1	12.7	9.3		12.0	12.9
9	6.1	2.0	10.5	1.6	10.3	7.7	8.3	12.0		1.1
10	7.0	1.0	11.5	1.1	11.3	8.5	9.3	12.9	1.1	

PAIRWISE DISTANCES

	1	6	7	2	4	9	10	3	5	8
1	0.0	2.3	5.1	8.1	7.7	6.1	7.0	9.2	8.9	10.9
6	2.3	0.0	5.6	9.5	9.2	7.7	8.5	11.1	10.8	12.7
7	5.1	5.6	0.0	10.1	9.5	8.3	9.3	8.1	8.0	9.3
2	8.1	9.5	10.1	0.0	0.9	2.0	1.0	12.0	11.8	13.3
4	7.7	9.2	9.5	0.9	0.0	1.6	1.1	11.2	10.9	12.5
9	6.1	7.7	8.3	2.0	1.6	0.0	1.1	10.5	10.3	12.0
10	7.0	8.5	9.3	1.0	1.1	1.1	0.0	11.5	11.3	12.9
3	9.2	11.1	8.1	12.0	11.2	10.5	11.5	0.0	0.5	1.7
6	8.9	10.8	8.0	11.8	10.9	10.3	11.3	0.5	0.0	2.1
8	10.9	12.7	9.3	13.3	12.5	12.0	12.9	1.7	2.1	0.0

REARRANGED DISTANCES

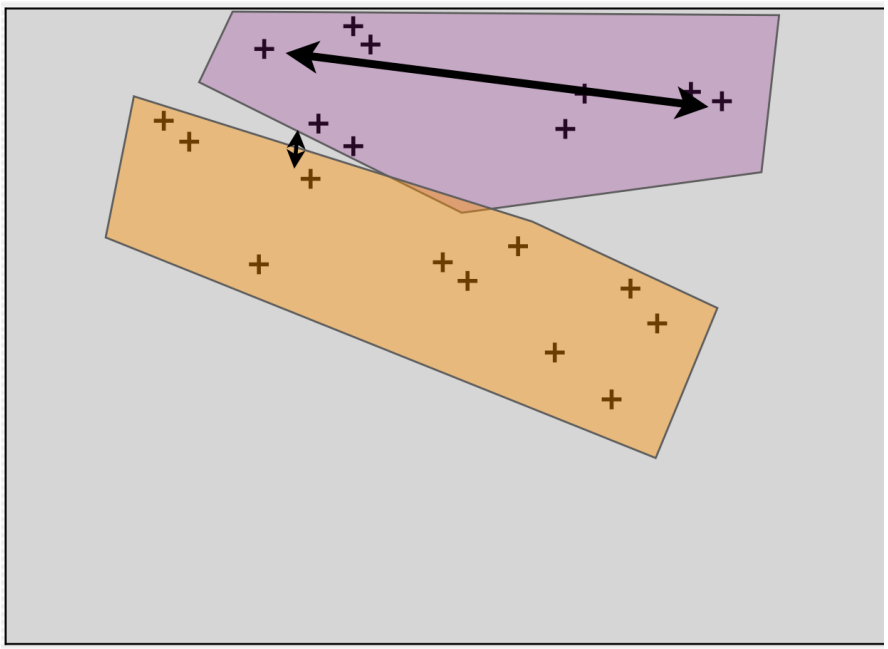
The Clustering Problem

- Motivation: Find patterns in a sea of data
- Input
  - A (large) number of datapoints: N
  - A measure of distance between any two data points d_{ij}
- Output
  - Groupings (clustering) of the elements into K (the number can be user-specified or automatically determined) 'similarity' classes
  - Sometimes there is also an objective measure that the obtained clustering seeks to minimize.

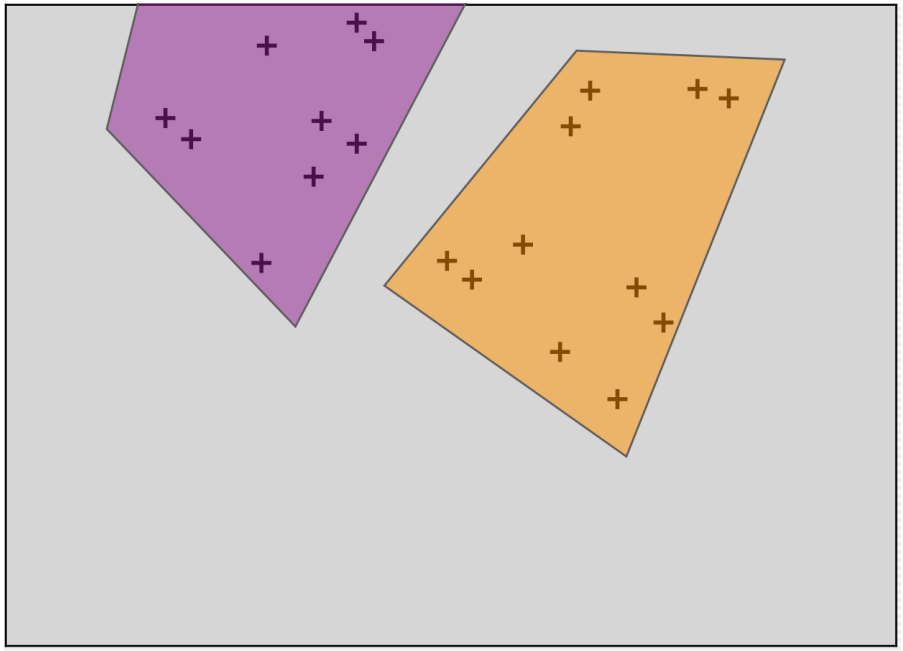
Clustering Principles

- Homogeneity – elements of the same cluster are maximally close to each other.
- Separation – elements in separate clusters are maximally far apart from each other.
- One is actually implied by the other (in many cases).
- Generally it is a hard problem.
  - Clustering in 2 dimensions looks easy
  - Clustering small amounts of data looks easy
  - High-dimensional spaces look different – Almost all pairs of points are at about the same distance

Some Examples

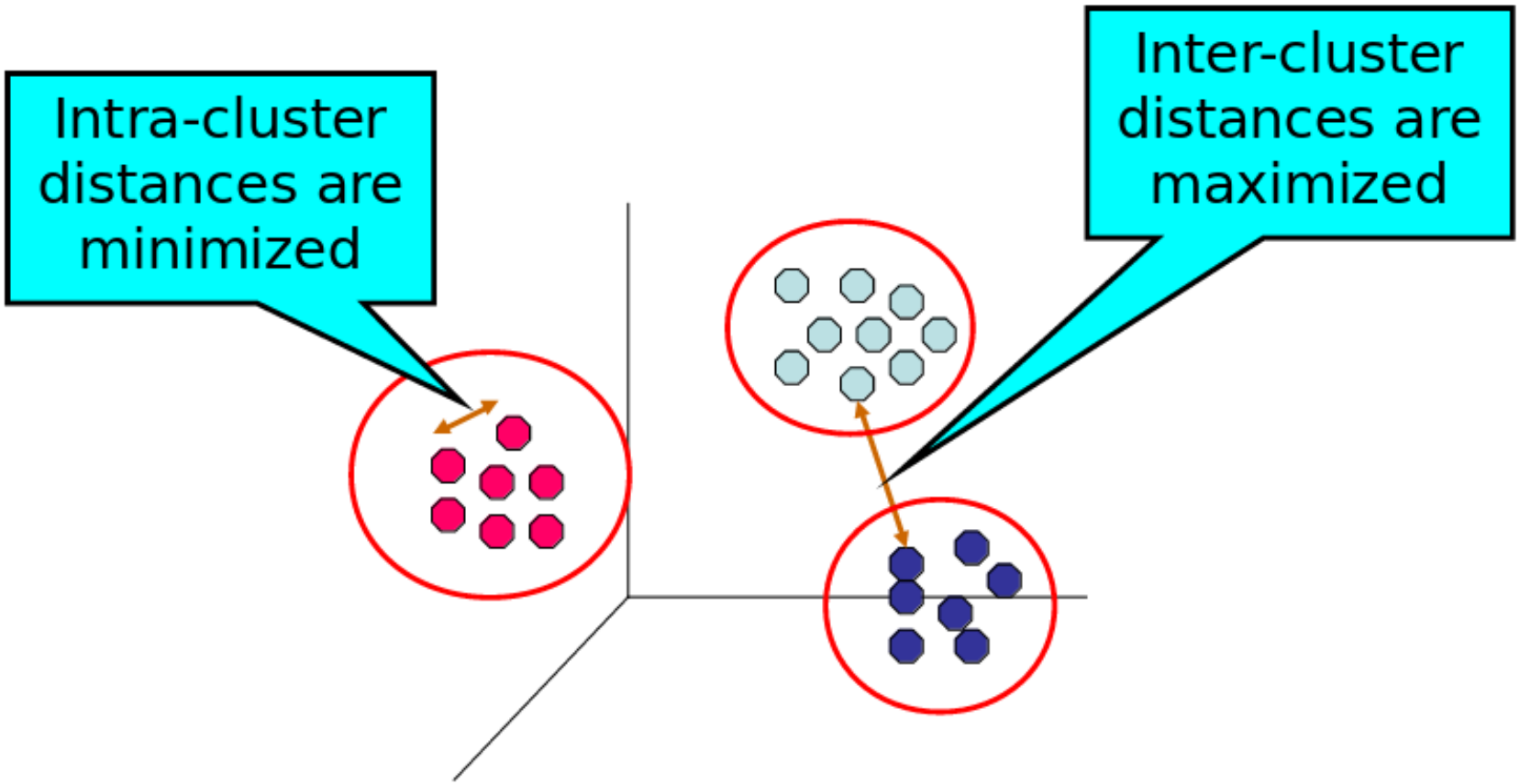


- Both principles are violated
- Points in the same cluster are far apart
- Points in different cluster are close



- More reasonable assignment.
- We need to use an objective function to optimize cluster assignment.

Intra/Inter Cluster Distances



- Suitably select distance metric.
- Maximize Inter-cluster distances.
- Minimize Intra-cluster distances.

Distance Measures

- Each clustering problem is based on some kind of "distance" between points.
- Two major classes of distance measure:
  1. Euclidean
  2. Non-Euclidean
- A Euclidean space has some number of real-valued dimensions.
  - There is a notion of "average" of two points.
  - A Euclidean distance is based on the locations of points in such a space.
- A Non-Euclidean distance is based on properties of points, but not their "location" in a space.

Axioms of a Distance Measure

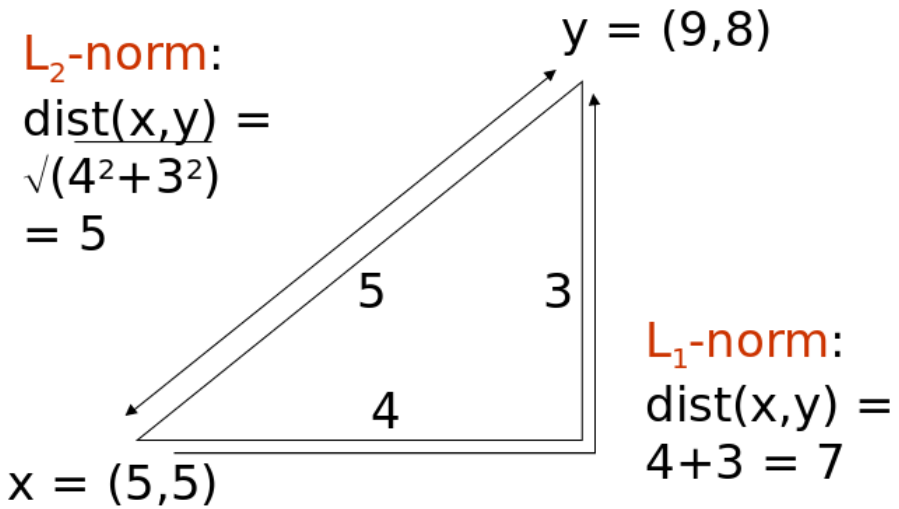
d is a distance measure if it is a function from pairs of points to real numbers such that:

1. $d(x, y) > 0$
2. $d(x, y) = 0$ *iff* $x = y$
3. $d(x, y) = d(y, x)$
4. $d(x, y) < d(x, z) + d(z, y)$ (triangle inequality)

Some Euclidean Distances

L_2-norm: $d(x, y)$ = square root of the sum of the squares of the differences between x and y in each dimension.
The most common notion of "distance".

L_1-norm: sum of the differences in each dimension.
Manhattan distance = distance if you had to travel along coordinates only.



Some Non-Euclidean Distances

Jaccard distance for sets = 1 minus ratio of sizes of intersection and union.

Edit distance = number of inserts and deletes to change one string into another.

Jaccard Distance for Sets

Example: $p_1 = 10111$; $p_2 = 10011$.
Size of intersection $= 3$; size of union $= 4$, Jaccard similarity (not distance) $= \frac{3}{4}$.
$d(x, y) = 1$–(Jaccard similarity) $= \frac{1}{4}$.

Why JD is a distance measure?

1. $d(x, x) = 0$ because $x \cap x = x \cup x$
2. $d(x, y) = d(y, x)$ because union and intersection are symmetric
3. $d(x, y) \geq 0$ because $|x \cap y| \leq |x \cup y|$
4. $d(x, y) < d(x, z) + d(z, y)$ more difficult...
   $\left(1 - \frac{|x \cap z|}{|x \cup z|}\right) + \left(1 - \frac{|y \cap z|}{|y \cup z|}\right) \geq 1 - \frac{|x \cap y|}{|x \cup y|}$

Edit Distance

The edit distance of two strings is the number of inserts and deletes of characters needed to turn one into the other. Equivalently:

$$d(x, y) = |x| + |y| - 2|LCS(x, y)|$$

LCS = longest common subsequence = any longest string obtained both by deleting from x and deleting from y.

Example

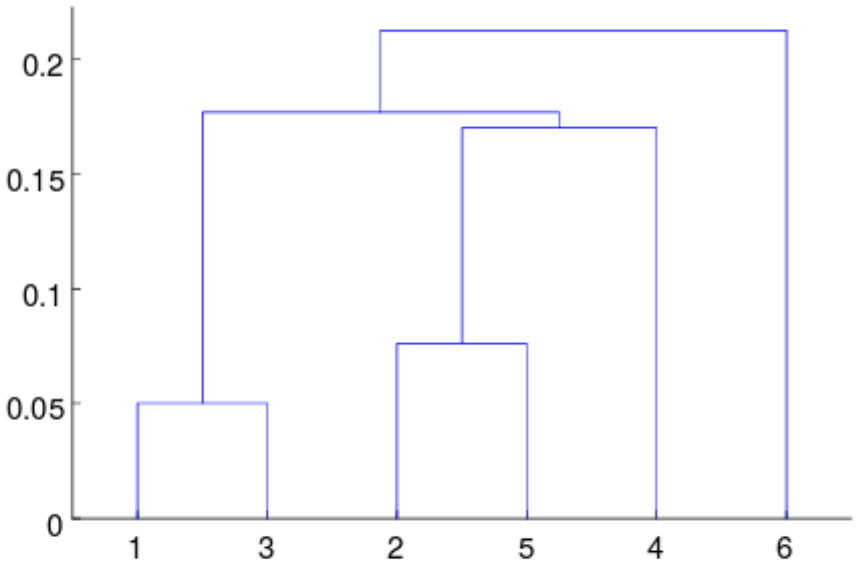
- x = abcde ; y = bcduve.
- Turn x into y by deleting a, then inserting u and v after d. Edit distance = 3.
- Or, LCS(x,y) = bcde.
- Note: $|x| + |y| - 2|LCS(x, y)| = 5 + 6 - 2 \times 4 = 3 =$ edit dist

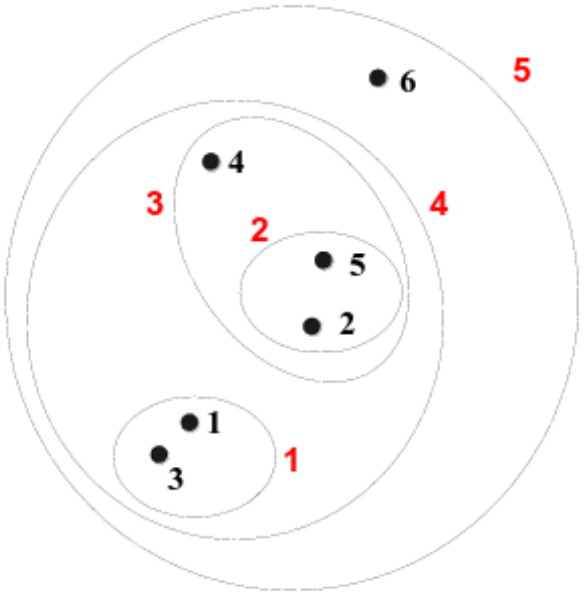
Why Edit Distance is a Distance Measure?

1. $d(x, x) = 0$ because 0 edits suffice.
2. $d(x, y) = d(y, x)$ because insert/delete are inverses of each other
3. $d(x, y) \geq 0$ no notion of negative edits
4. $d(x, y) < d(x, z) + d(z, y)$ Triangle inequality: changing x to z and then to y is one way to change x to y.

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





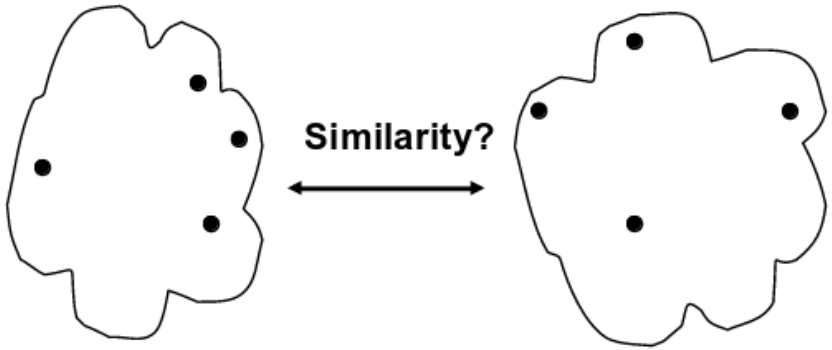
Agglomerative Hierarchical Clustering

- Initially, each point is a cluster
- Key operation: computation of the proximity of two clusters
- Different approaches to defining the distance between clusters distinguish the different algorithms

HIERARCHICALCLUSTERING($\mathbf{d}, n$) d = distance matrix

1 Form n clusters, each with 1 element n points
2 Construct a graph T by assigning an isolated vertex to each cluster
3 **while** there is more than 1 cluster
4 Find the two closest clusters C_1 and C_2
5 Merge C_1 and C_2 into new cluster C with $|C_1| + |C_2|$ elements
6 Compute distance from C to all other clusters
7 Add a new vertex C to T and connect to vertices C_1 and C_2
8 Remove rows and columns of $\mathbf{d}$ corresponding to C_1 and C_2
9 Add a row and column to $\mathbf{d}$ for the new cluster C
10 **return** T

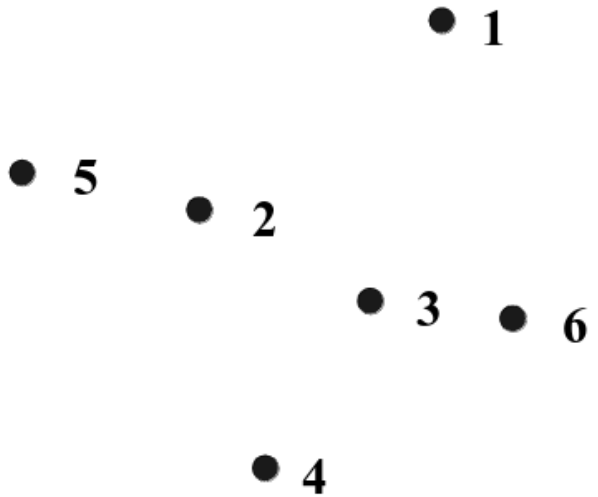
How to define Inter-cluster similarity?

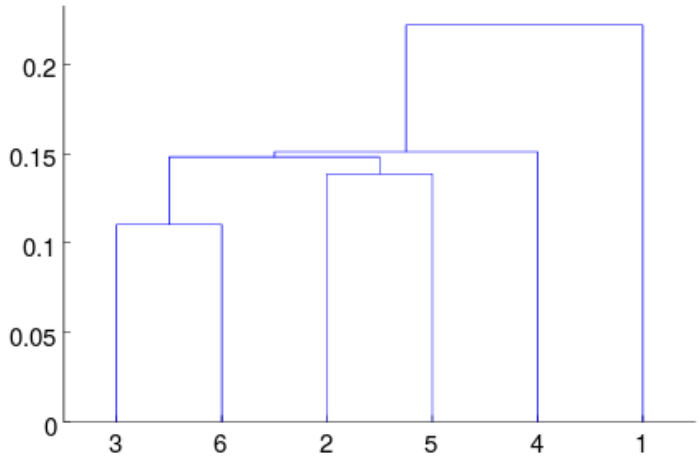


- Minimum, based on the two most similar (closest) points in different clusters: $d_{\min(C^*,C)} = \min_{x \in C^*, y \in C} d(x,y)$
- Maximum, based on the two least similar (most distant) points in different clusters: $d_{\max(C^*,C)} = \max_{x \in C^*, y \in C} d(x,y)$
- Group Average: $d_{\text{avg}(C^*,C)} = \frac{1}{|C^*||C|} \sum_{x \in C^*, y \in C} d(x,y)$

Minimum – Example

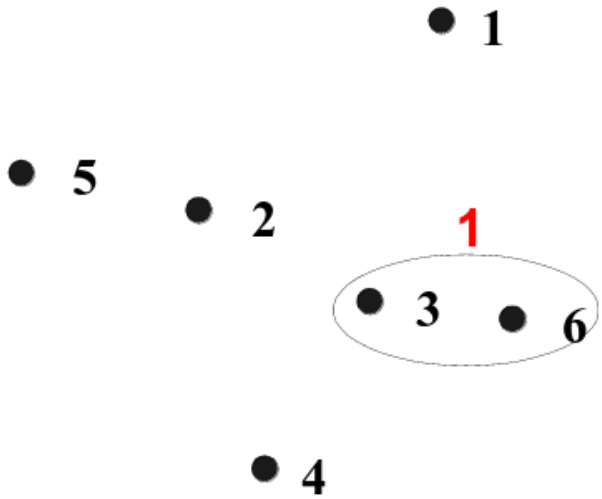
Minimum – based on the two most similar (closest) points in the different clusters

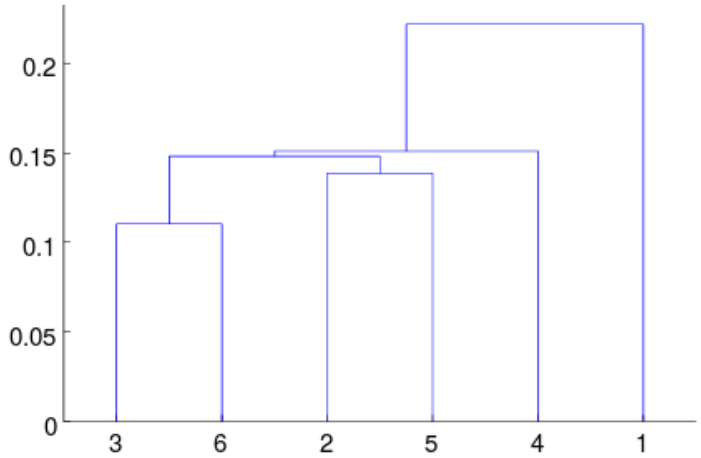




Minimum – Example

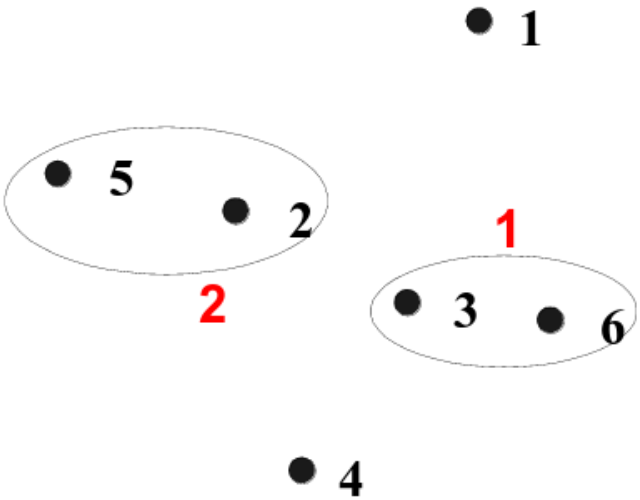
Minimum – based on the two most similar (closest) points in the different clusters

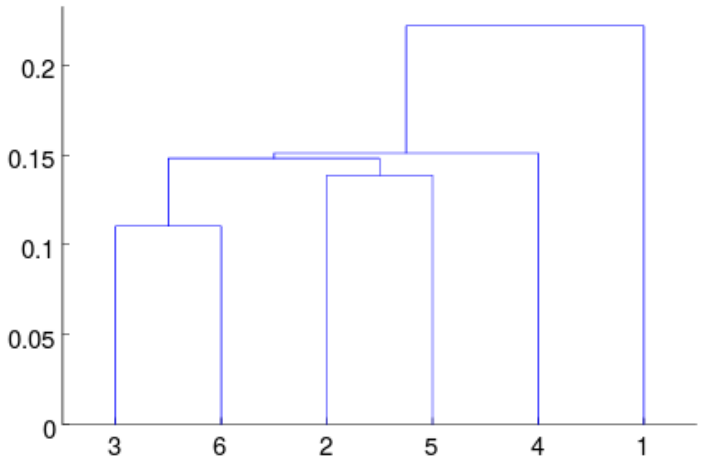




Minimum – Example

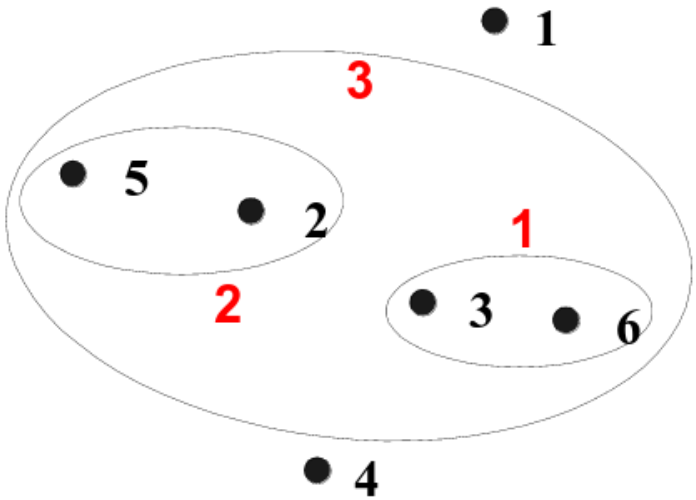
Minimum – based on the two most similar (closest) points in the different clusters

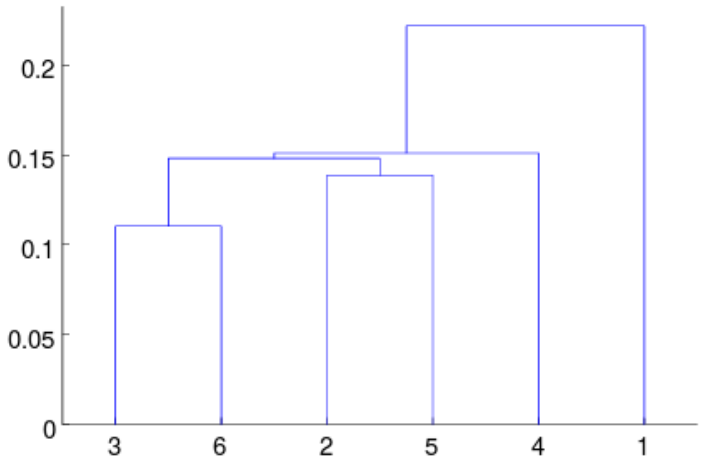




Minimum – Example

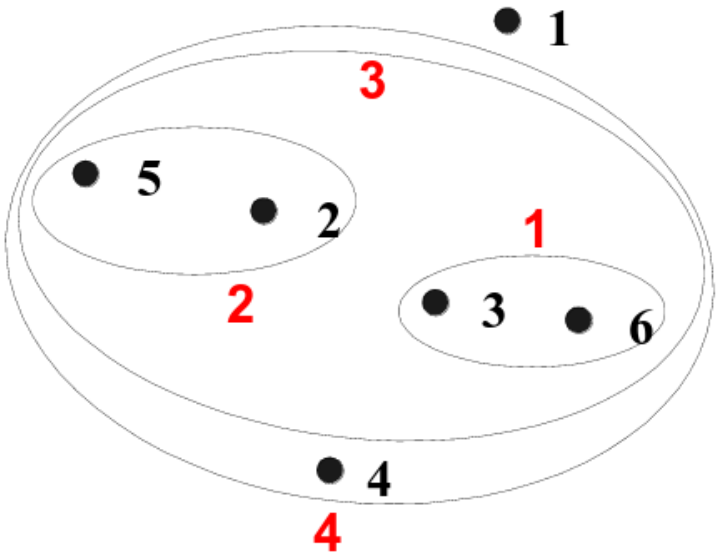
Minimum – based on the two most similar (closest) points in the different clusters

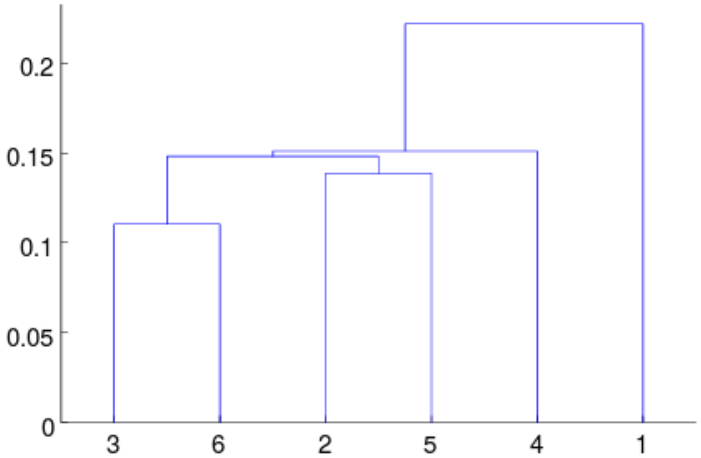




Minimum – Example

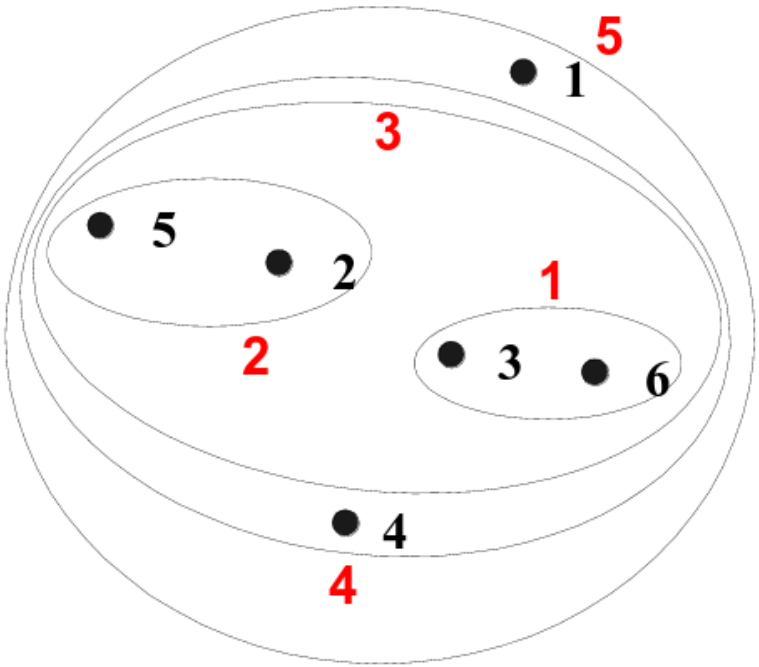
Minimum – based on the two most similar (closest) points in the different clusters

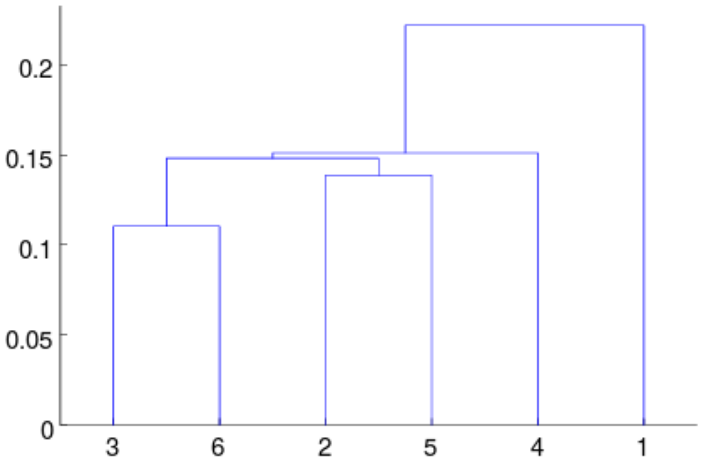




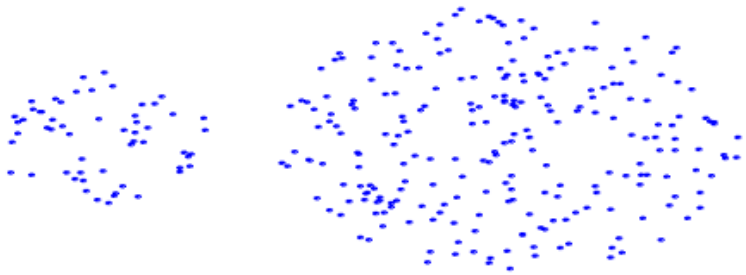
Minimum – Example

Minimum – based on the two most similar (closest) points in the different clusters

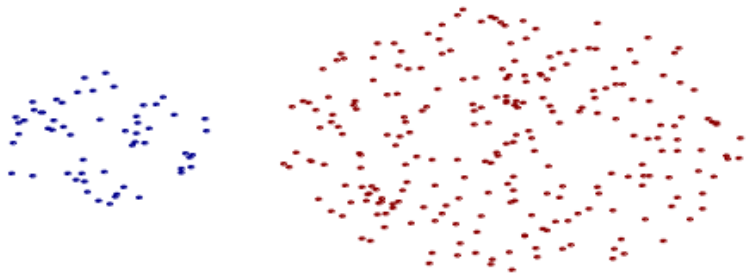




Minimum – Strength

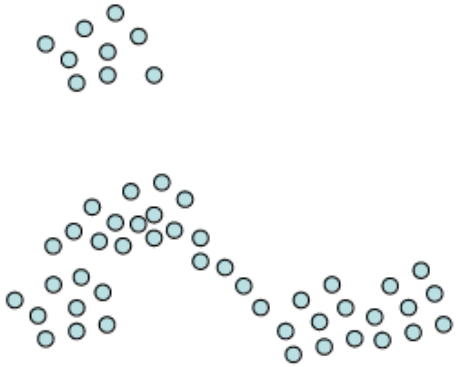


Original Points

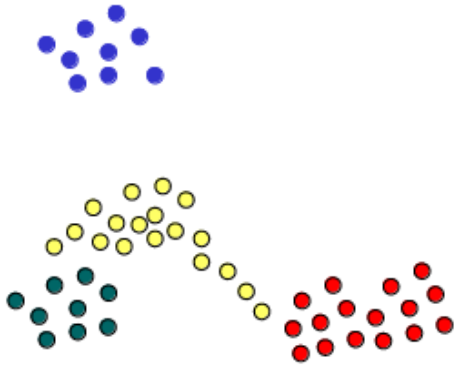


Two Clusters

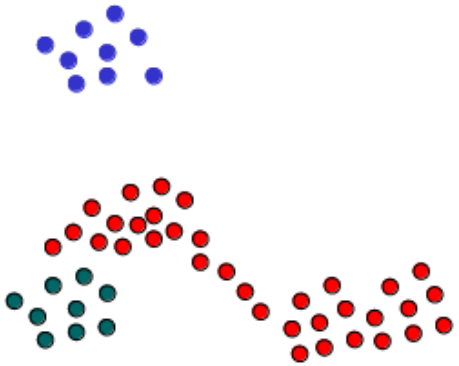
Minimum – Limitations



Original Points



Four clusters



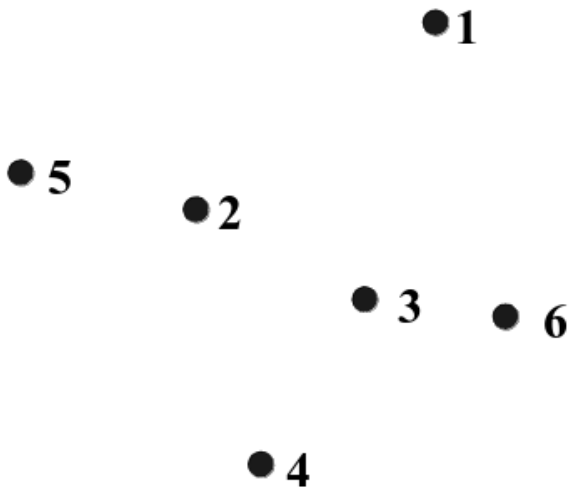
Three clusters:

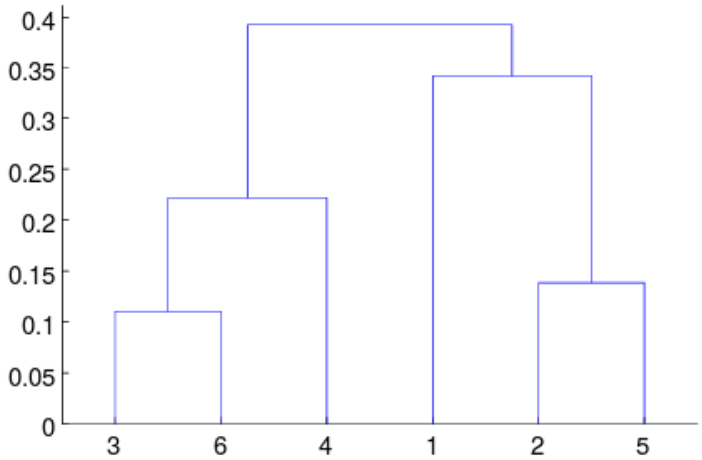
The yellow points got wrongly merged with the red ones, as opposed to the green one.

Sensitive to noise and outliers

Maximum – Example

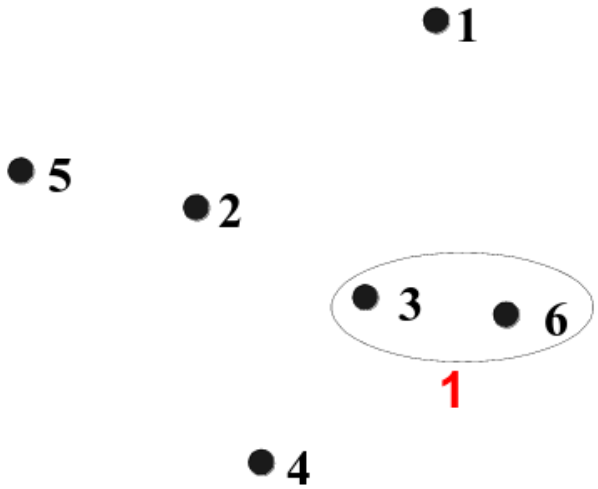
Maximum – based on the two least similar (most distant) points in the different clusters

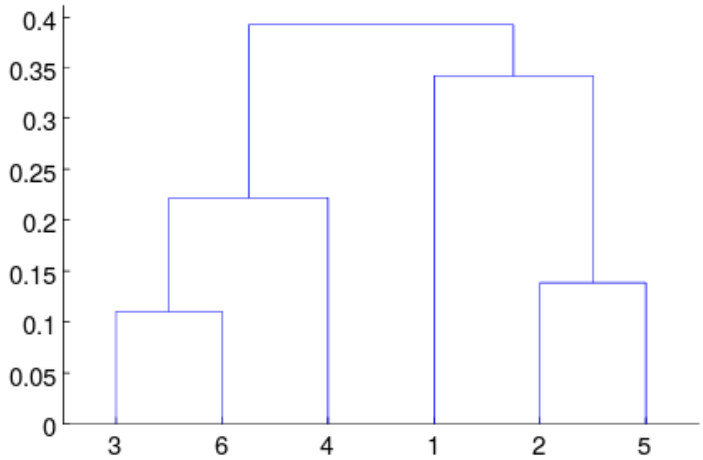




Maximum – Example

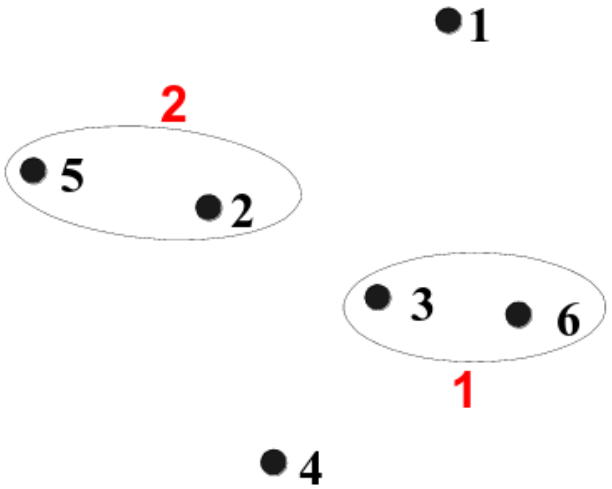
Maximum – based on the two least similar (most distant) points in the different clusters

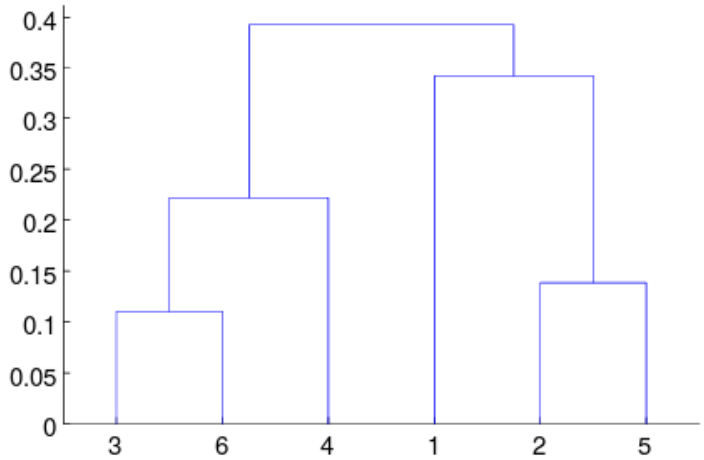




Maximum – Example

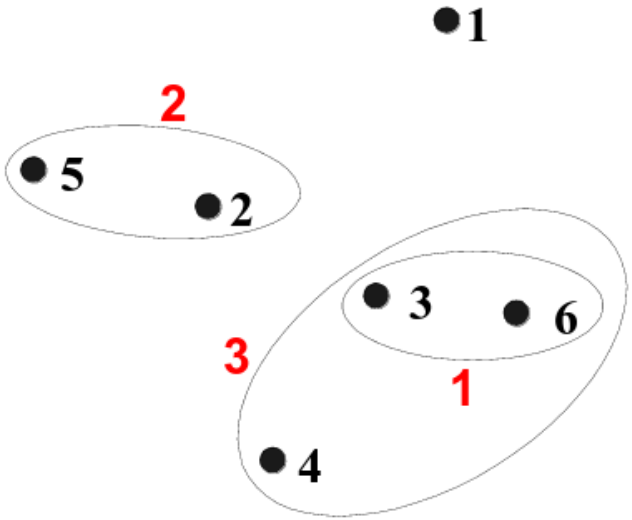
Maximum – based on the two least similar (most distant) points in the different clusters

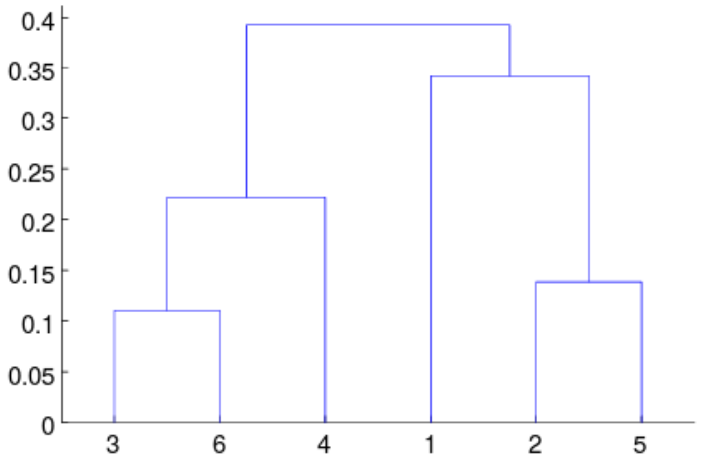




Maximum – Example

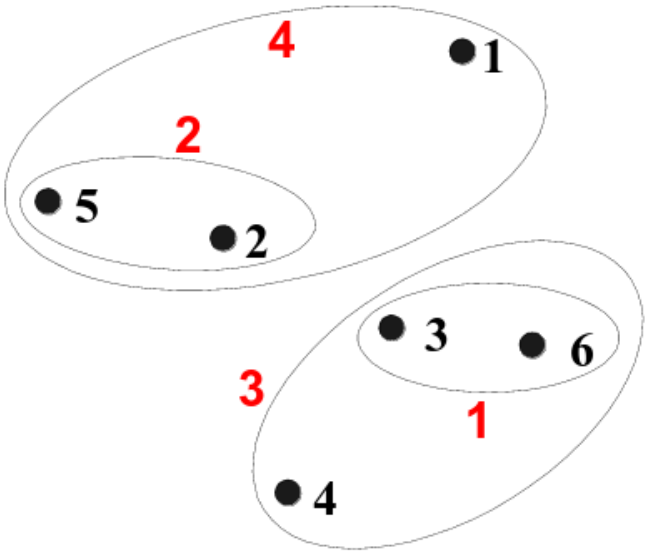
Maximum – based on the two least similar (most distant) points in the different clusters

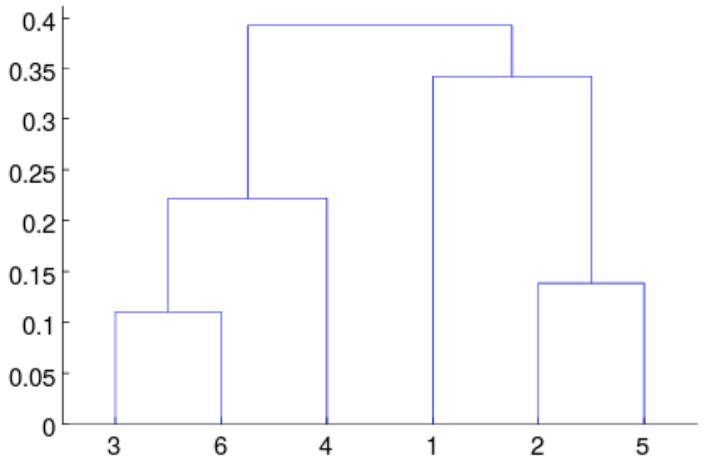




Maximum – Example

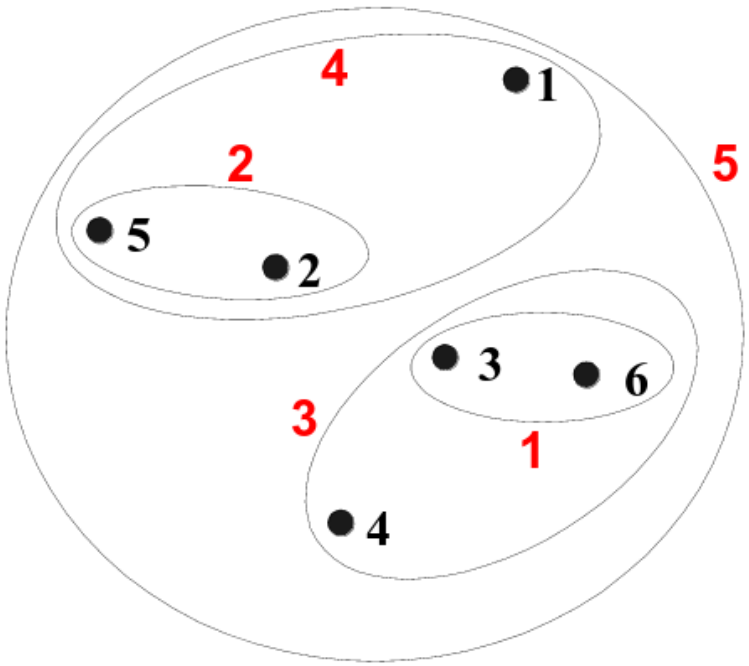
Maximum – based on the two least similar (most distant) points in the different clusters

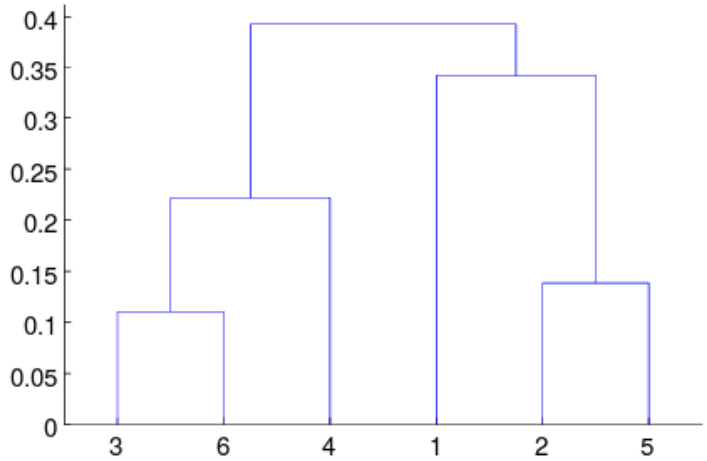




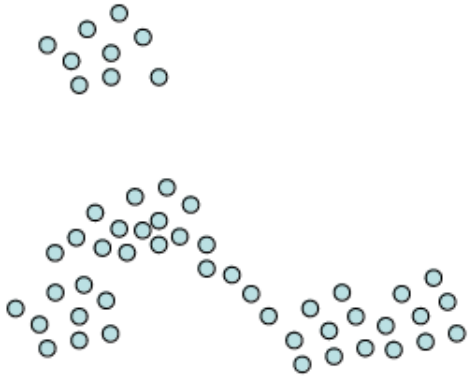
Maximum – Example

Maximum – based on the two least similar (most distant) points in the different clusters

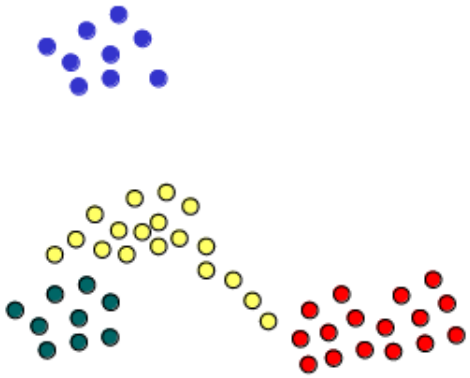




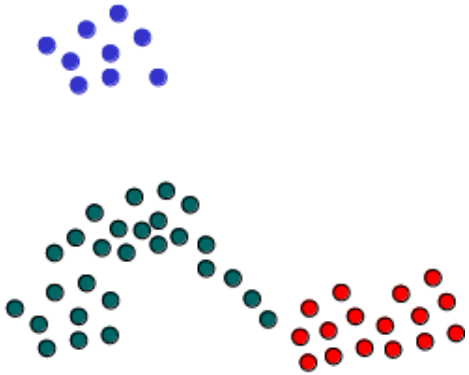
Maximum – Strength



Original Points



Four clusters

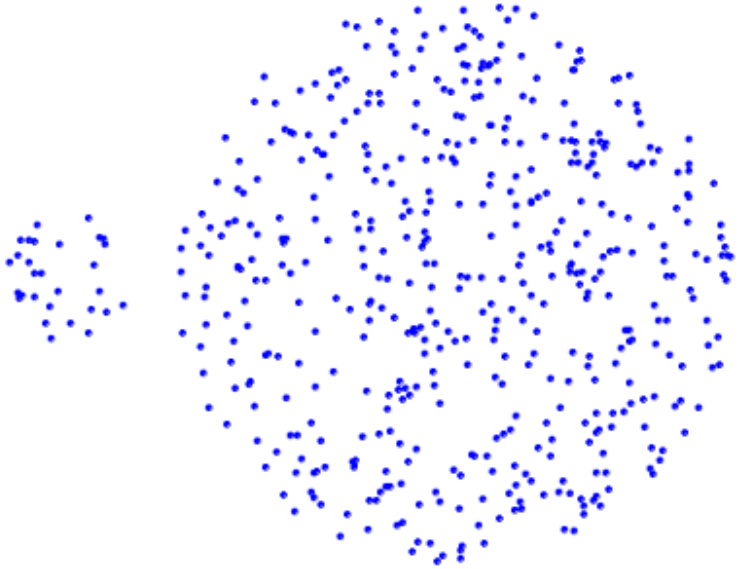


Three clusters:

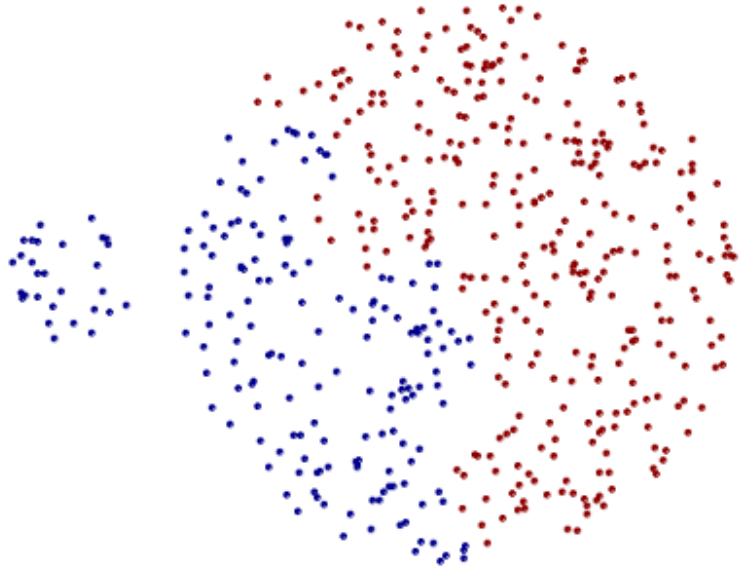
The yellow points get now merged with the green one.

Less susceptible respect to noise and outliers

Maximum – Limitations



Original Points



Two Clusters

K-means Algorithm

- Developed and published in Applied Statistics by Hartigan and Wong, 1979.
- Many variations have been proposed since then.
- Standard/core function of R, Python, Matlab, ...
- Assumes Euclidean space/distance

Given a set of n points in m dimensions $\mathcal{V} = \{x_1, \ldots, x_n\}$ and a set of k centers of $\mathcal{X} = \{x_1, \ldots, x_k\}$, the aim of the K-means algorithm is to divide n points into k clusters so that the following within-cluster sum of squares is **minimized**:

$$d(\mathcal{V}, \mathcal{X}) = \frac{\sum_{i \in C_k, i' \in \mathcal{X}} \sum_{j=1}^{m} (x_{ij} - x_{i'j})^2}{n}$$

Cluster Initialization

- Start by picking k, the number of clusters
- Initialize clusters by picking one point per cluster

Example: Pick one point at random, then $k - 1$ other points, each as far away as possible from the previous points

Populating Clusters

1. For each point, place it in the cluster whose current centroid it is nearest
2. After all points are assigned, update the locations of centroids of the k clusters
3. Reassign all points to their closest centroid
   - Sometimes moves points between clusters
4. Repeat 2 and 3 until convergence

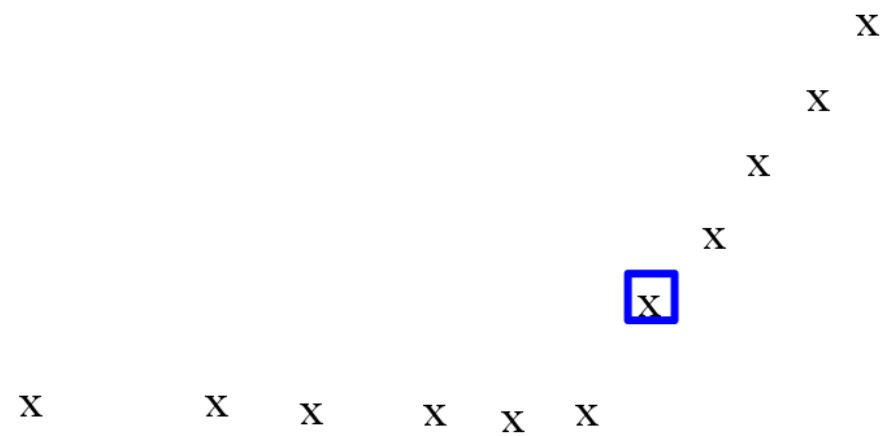
Convergence: Points do not move between clusters and centroids stabilize

A Simple Example

x … data point
□ … centroid

Clusters after round 1

A Simple Example



x … data point
□ … centroid

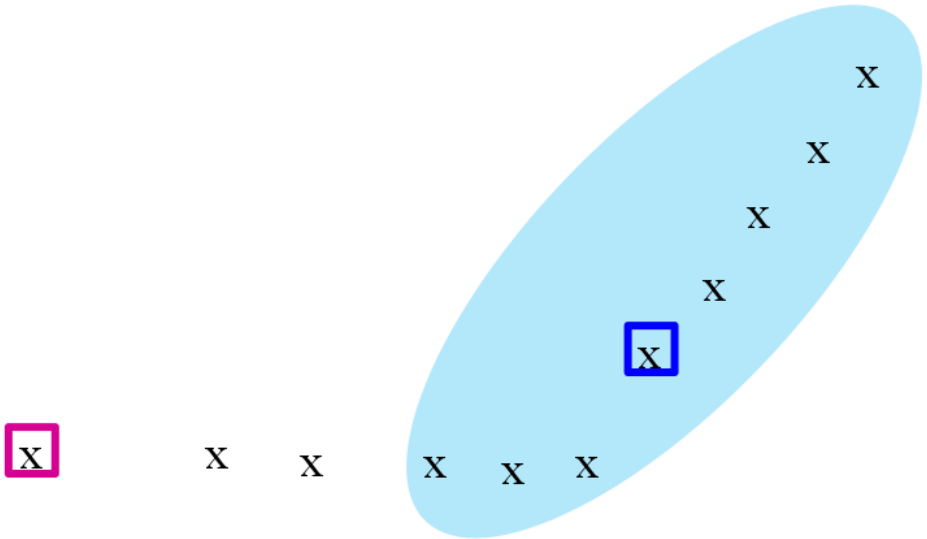
Clusters after round 1

A Simple Example

x … data point
□ … centroid

Clusters after round 1

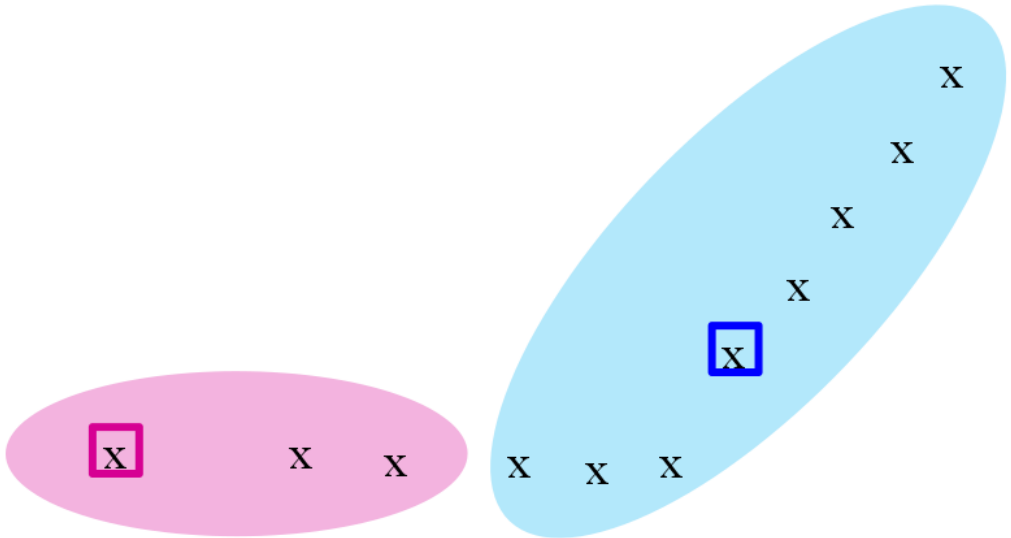
A Simple Example



x … data point
□ … centroid

Clusters after round 1

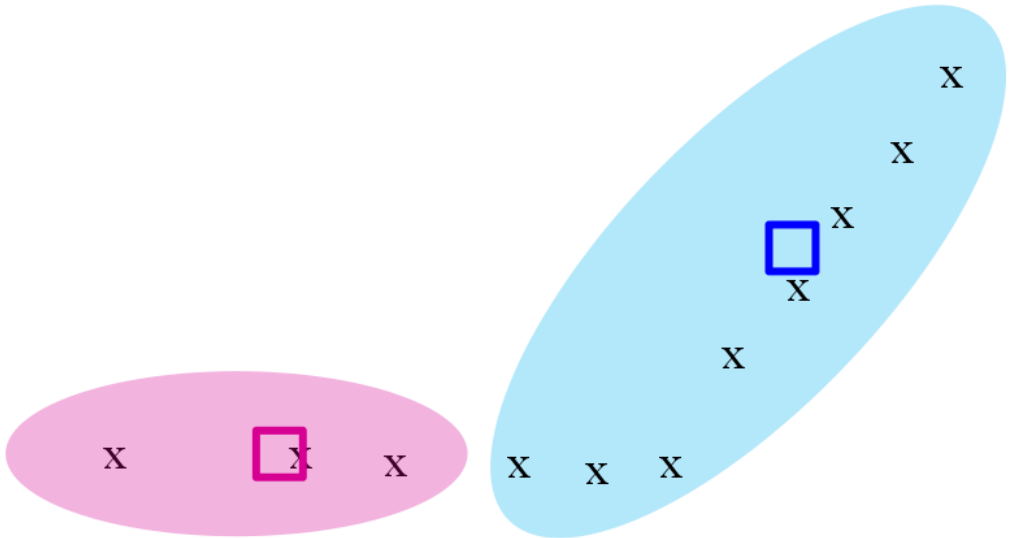
A Simple Example



x … data point
□ … centroid

Clusters after round 1

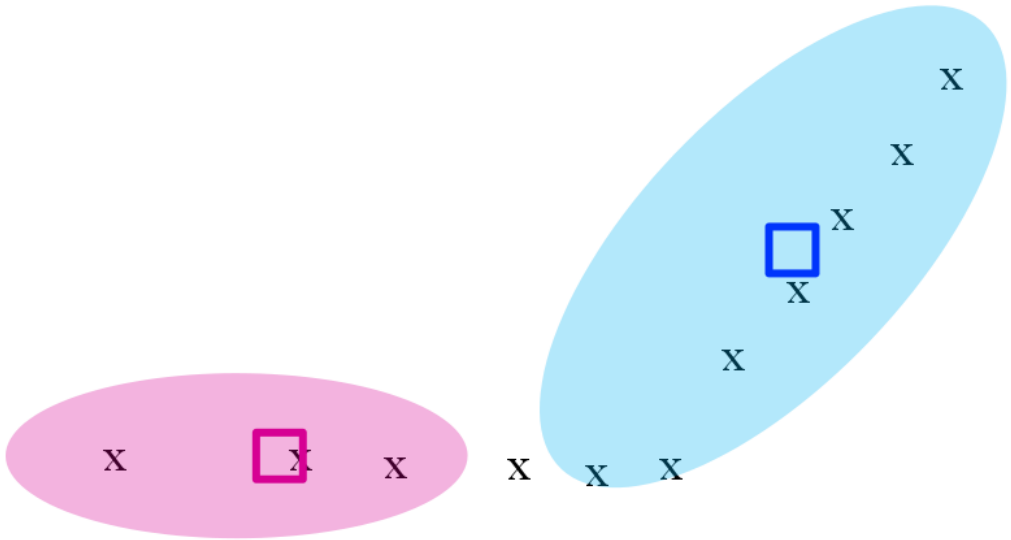
A Simple Example



x … data point
□ … centroid

Clusters after round 2

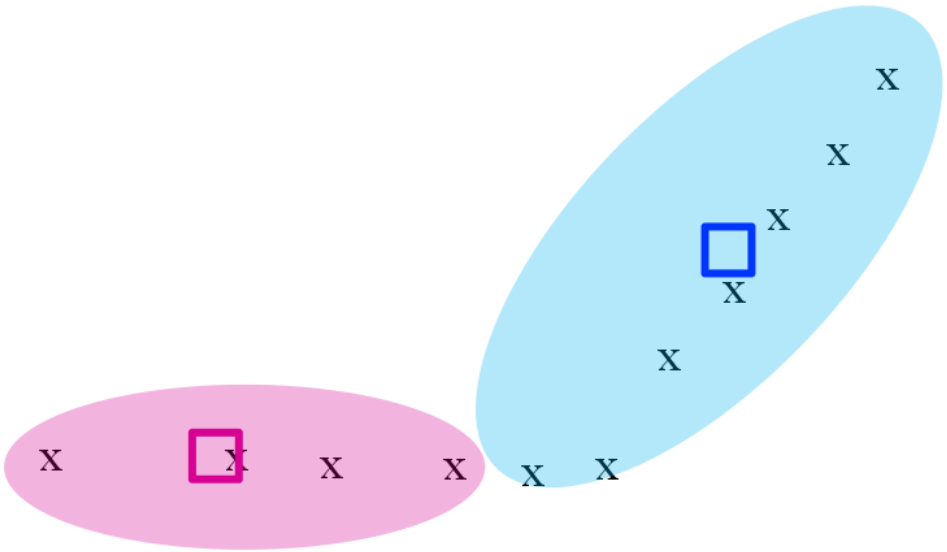
A Simple Example



x … data point
□ … centroid

Clusters after round 2

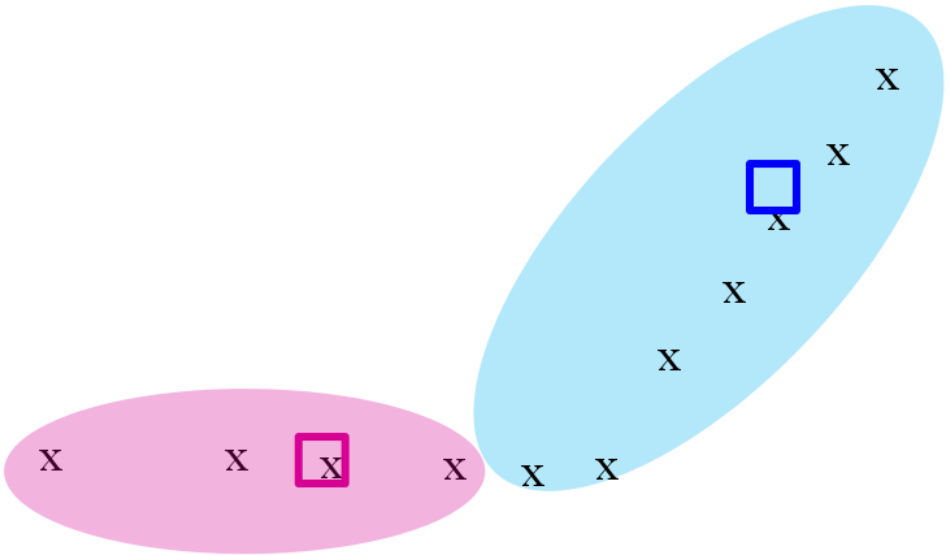
A Simple Example



x … data point
□ … centroid

Clusters after round 2

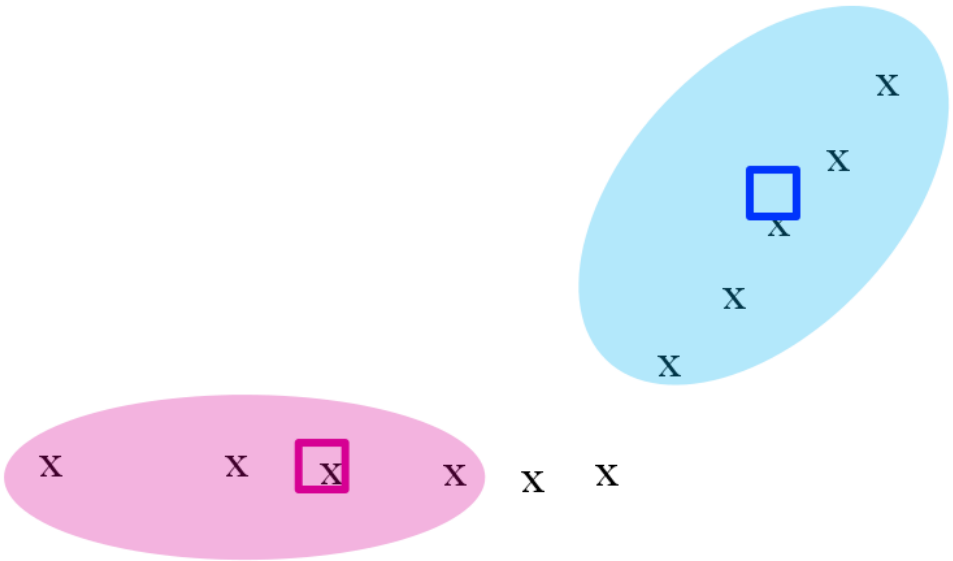
A Simple Example



x … data point
□ … centroid

Clusters at the end

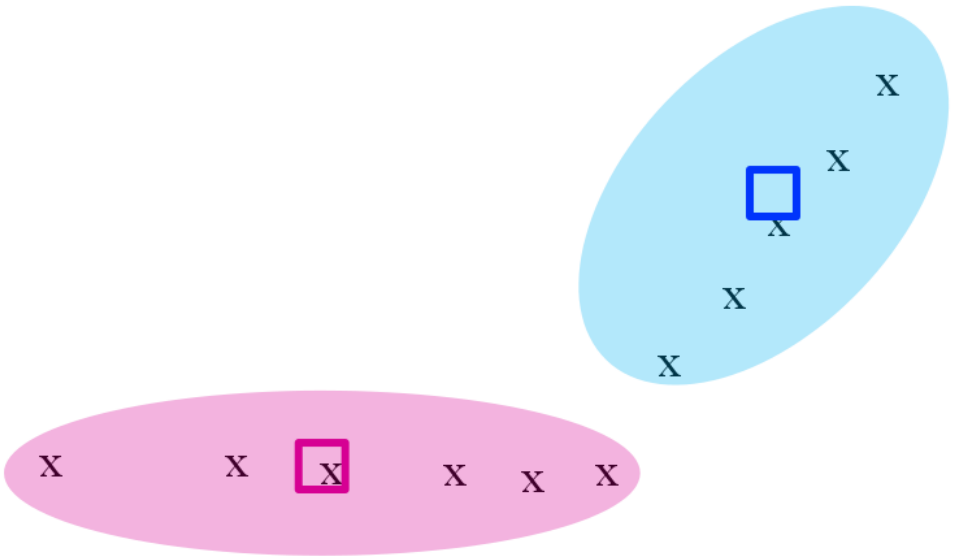
A Simple Example



x … data point
□ … centroid

Clusters at the end

A Simple Example

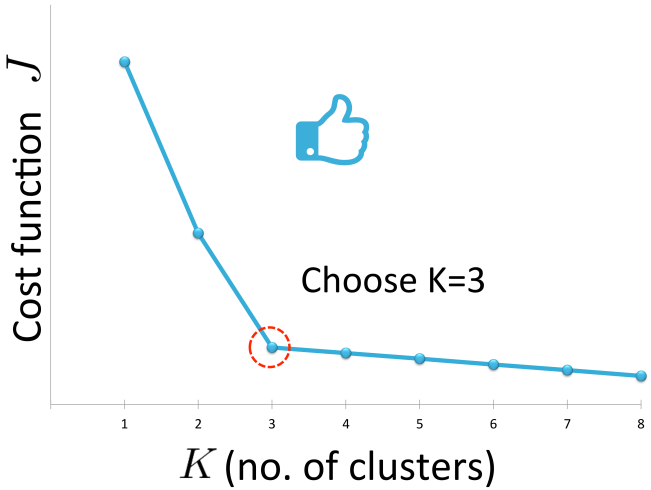


x … data point
□ … centroid

Clusters at the end

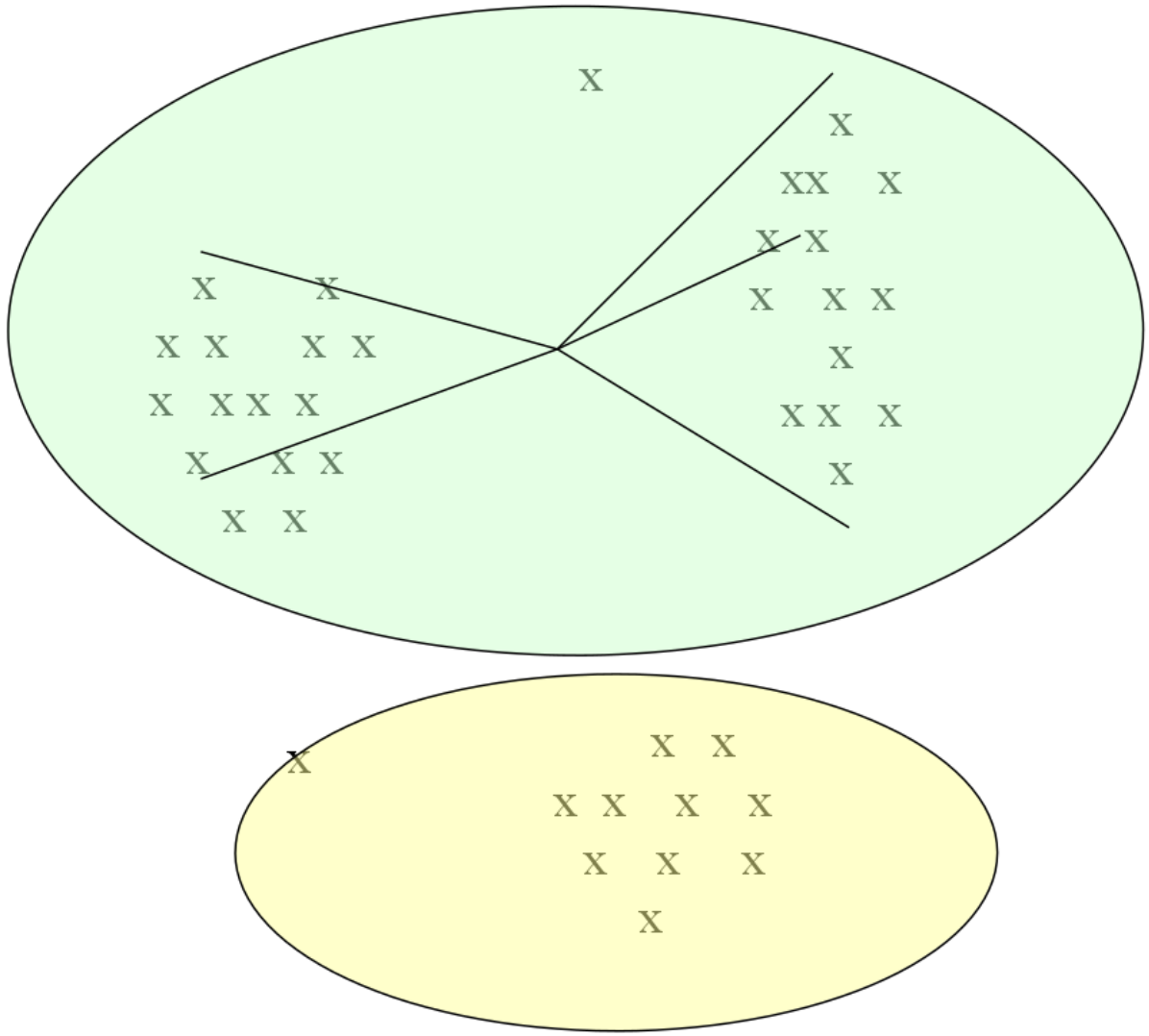
How to select k?

- We use the elbow method to determine the optimum number of clusters.
- Try different k, looking at the change in the average distance to centroid as k increases.
- Average falls rapidly until right k, then changes little.



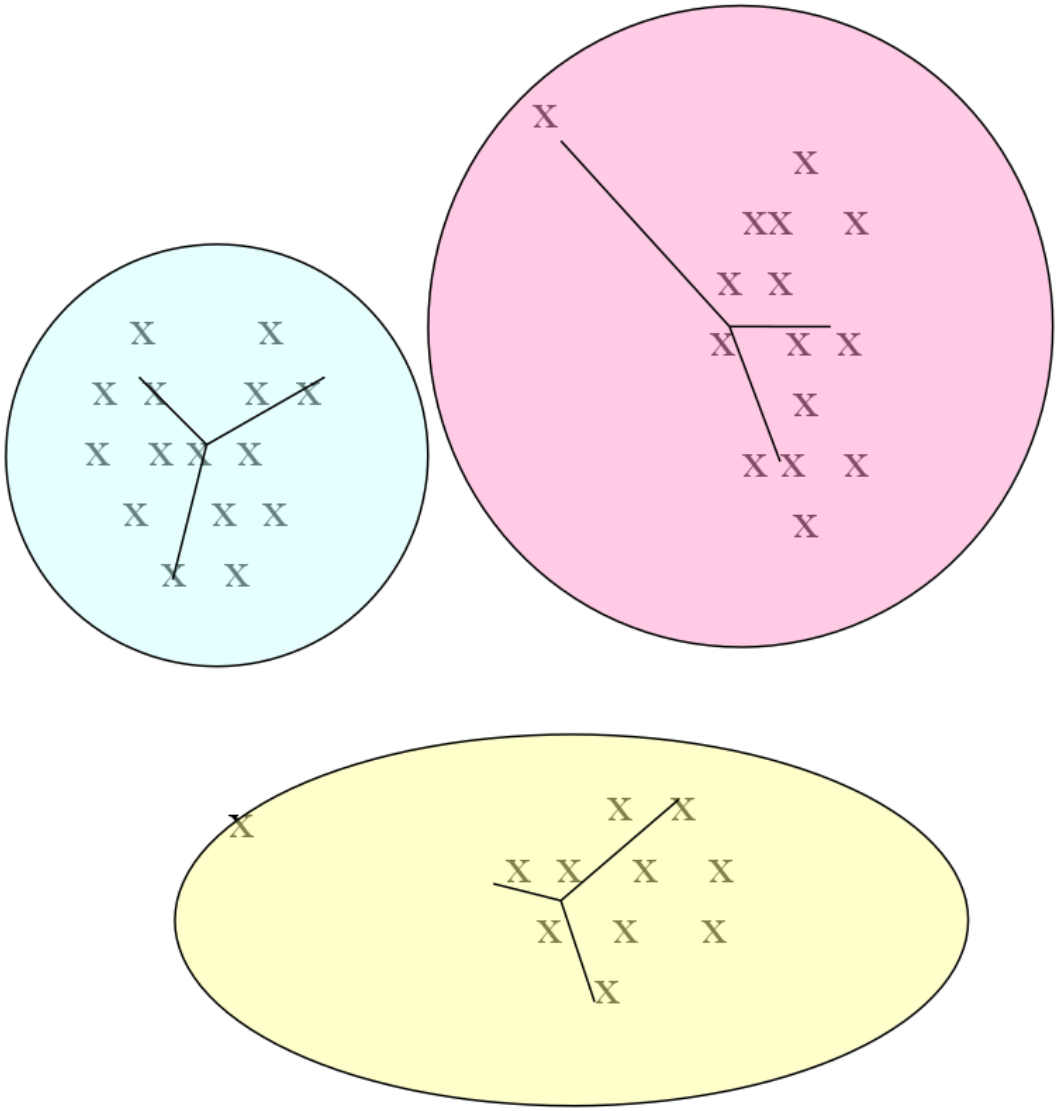
Selection of k – an example

Too few;
many long
distances
to centroid.



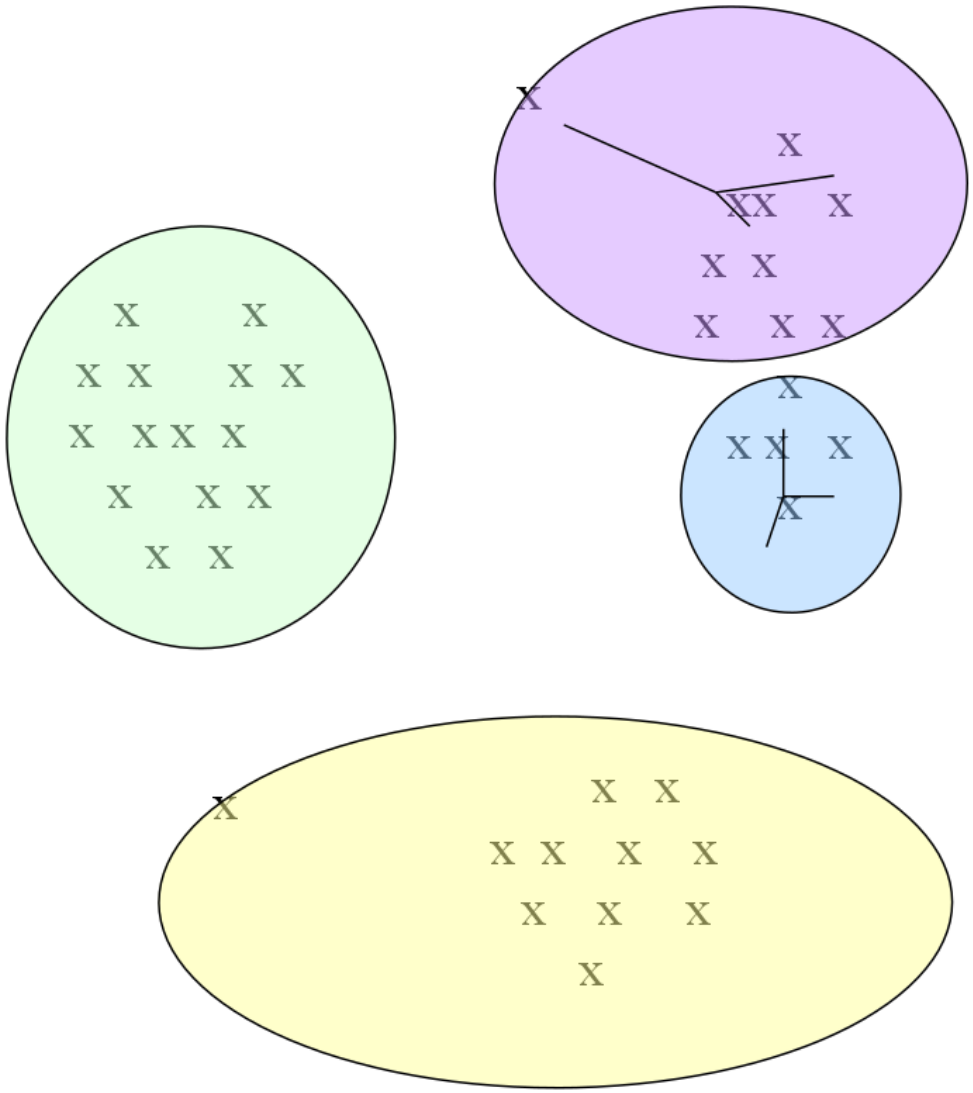
Selection of k – an example

Just right;
distances
rather short.



Selection of k – an example

Too many;
little improvement
in average
distance.



K-means in Python

Open this GDrive folder and let us see how to use K-means:
`https://drive.google.com/drive/folders/1kZDtH_7eo0K3ERXFerUt9J1fYZXf00kM?usp=sharing`