UNSUPERVISED LEARNING

Kristina Lerman
USC Information Sciences Institute

DSCI 552 – Spring 2021
February 8, 2021

Topics this week

- Schedule adjustments:
  - Quiz 3 due Wed 2/10
  - PS1 due Thu 2/11
  - Literature review – delayed til Friday 2/19 (Project Outline due Mar 5)
- Unsupervised learning
  - Clustering: K-means, hierarchical agglomerative clustering
  - Mixture models: GMM, EM
- Non-parametric methods
  - Density estimation: histograms, …
  - Embedding techniques: PCA, t-SNE
- Online office hours following the class (see link on BB or slides)

Final Project: some questions to guide you

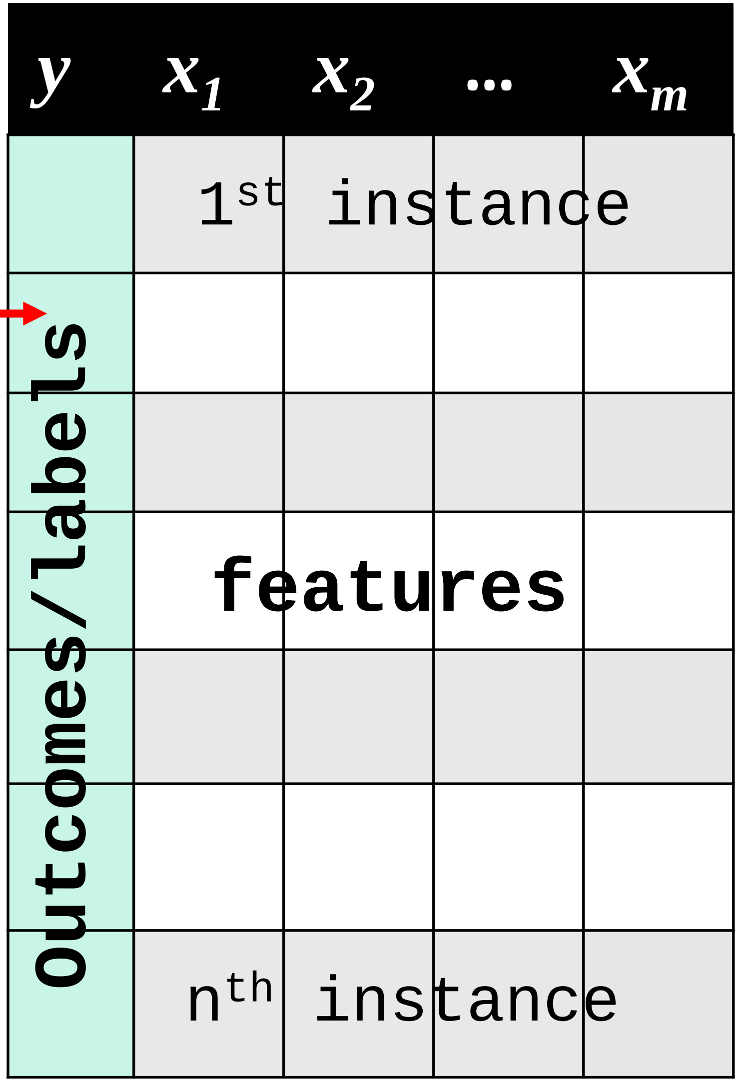
- What is the problem you are trying to solve?
  - I want to produce a better way to predict airline ticket prices
- Where is the data coming from?
  - Historic ticket prices…
- How have others solved the problem before? Did they approach work?
- How can you improve it?
  - If proposing a model based on other data sources, e.g., weather, where will they come from?
- How will you know that your method works? How will you evaluate it?
- What are the limitations of your approach?

WHY UNSUPERVISED LEARNING?

Why unsupervised learning?

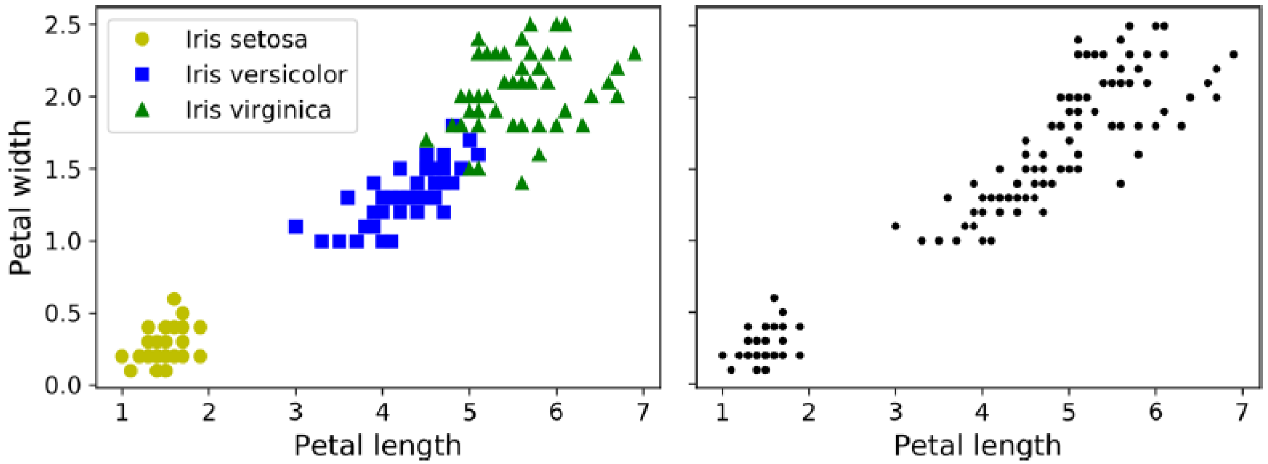
- Learning from unlabeled data
- Usually, no ground truth data, labels or outcomes are given

- "If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake." – Yann LeCun



What is unsupervised learning?

- **Supervised learning**: e.g, Classification
- **Unsupervised learning**:
  - Just as in classification, each data point is assigned to a group/label
  - But groups/labels are not known
  - But we got data! (sometimes a lot!)
- And what can we find in the data?



Classification vs Clustering

- Supervised: X = $\{\boldsymbol{x}^t, \boldsymbol{l}^t\}_t$
- Classes C_i i=1,...,K

$$p(\mathbf{x}) = \sum_{i=1}^{K} p(\mathbf{x} \mid C_i) P(C_i)$$

where $p(\boldsymbol{x}|C_i) \sim N(\boldsymbol{\mu}_i, \Sigma_i)$

- Model $\Phi = \{P(C_i), \boldsymbol{\mu}_i, \Sigma_i\}_{i=1}^{K}$

$$\hat{P}(C_i) = \frac{\sum_t l_i^t}{N} \quad \mathbf{m}_i = \frac{\sum_t l_i^t \mathbf{x}^t}{\sum_t l_i^t}$$

$$\mathrm{S}_i = \frac{\sum_t l_i^t (\mathbf{x}^t - \mathbf{m}_i)(\mathbf{x}^t - \mathbf{m}_i)^T}{\sum_t l_i^t}$$

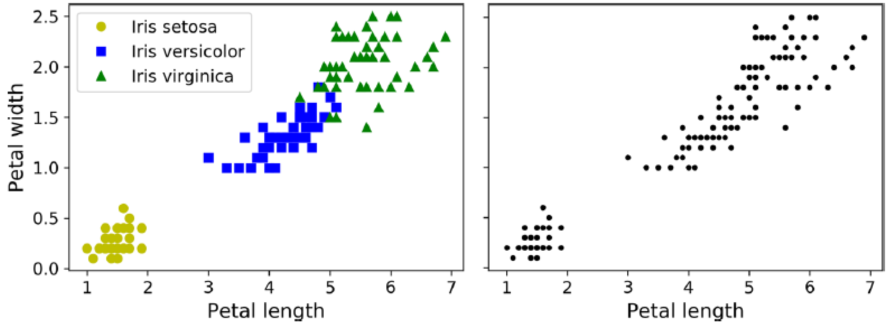
- Unsupervised : X = $\{\boldsymbol{x}^t\}_t$
- Clusters G_i i=1,...,k

$$p(\mathbf{x}) = \sum_{i=1}^{k} p(\mathbf{x} \mid G_i) P(G_i)$$

where $p(\boldsymbol{x}|G_i) \sim N(\boldsymbol{\mu}_i, \Sigma_i)$

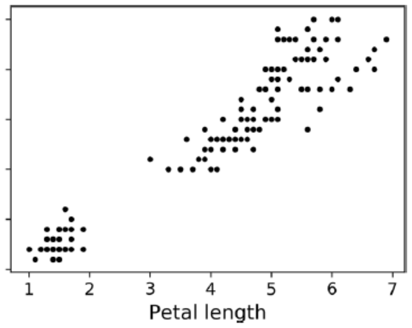
- Model $\Phi = \{P(G_i), \boldsymbol{\mu}_i, \Sigma_i\}_{i=1}^{k}$

Labels $\boldsymbol{l}^t_i$?



What is clustering?

- Want to find a ‘natural’ grouping between data instances
  – We want to find ‘similar’ instances and treat them in the same way
- No universal definition of what a good cluster is. Different algorithms capture different kinds of clusters.
  – Data points near a centroid
  – Dense regions
  – Well-separated regions

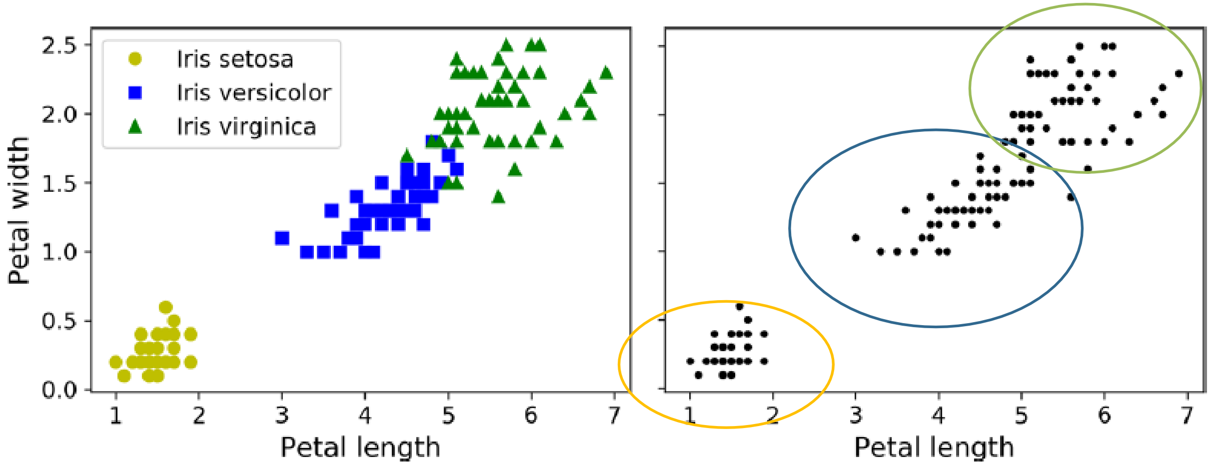


- Data instances within a cluster are closer to each other than to data points in different clusters
  – **High intra-cluster similarity**
  – **Low inter-cluster similarity**

What does it mean to be **similar**?

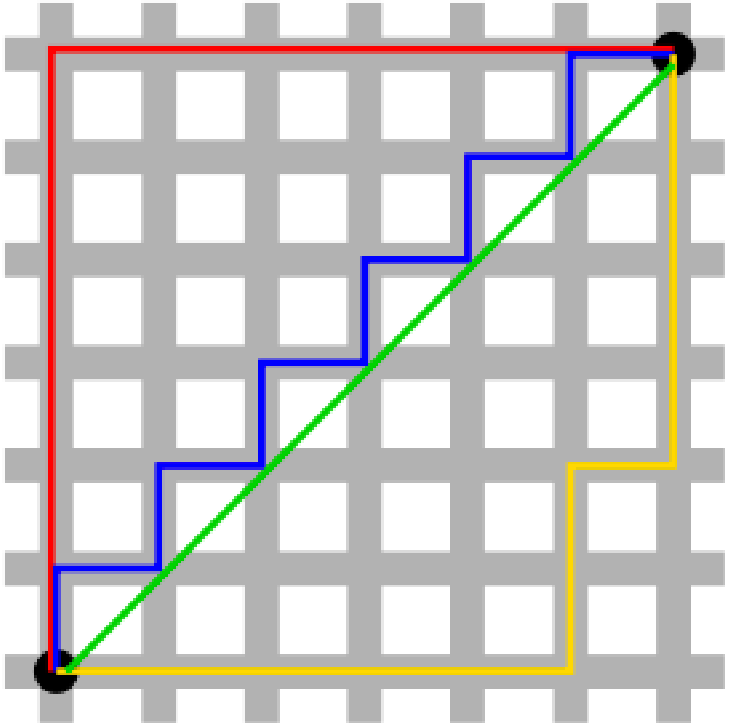
- How to define similarity is one of the most important questions in unsupervised learning.
- Similarity often measured using **distance measures** (e.g.
- Euclidean distance, or Mahalanobis Distance)

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



Numeric Distance Metrics

Names	Formula
Euclidean distance	$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
squared Euclidean distance	$\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$
Manhattan distance	$\|a-b\|_1 = \sum_i \lvert a_i - b_i \rvert$
maximum distance	$\|a-b\|_\infty = \max_i \lvert a_i - b_i \rvert$
Mahalanobis distance	$\sqrt{(a-b)^\top S^{-1}(a-b)}$ where S is the covariance matrix
cosine similarity	$\frac{a \cdot b}{\|a\|\|b\|}$



Symbolic Distance Metrics

- **Hamming distance** between two symbolic strings of equal length is the number of positions at which the corresponding systems are different.
  - Measures the minimum number of substitutions required to change one string into the other
- **Levenshtein (edit) distance** is a metric for measuring the amount of difference between two sequences.
  - Is defined as the minimum number of edits needed to transform one string into the other.

100**1001**
100**0100**
D=3

LD(BIOLOGY, BIOLGIA) = 2
BIOLOG**Y** -> BIOLOG**I** **(1 substitution)** BIOLOGI->BIOLOG**IA** **(1 insertion)**

http://www.astro.caltech.edu/~george/aybi199/Donalek_Classif.pdf

Distance axioms

- You can choose any distance measure as similarity metric
- Distance metrics satisfy the following properties: Given a distance measure d(.,.)

Self-similarity: $d(a,a) = d(b,b) \;\; \forall a, b \in X$

Minimality: $d(a,a) < d(a,b), \; \forall a, b \in X, a \neq b$

Symmetry: $d(a,b) = d(b,a), \;\; \forall a, b \in X$

Triangle inequality: $d(a,b) + d(b,c) >= d(a,c), \;\; \forall a, b, c \in X,$

DATA STANDARDIZATION

Data Standardization

- In the Euclidean space, standardization of features is recommended so that all attributes can have equal impact on the computation of distances.
- Consider the following pair of data points
  - $\mathbf{x}_i$: (0.1, 20) and $\mathbf{x}_j$: (0.9, 720).

$$dist(\mathbf{x}i, \mathbf{x}j) = \sqrt{(0.9-0.1)^2 + (720-20)^2} = 700.00,$$

- The distance is almost completely dominated by (720-20) = 700.
- Standardize attributes: to force the features to have a common value range

Interval-scaled attributes

- Their values are real numbers following a linear scale
  - The difference in Age between 10 and 20 is the same as that between 40 and 50.
  - The key idea is that intervals keep the same importance through out the scale
- Two main approaches to standardize interval scaled attributes, **range** and **z-score**. *f* is an attribute

$$range(x_{if}) = \frac{x_{if} - \min(f)}{\max(f) - \min(f)},$$

//www.cs.uic.edu/~liub/

Interval-scaled attributes

- Z-score: transforms the attribute values so that they have a mean of zero and a **mean absolute deviation** of 1. The mean absolute deviation of attribute f, denoted by s_f, is computed as follows

$$s_f = \frac{1}{n}\left(|x_{1f} - m_f| + |x_{2f} - m_f| + \ldots + |x_{nf} - m_f|\right),$$

$$m_f = \frac{1}{n}\left(x_{1f} + x_{2f} + \ldots + x_{nf}\right),$$

Z-score: $$z(x_{if}) = \frac{x_{if} - m_f}{s_f}.$$

//www.cs.uic.edu/~liub/

Ratio-scaled attributes

- Numeric attributes, but unlike interval-scaled features, their scales are exponential,
- For example, the total amount of microorganisms that evolve in a time t is approximately given by

  Ae^{Bt},

  where A and B are some positive constants.
- Do log transform:
  - Then treat it as an interval-scaled feature

$$\log(x_{if})$$

Nominal features

- Sometimes, we need to transform nominal attributes to numeric attributes.
  - Transform nominal attributes to binary attributes.
  - The number of values of a nominal attribute is *v*.
  - Create *v* binary attributes to represent them.
  - If a data instance for the nominal attribute takes a particular value, the value of its binary attribute is set to 1, otherwise it is set to 0.
- The resulting binary attributes can be used as numeric attributes, with two values, 0 and 1.

//www.cs.uic.edu/~liub/

Nominal attributes: one hot encoding

- Nominal attribute *food*: has three values,
  - Apple, Chicken, and Broccoli
- We create three binary attributes in the new data: Apple, Chicken, and Broccoli
- If a particular data instance in the original data has Apple as the value for *food*,
  - then in the transformed data, we set the value of the attribute Apple to 1, and the values of attributes Chicken and Broccoli to 0

Label Encoding

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

→

One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
0	0	1	50

Ordinal attributes

- Ordinal attribute: an ordinal attribute is like a nominal attribute, but its values have a numerical ordering.
  - Age attribute with values: Young, MiddleAge and Old. They are ordered.
  - Common approach to standardization: treat is as an interval-scaled attribute.

HIERARCHICAL CLUSTERING

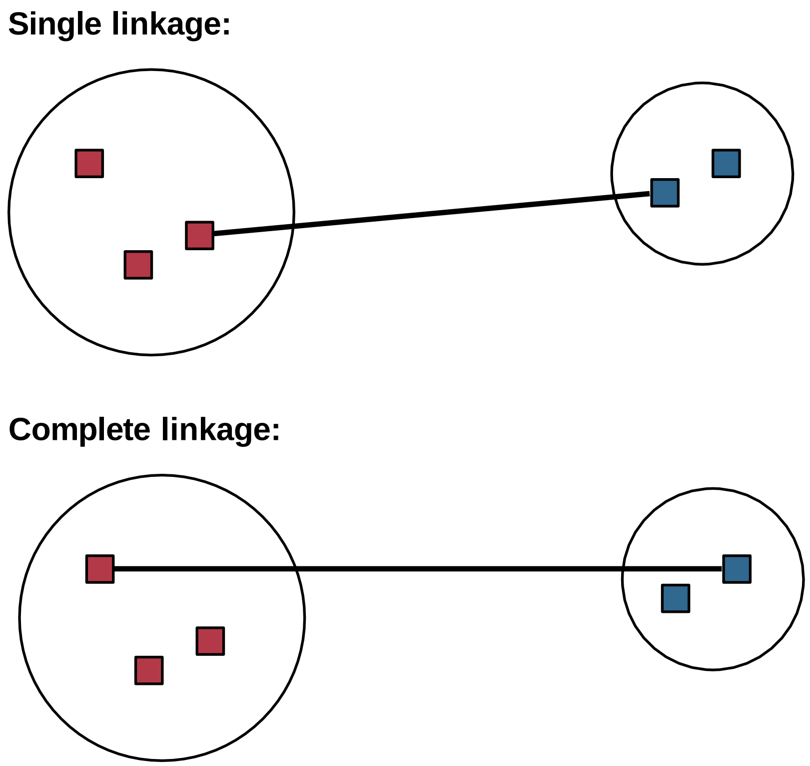
Hierarchical Clustering

- Two basic types of hierarchical clustering
  - Agglomerative clustering (bottom to top)
  - Divisive clustering (top to bottom)
- Based on a pre-defined distance measure and linkage criterion we split (divisive) and merge (agglomerative) clusters depending on the type
- Based on greedy search, therefore very slow and not scalable

- **Distance measures and linkage criteria have a big influence on the outcome of clusters!**

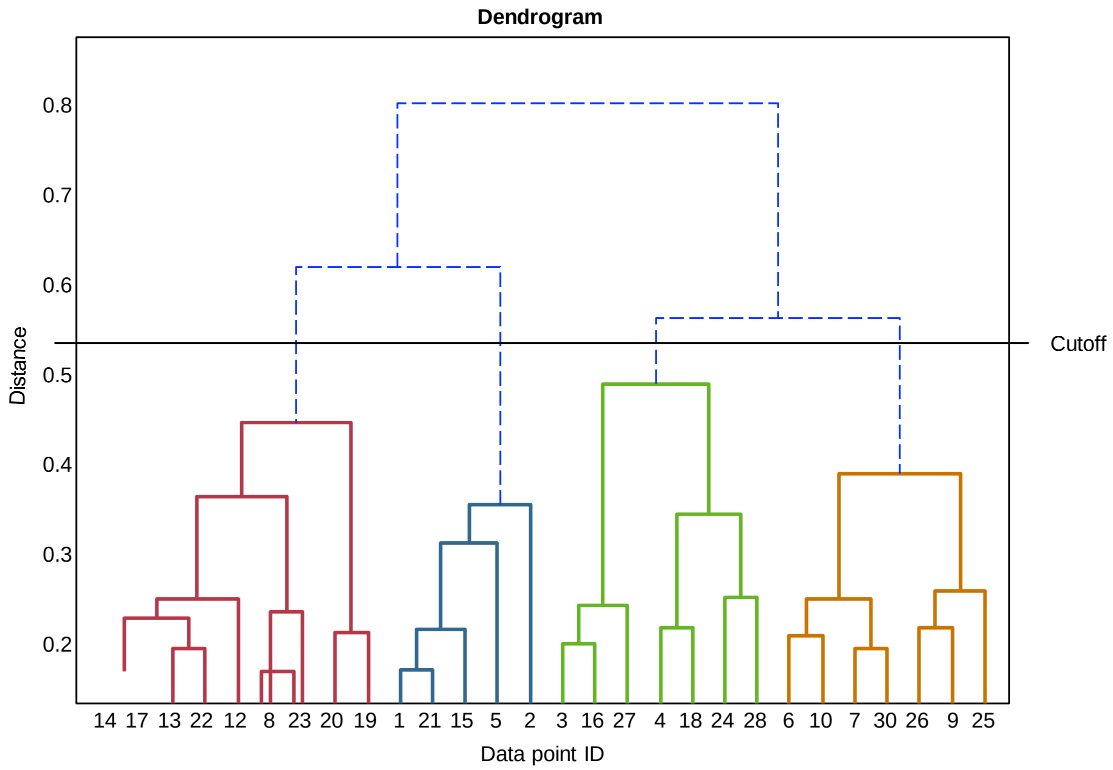
Hierarchical Clustering (2)

- Linkage types:
  - Single linkage (single minimal distance)
  - Complete linkage (maximal minimal distance)
  - Average linkage (minimal average distance)
- Different problems can occur. (e.g. chaining effects)



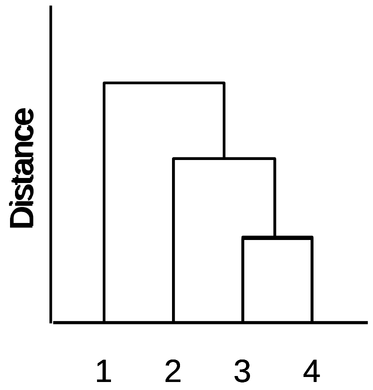
Dendrogram

- Tree diagrams
- Dendrograms are used to visualize distances and clusters
- Useful for analyzing hierarchical clusters with different cut-offs



Agglomerative Clustering

- Algorithm:
    1. Start: each data point belongs to a separate cluster
    2. Step: merge closest clusters based on linkage and distance
    3. End: all data points belong to the same cluster

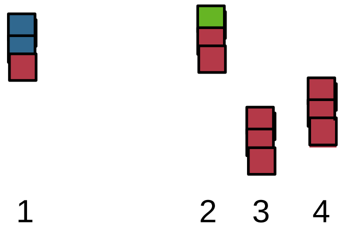


Agglomerative clustering: Step 1

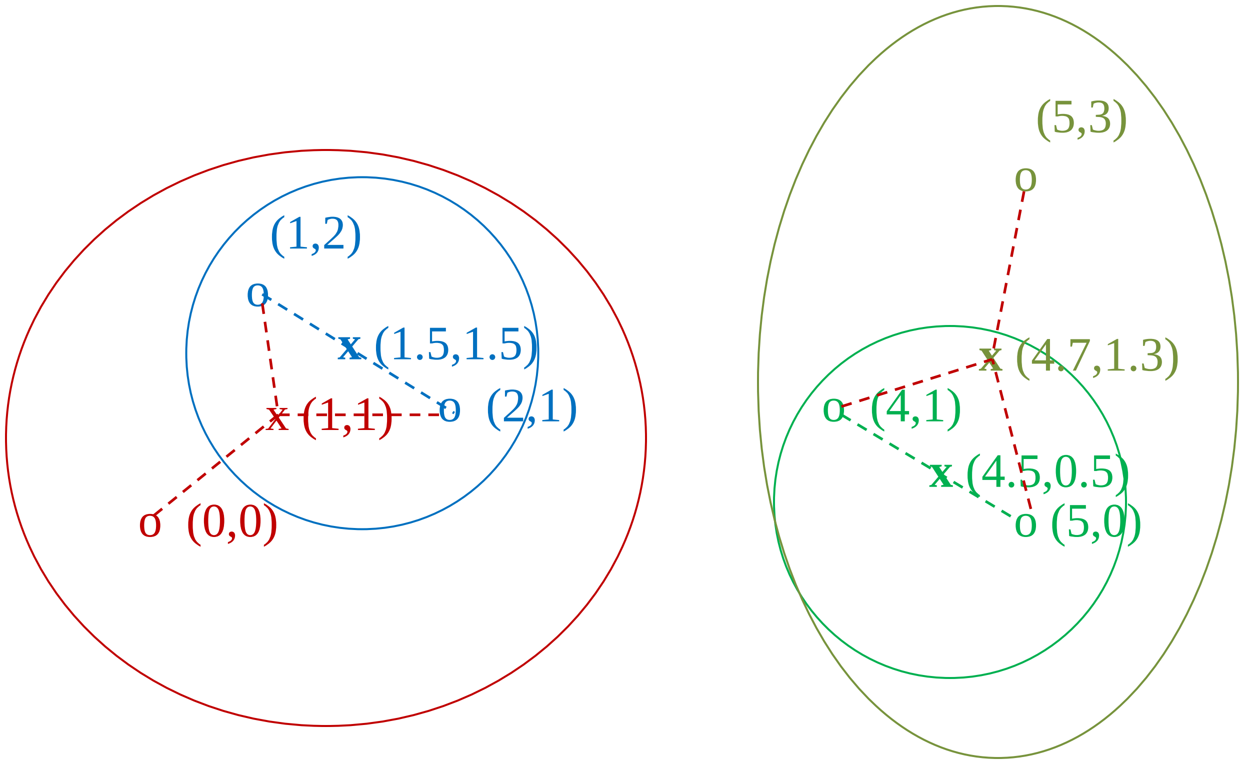
Agglomerative clustering: Step 2

Agglomerative clustering: Step 3

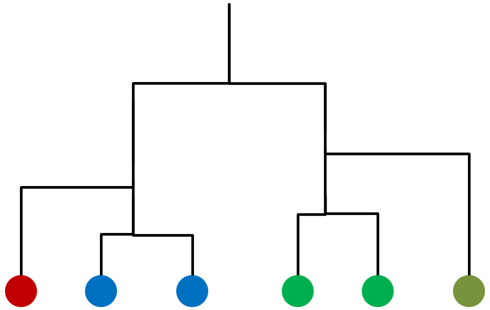
Agglomerative clustering: Step 4



Example: Hierarchical clustering



Data:
o … data point
x … centroid



Dendrogram

COMPETITIVE CLUSTERING

Competitive Clustering

- Base strategy of defining clusters with different variants
  - K-means clustering
  - Self-organizing maps
  - Affinity propagation, …
- Again chosen distance measure is the key
- Based on the "winner takes it all" principle
- After enough training steps it is assumed that winners (i.e. cluster centroids) are good representations of data within their cluster.

$$j^* = \arg\min_j ||x - c_j||$$

k-Means Clustering

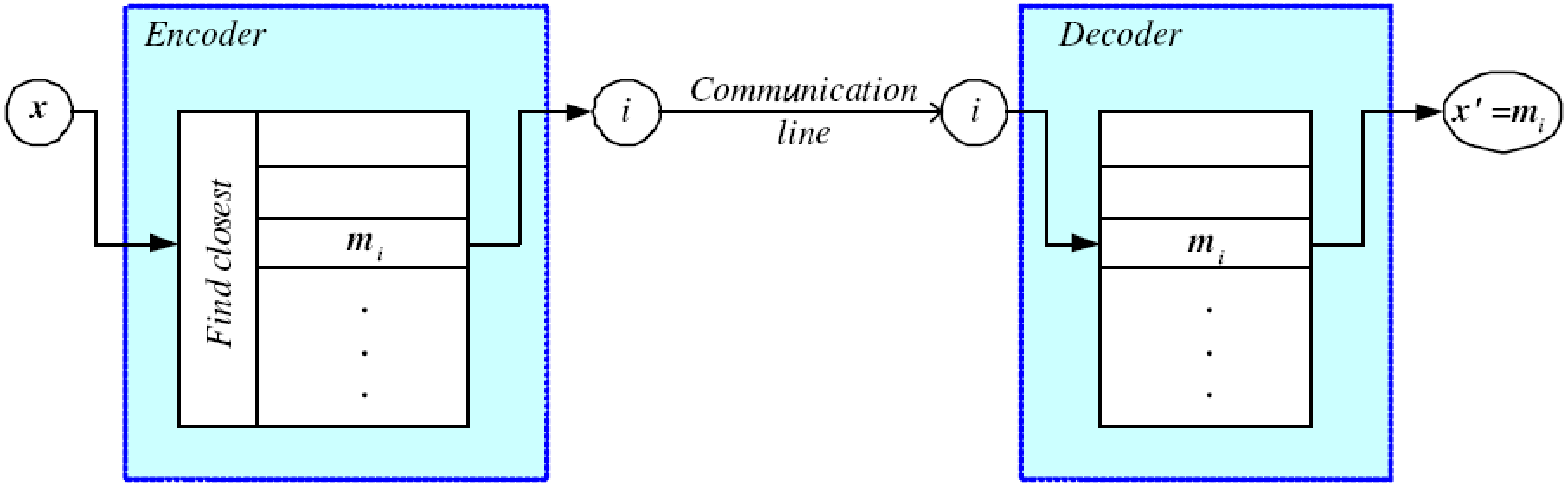
- Find k reference vectors (prototypes/codewords) which best represent data
- Reference vectors, $\boldsymbol{m}_j$, j =1,...,k
- Use nearest (most similar) reference:

$$\left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_j \left\|\mathbf{x}^t - \mathbf{m}_j\right\|$$

- Reconstruction error $E\left(\{\mathbf{m}_i\}_{i=1}^k \middle| \mathcal{X}\right) = \sum_t \sum_i b_i^t \left\|\mathbf{x}^t - \mathbf{m}_i\right\|$

$$b_i^t = \begin{cases} 1 & \text{if } \left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_j \left\|\mathbf{x}^t - \mathbf{m}_j\right\| \\ 0 & \text{otherwise} \end{cases}$$

Encoding/Decoding



k-means Clustering

Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to k random $\boldsymbol{x}^t$

Repeat

Assign to centroid

For all $\boldsymbol{x}^t \in \mathcal{X}$

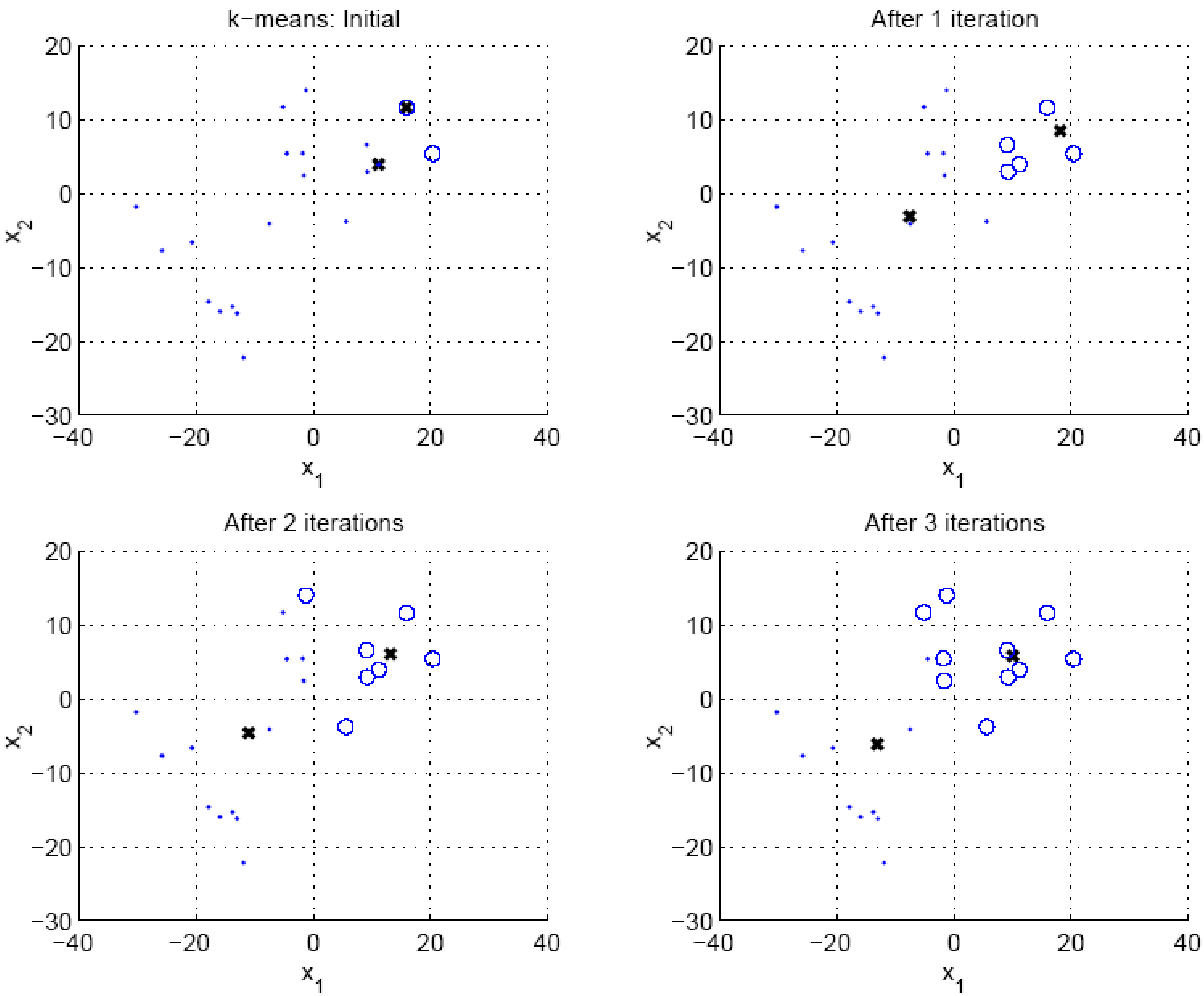
$$b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

Update centroid

For all $\boldsymbol{m}_i, i = 1, \ldots, k$

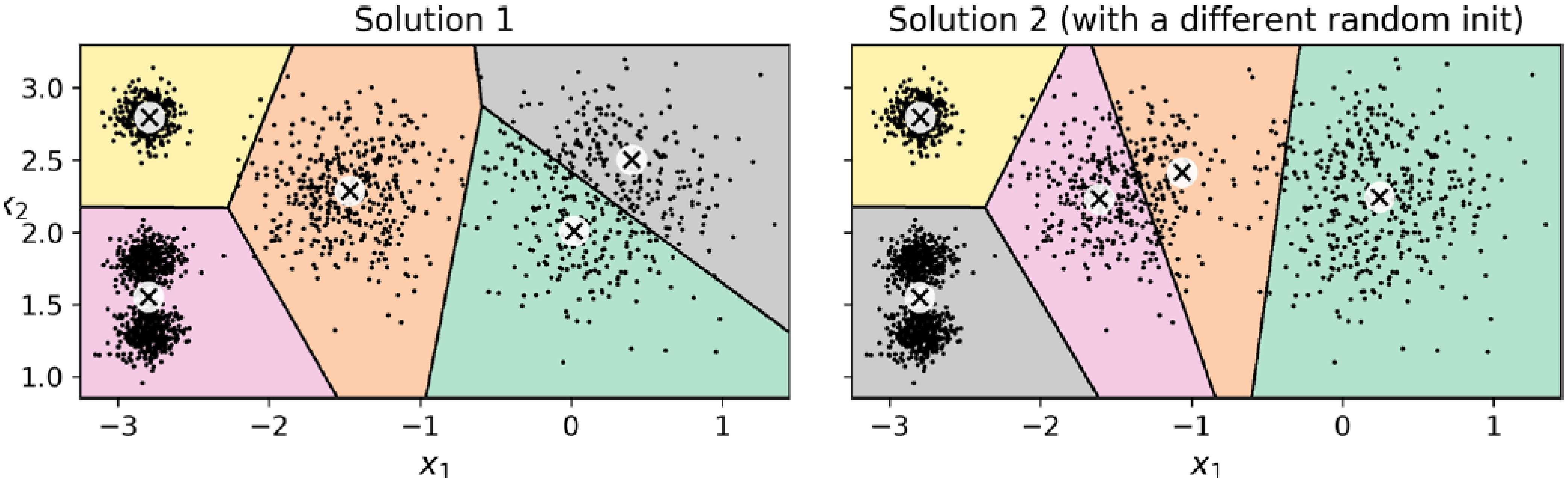
$$\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$$

Until $\boldsymbol{m}_i$ converge



k−means: Initial
After 1 iteration
After 2 iterations
After 3 iterations
x_1
x_2

Watch out for local minima!



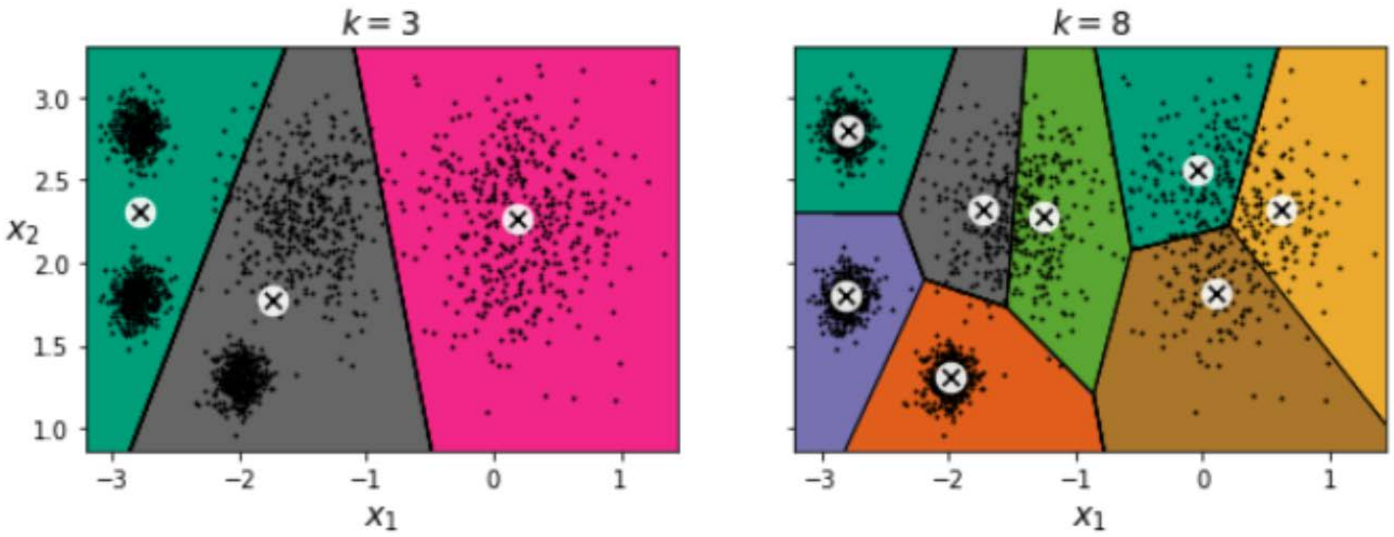
Convergence

- Several possibilities, e.g.,
  - A fixed number of iterations.
  - Data partition is unchanged.
  - Centroid positions don’t change → convergence
- Does k-means always converge? i.e., reach a state in which clusters don’t change.
  - Yes. It does, as it minimizes the overall distortion towards a local optimum.
  - How to find a better minimum?
    - Solution: e.g. run it multiple times
- K-means is a special case of a general procedure known as the Expectation Maximization (EM) algorithm.

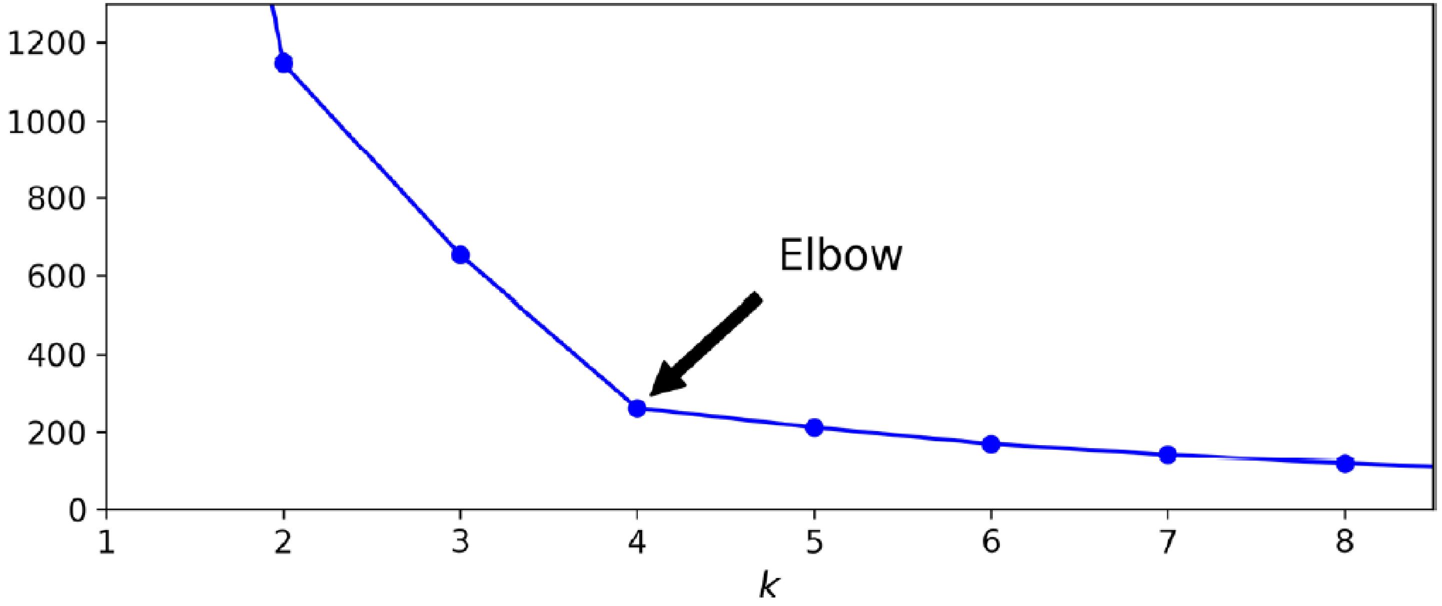
Computational complexity

- The computational complexity of the algorithm is generally linear with the number of instances N, the number of clusters K, and the number of dimensions m.
- However, this is only true when the data has a clustering structure. If it does not, then the complexity can increase exponentially with the number of instances.
- This is rare, and K-Means is generally one of the fastest clustering algorithms.

How many clusters?



Mean squared distance

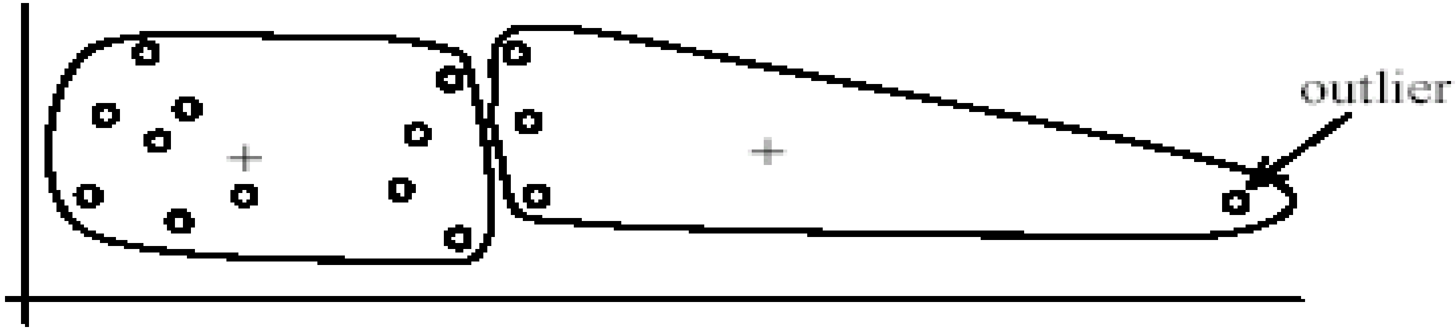


Weaknesses of K-Means

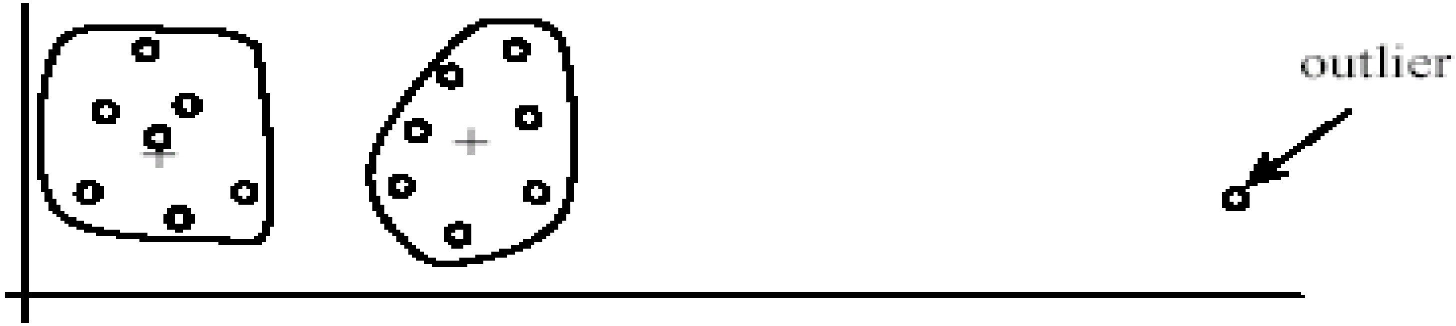
- The user needs to specify *k*
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are very far away from other data points
  - Outliers could be errors in the data recording or some special data points with very different values

//www.cs.uic.edu/~liub/

Weaknesses of k-means: Problems with outliers



(A): Undesirable clusters



(B): Ideal clusters

//www.cs.uic.edu/~liub/

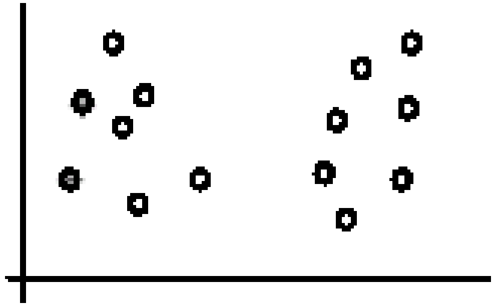
To deal with outliers

- One method is to remove some data points in the clustering process that are much further away from the centroids than other data points
  - To be safe, we may want to monitor these possible outliers over a few iterations and then decide to remove them.
- Another method is to perform random sampling
  - Since in sampling we only choose a small subset of the data points, the chance of selecting an outlier is very small
  - Assign the rest of the data points to the clusters by distance or similarity comparison, or classification

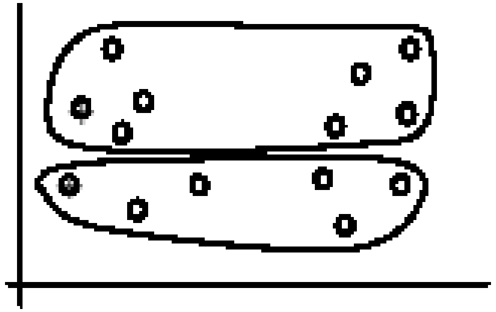
//www.cs.uic.edu/~liub/

Weaknesses of k-means

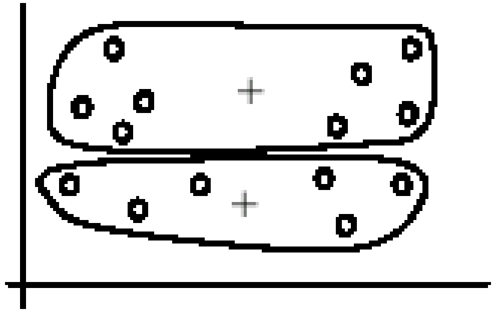
- The algorithm is sensitive to initial seeds



(A). Random selection of seeds (centroids)



(B). Iteration 1

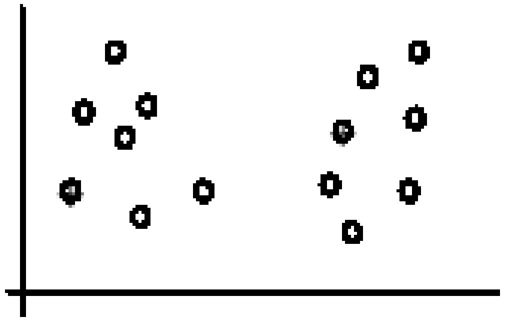


(C). Iteration 2

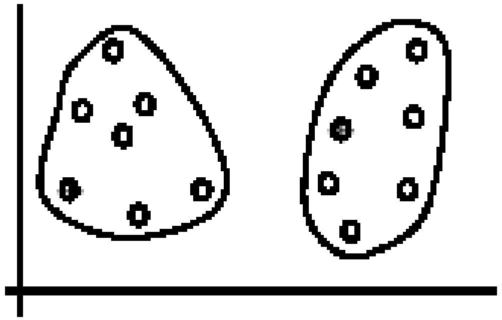
Weaknesses of k-means (cont …)

- If we use different seeds: good results

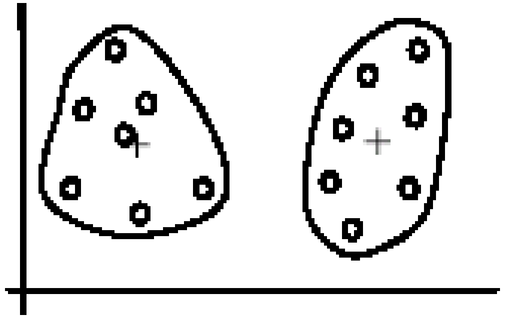
There are some methods to help choose good seeds



(A). Random selection of k seeds (centroids)



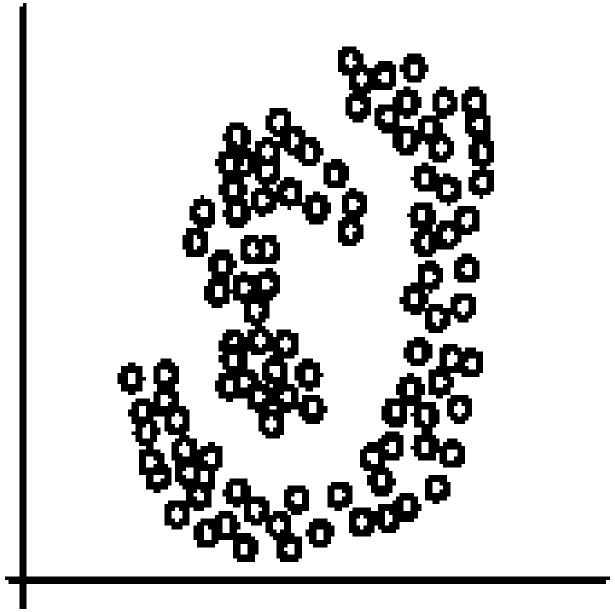
(B). Iteration 1



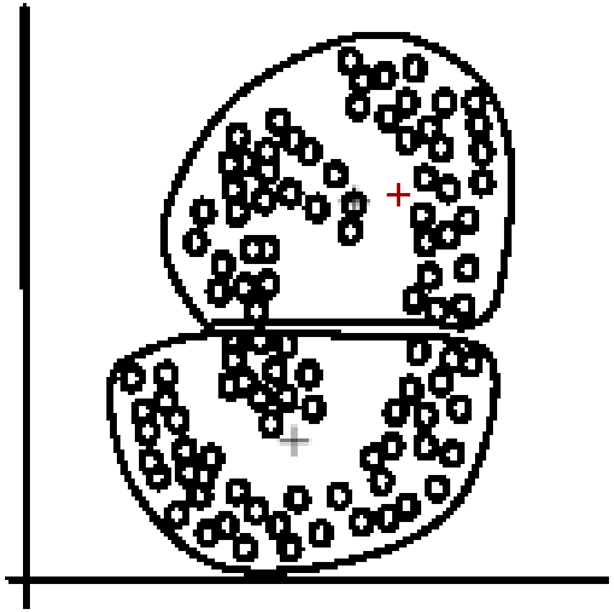
(C). Iteration 2

Weaknesses of k-means (cont …)

The *k*-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres)



(A): Two natural clusters



(B): *k*-means clusters

//www.cs.uic.edu/~liub/

Cluster Evaluation: a hard problem

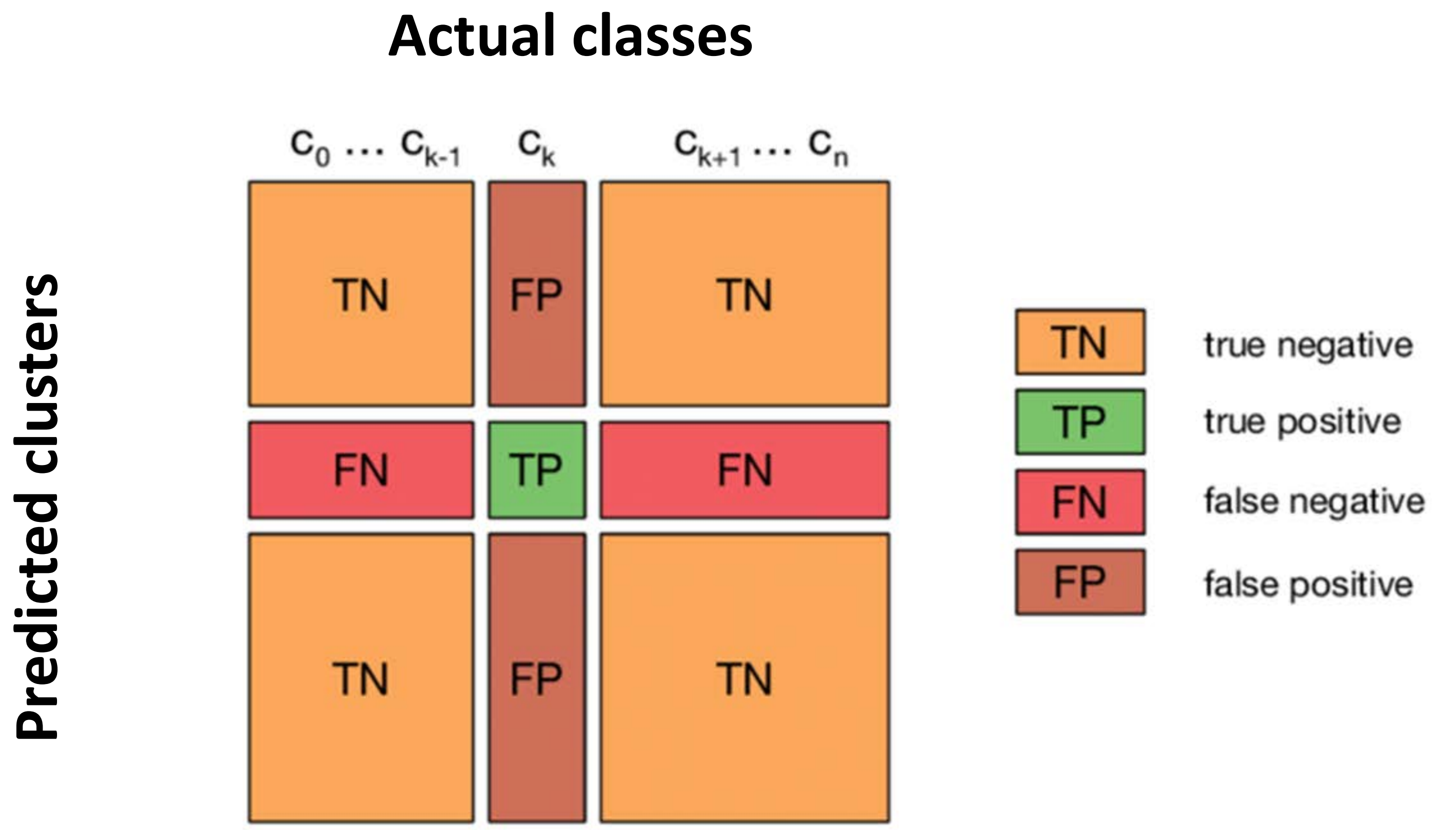
- The quality of a clustering is very hard to evaluate because we do not know the correct clusters
- Some methods are used:
  - User inspection
  - Study centroids, and spreads
  - Entropy, Purity
  - Silhouette score

//www.cs.uic.edu/~liub/

Cluster evaluation: ground truth

- We use some labeled data (for classification)
- Assumption: Each class is a cluster.
- After clustering, a confusion matrix is constructed. From the matrix, we compute various measurements, entropy, purity, precision, recall and F-score.
  - Let the classes in the data D be $C = (c_1, c_2, \ldots, c_k)$. The clustering method produces k clusters, which divides D into k disjoint subsets, $D_1, D_2, \ldots, D_k$.

Multi-class Confusion Matrix



Evaluation measures: Entropy

Entropy: For each cluster, we can measure its entropy as follows:

$$entropy(D_i) = -\sum_{j=1}^{k} \Pr_i(c_j)\log_2 \Pr_i(c_j), \qquad (29)$$

where $\Pr_i(c_j)$ is the proportion of class c_j data points in cluster i or D_i. The total entropy of the whole clustering (which considers all clusters) is

$$entropy_{total}(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times entropy(D_i) \qquad (30)$$

//www.cs.uic.edu/~liub/

Evaluation measures: purity

Purity: This again measures the extent that a cluster contains only one class of data. The purity of each cluster is computed with

$$purity(D_i) = \max_j(\Pr_i(c_j)) \tag{31}$$

The total purity of the whole clustering (considering all clusters) is

$$purity_{total}(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times purity(D_i) \tag{32}$$

//www.cs.uic.edu/~liub/

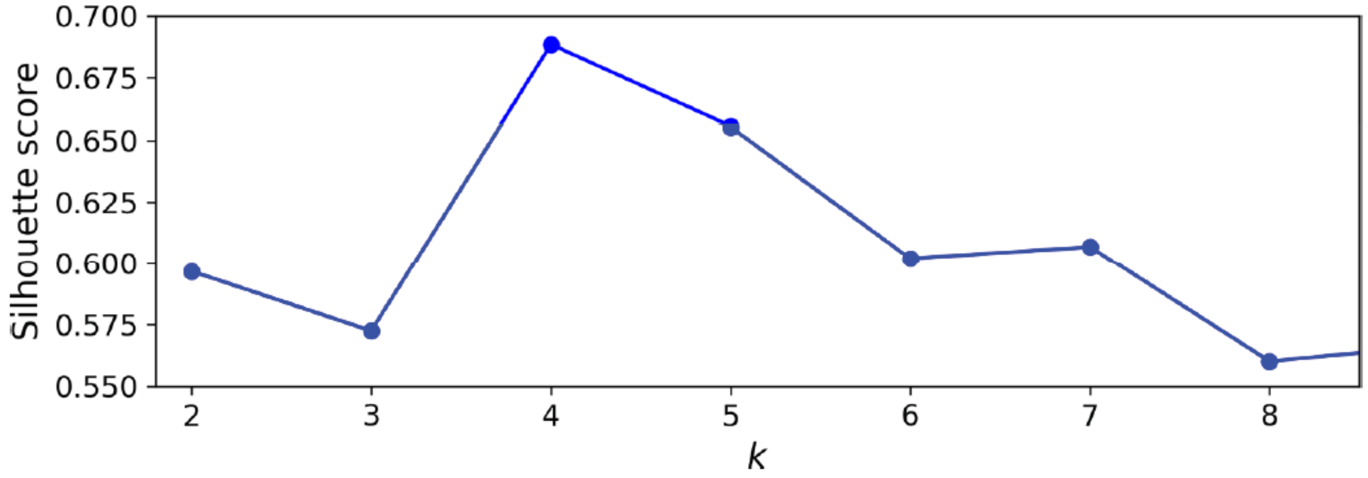
Evaluation based on internal information

- **Intra-cluster cohesion** (compactness):
  - Cohesion measures how similar to each other the data points in the same cluster are
  - i.e., how near they are to the cluster centroid.
  - Sum of squared error (SSE) is a commonly used measure.
- **Inter-cluster separation** (isolation):
  - Separation means that different cluster centroids should be far away from one another.

//www.cs.uic.edu/~liub/

Silhouette score

- Silhouette score measures how similar an instance is to its own cluster (cohesion) compared to other clusters (separation) and ranges from −1 to +1.
- If most instances have a high value, then the clustering configuration is appropriate. If many points have negative values, then there are too many or too few clusters.

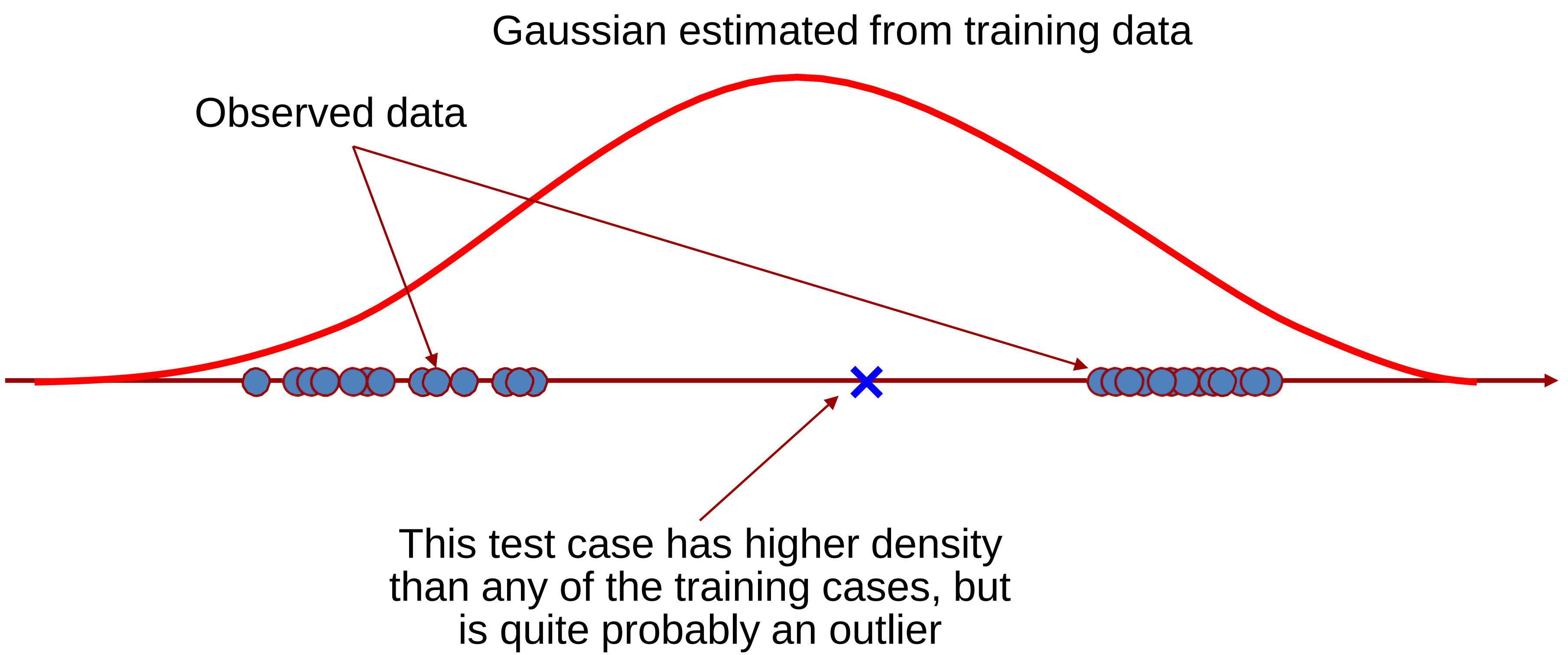


Other clustering methods

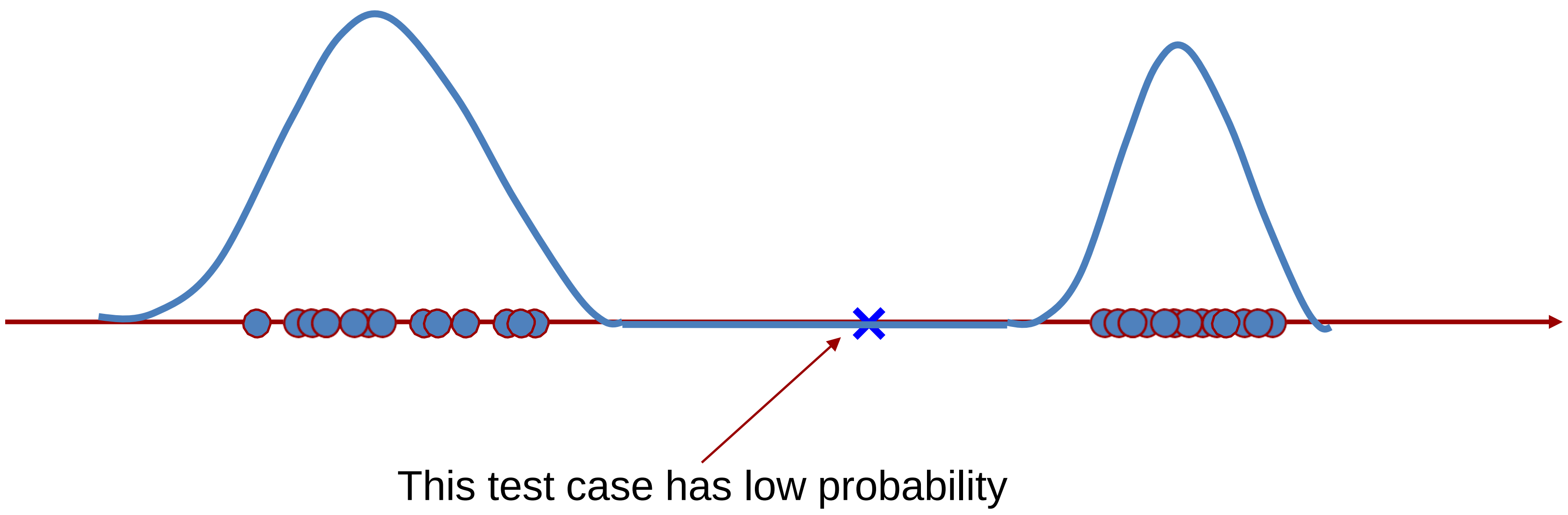
- DBSCAN
  - Clusters as continuous regions of high density. Good for spatial clustering
  - Can identify any number of clusters of any shape (with 2 hyperparameters)
  - Robust to outliers
- Affinity propagation
  - Every data point votes for another data point as its representative
  - After convergence, each representative and its voters form a cluster
  - Computationally expensive
- Spectral clustering
  - Creates a low-dimensional embedding from the similarity matrix
  - Can capture complex cluster structures
  - Efficient approximate methods

GAUSSIAN MIXTURE MODELS

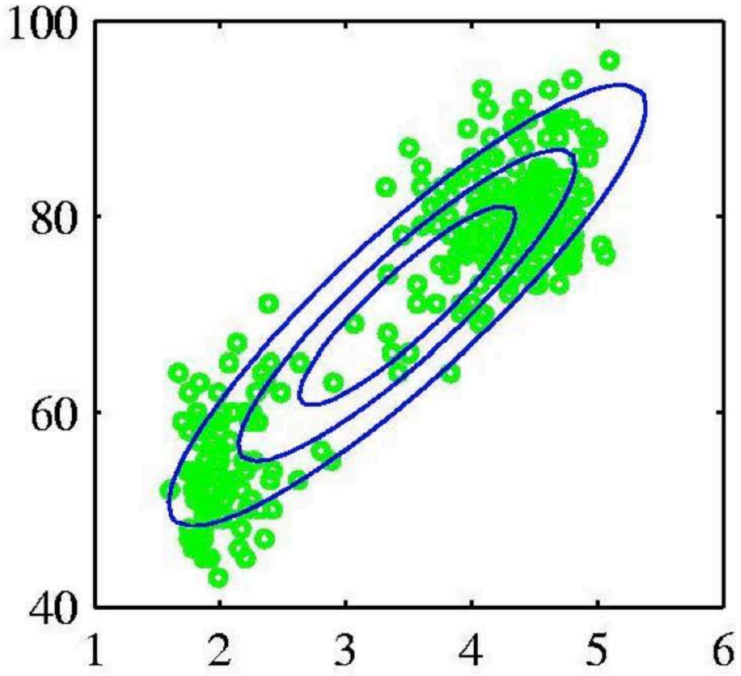
Illustration



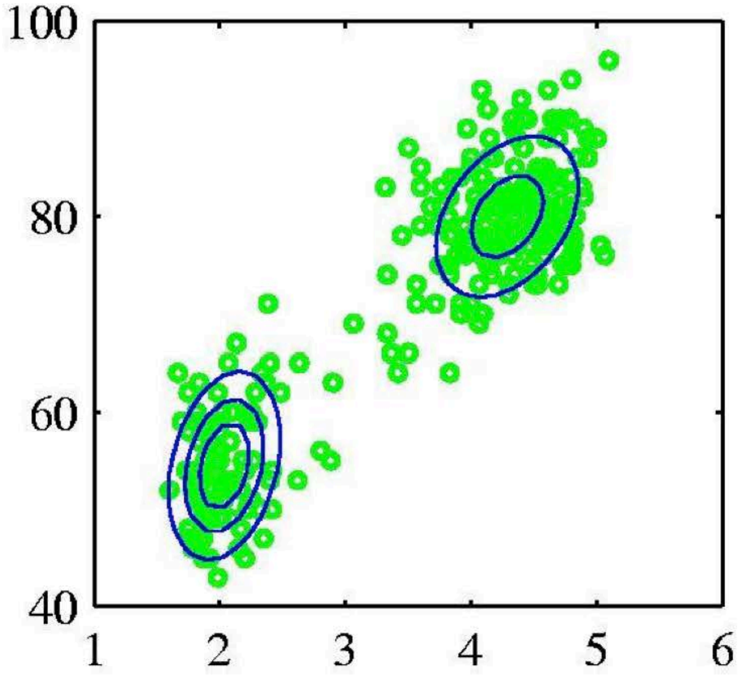
Illustration



Mixture of Gaussians



Single Gaussian



Mixture of two Gaussians

- Any data can be represented by a mixture of Gaussian clusters
- Each cluster can have a different ellipsoidal shape, size, density, and orientation

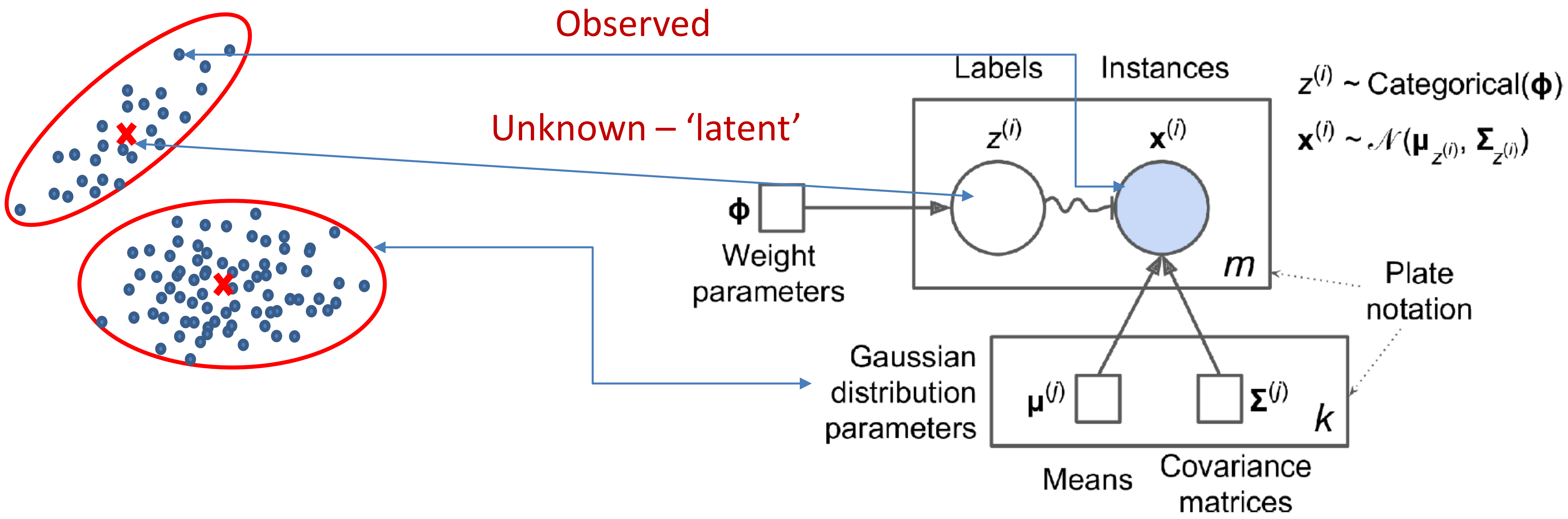
Mixture Densities

- Mixture Density: $p(\boldsymbol{x}) = \sum_{i=1}^{k} p(\boldsymbol{x}|\mathcal{G}_i)P(\mathcal{G}_i)$

- Component Density: $p(\boldsymbol{x}|\mathcal{G}_i)$
- Mixture Proportion: $P(\mathcal{G}_i)$

- **Learning:** estimating *component densities* and *proportions*.
- k is a hyper parameter.

What are we learning?



Gaussian mixture models

- **Expectation-Maximization algorithm** (EM algorithm): Iterative algorithm that finds the model parameters for which the observed data is most likely
  - Parameters: mean and covariance
    - **E-step:** Estimate labels/responsibilities for data given the parameters.
    - **M-step:** Update parameters given the estimated labels of E-step.
  - Repeat until convergence.
- K-means is a special case of EM algorithm
- **Issues:** Can end up in local minima, number of mixtures again is to be chosen and validated

Expectation Maximization Algorithm

- **The steps:**

$$\begin{aligned} \text{E-step} &: \quad \mathcal{Q}(\Phi|\Phi^l) = E[\mathcal{L}_c(\Phi|\mathcal{X},\mathcal{Z})|\mathcal{X},\Phi^l] \\ \text{M-step} &: \quad \Phi^{l+1} = \arg\max_{\Phi} \mathcal{Q}(\Phi|\Phi^l) \end{aligned}$$

- L_c is the *complete* likelihood dependent on observable X and unobservable random variables Z
- Φ are the model parameters.
- Some definitions:
  - **z** is a vector of indicator variables $\boldsymbol{z}^t = \{z_1^t, \ldots, z_k^t\}$
  - **z** acts like **r** in the supervised case
  - π are prior probabilities $P(G_i)$
  - We assume Gaussian components $\hat{p}_i(\boldsymbol{x}^t|\Phi) \sim \mathcal{N}(\boldsymbol{m}_i, \mathbf{S}_i)$

Summary

- Why learning from unlabeled data is important?
- Clustering algorithms
  - Hierarchical clustering
  - K-means
  - Gaussian mixture models
  - Expectation Maximization
- Non-parametric methods
  - Probability density estimation
- Dimensionality reduction/data embedding

Looking ahead

- Virtual office hour
- https://usc.zoom.us/j/95136500603?pwd=VEJhblhWK25lT2N3RC9FNWk3eTJKQT09