CLUSTERING

Based on

“Foundations of Statistical NLP”, C. Manning & H. Schütze, MIT Press, 2002, ch. 14

and “Machine Learning”, T. Mitchell, McGRAW Hill, 1997, ch. 6.12

- **Clustering:**
  partition a set of objects into groups/clusters.
- **The goal: place objects which are similar (according to a certain *similarity measure*) in a same group, and assign dissimilar objects to different groups.**
- **Objects are usually described and clustered using a set of features and values (often known as *the data representation model*).**

Clustering vs Classification

Classification = supervised learning,
i.e. we need a set of labeled training instances for each group/class.

Clustering = unsupervised learning,
because there is no teacher who provides the examples in the training set with class labels.
It assumes no pre-existing categorization scheme;
the clusters are induced from data.

Clustering: Overview

- **Hierarchical clustering**
  - Bottom-up (agglomerative) clustering
  - Top-down (divisive) clustering
- **Non-hierarchical clustering**
  - k-means
  - EM for estimating k-means
- **Soft vs hard assignments** in clustering
- **Similarity functions** in clustering:
  single link, complete link, group average
- **Application:**
  cluster-based improvement of language modeling

Hierarchical vs Non-hierarchical Clustering

Hierarchical Clustering

produces a tree of groups/clusters, each node being a subgroup of its mother.

Non-hierarchical Clustering (or, flat clustering):

the relation between clusters is often left undetermined.

Most non-hierarchical clustering algorithms are iterative. They start with a set of initial clusters and then iteratively improve them using a reallocation scheme.

Soft vs Hard Assignments in Clustering

Hard assignment:

each object is assigned to one and only one cluster.
This is the typical choice for hierarchical clustering.

Soft assignment: allows degrees of membership, and membership in multiple clusters.

In a vector space model, the degree of membership of $\overline{x}$ in multiple clusters can be: the distance between x and the centroid (or, center of gravity) of each cluster c:

$$\overline{\mu} = \frac{1}{\mid c \mid} \sum_{\overline{x} \in c} \overline{x}$$

Non-hierarchical clustering works with both hard assignments and soft assignments.

Hierarchical/Non-hierarchical Clustering: Pros and Cons

Hierarchical Clustering:

- ***preferable for detailed data analysis:*** **provides more infos than non-hierarchical clustering;**
- ***less efficient than non-hierarchical clustering:*** **one has to compute at least $n \times n$ similarity coefficients and then update them during the clustering process.**

Non-hierarchical Clustering:

- ***preferable if data sets are very large, or efficiency is a key issue*****;**
- **the k-means algo is conceptually the simplest method and should be used first on a new data set (its results are often sufficient);**
- **k-means (using a simple Euclidian metric), is** ***not usable on "nominal" data*** **like colours. In such cases, use the EM algorithm.**

1. Hierarchical Clustering

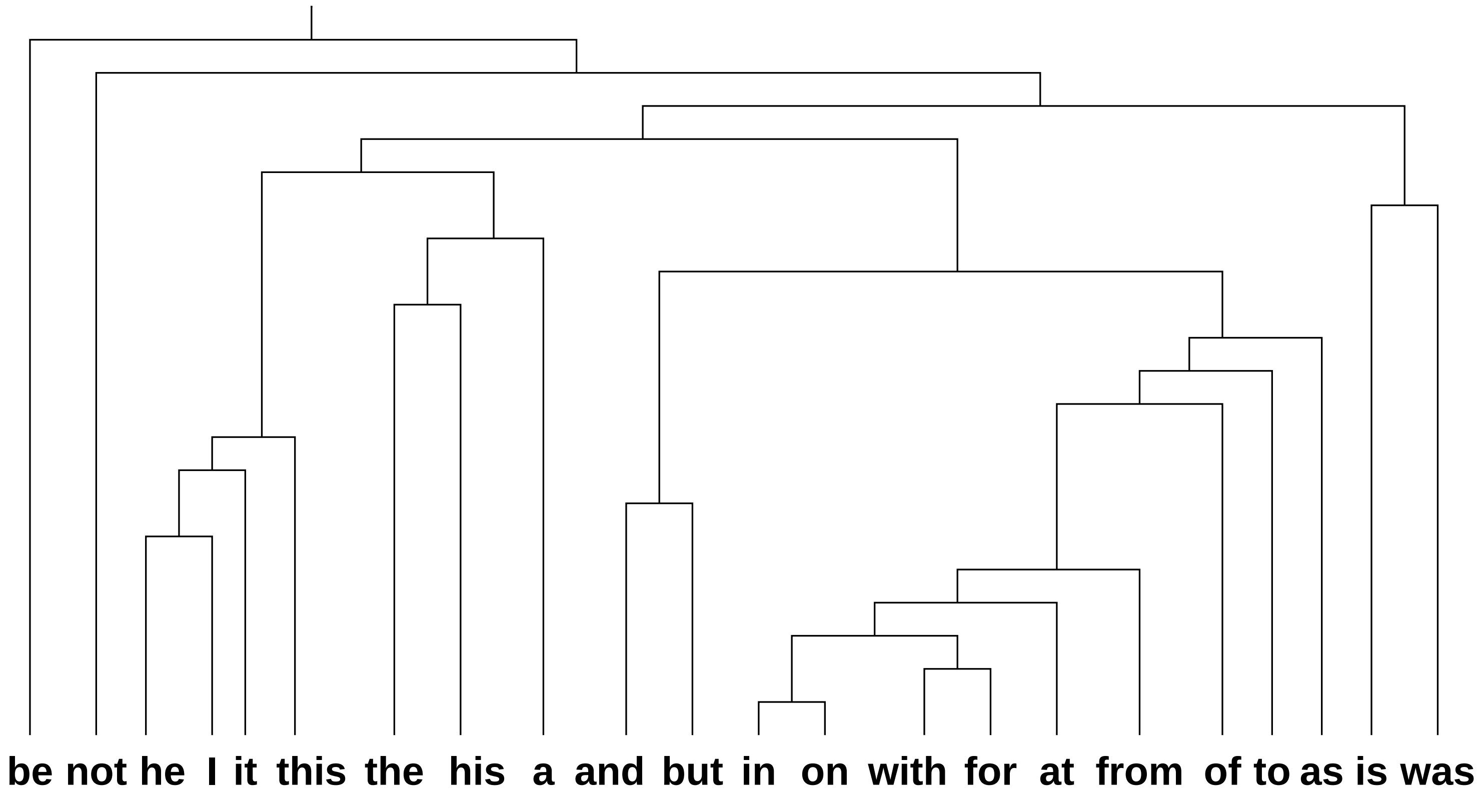
Botom-up (Agglomerative) Clustering:

Start with the individual objects.
Greedly put objects with “maximum similarity” (or “minimum distance”) together in a cluster.
Continue until all objects are contained in a single cluster.

Top-down (Divisive) Clustering:

Start with a cluster containing all objects.
Greedly split the cluster into two, assigning objects to clusters so to maximize the within-group similarity.
Continue splitting clusters which are the least coherent until have clusters containing a single object or the number of desired clusters.

An Example: A Dendogram showing a clustering of 22 high frequency words from the Brown corpus


be not he I it this the his a and but in on with for at from of to as is was

The Dendogram Commented

- **Similarity in this case is based on the left and right context of words. (Firth: “one can characterize a word by the words that occur around it”.)**

◦ **For instance:**
he, I, it, this have more in common with each other than they have with _and, but_;
_in, on_ have a greater similarity than _he, I_.

- **Each node in the tree represents a cluster that was created by merging two child nodes.**

- **The height of a connection corresponds to the apparent (di)similarity between the nodes at the bottom of the diagram.**

Exemplifying the Main Uses of Clustering (I)

Generalisation

We want to figure out the correct preposition to use with the noun *Friday* when translating a text from French into English.

The days of the week get put in the same cluster by a clustering algorithm which measures similarity of words based on their contexts.

Under the assumption that an environment that is correct for one member of the cluster is also correct for the other members,
we can infer the correctness of *on Friday* from the presence (in the given corpus) of *on Sunday, on Monday.*

Main Uses of Clustering (II)
Exploratory Data Analysis (EDA)

Any thechnique that lets one visualise the data better is likely to

- **bring to the fore new generalisations, and**
- **stop one from making wrong assumptions about data.**

This is a 'must' for domains like Statistical Natural Language Processing.

The Bottom-up Hierarchical Clustering Algo

```
Given: a set $X = \{x_1, \ldots, x_n\}$ of objects
       a function sim: $\mathcal{P}(X) \times \mathcal{P}(X) \rightarrow R$
for $i = 1, n$ do
    $c_i = \{x_i\}$ end
$C = \{c_1, \ldots, c_n\}$
$j = n + 1$
while $\mid C \mid > 1$
        $(c_{n_1}, c_{n_2}) = \text{argmax}_{(c_u, c_v) \in C \times C}$ sim$(c_u, c_v)$
        $c_j = c_{n_1} \cup c_{n_2}$
        $C = C \backslash \{c_{n_1}, c_{n_2}\} \cup \{c_j\}$
        $j = j + 1$
```

The Top-down Hierarchical Clustering Algo

```
Given: a set $X = \{x_1, \ldots, x_n\}$ of objects
       a function coh: $\mathcal{P}(X) \rightarrow R$
       a function split: $\mathcal{P}(X) \rightarrow \mathcal{P}(X) \times \mathcal{P}(X)$
$C = \{X\}(= \{c_1\})$
$j = 1$
while $\exists c_i \in C$ such that $\mid c_i \mid > 1$
        $c_u = \text{argmin}_{c_v \in C}$ coh$(c_v)$
        $c_{j+1} \cup c_{j+2} =$ split $(c_u)$
        $C = C \backslash \{c_u\} \cup \{c_{n_1}, c_{n_2}\}$
        $j = j + 2$
```

Top-down Hierarchical Clustering: Further Comments

- **Similarity functions (see next slide) can be used here also as coherence.**

  ◦ **In general, if d is a distance measure, then one can take $\text{sim}(x,y) = \frac{1}{1+d(x,y)}$.**

- **To split a cluster in two sub-clusters:**
  any bottom-up or non-hierarchical clustering algos can be used;
  better use the relative entropy (the Kulback-Leibler (KL) divergence):

$$D(p \,||\, q) = \sum_{x \in \mathcal{X}} p(x) \boldsymbol{log} \frac{p(x)}{q(x)}$$

  where $(0\ \boldsymbol{log} \frac{0}{q} = 0$**, and** $p\ \boldsymbol{log} \frac{p}{0} = \infty)$**.**

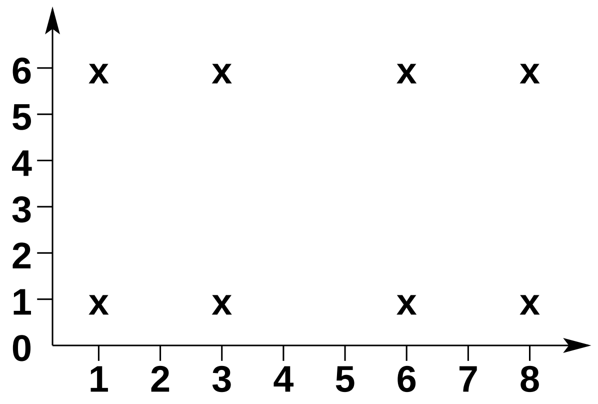
Monotonicity of the Similarity Function

The operation of merging is guaranteed not to increase the similarity:

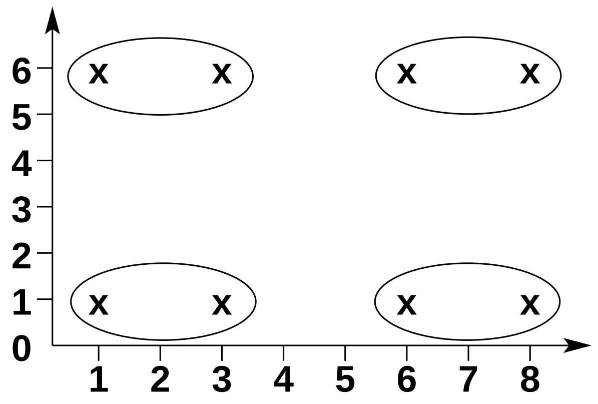
$$\forall c, c', c'' : \mathbf{min}(\mathbf{sim}(c, c'), \mathbf{sim}(c, c'')) \geq \mathbf{sim}(c, c' \cup c'').$$

Classes of Similarity Functions

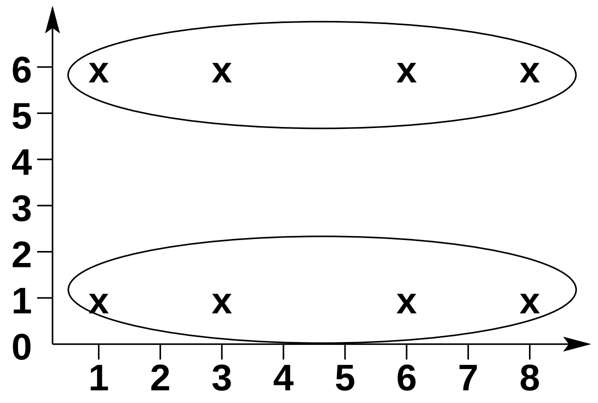
- single link: similarity of two clusters considered for merging is determined by the two most similar members of the two clusters
- complete link: similarity of two clusters is determined by the two least similar members of the two clusters
- group average: similarity is determined by the average similarity between all members of the clusters considered.



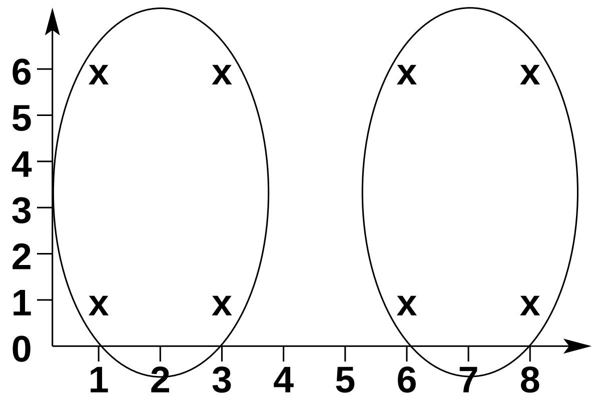
a set of points in a plane



first step in single/complete clustering



single−link clustering



complete−link clustering

Single-link vs Complete-link Clustering: Pros and Cons

Single-link Clustering:

- good local coherence, since the similarity function is locally defined
- can produce elongated clusters ("the chaining effect")
- Closely related to the Minimum Spanning Tree (MST) of a set of points.
  (Of all trees connecting the set of objects, the sum of the edges of the MST is minimal.)
- In graph theory, it corresponds to finding a maximally connected graph. Complexity: $O(n^2)$.

Complete-link Clustering:

- The focuss is on the global cluster quality.
- In graph theory, it corresponds to finding a maximally complete clique. Complexity: $O(n^3)$.

Group-average Agglomerative Clustering

The cirterion for merges: average similarity, which in some cases can be efficiently computed, implying $O(n^2)$. For example, one can take

$$sim(\overline{x}, \overline{y}) = cos(\overline{x}, \overline{y}) = \frac{\overline{x} \cdot \overline{y}}{\mid \overline{x} \mid\mid \overline{y} \mid} = \sum_{i=1}^{m} x_i y_i$$

with $\overline{x}, \overline{y}$ being length-normalised, i.e., $\mid \overline{x} \mid = \mid \overline{y} \mid = 1$.

Therefore, it is a good compromise between single-link and complete-link clustering.

Group-average Agglomerative Clustering: Computation

Let $\mathcal{X} \subseteq R^m$ **be the set of objects to be clustered**
The average similarity of a cluster c_j **is:**

$$S(c_j) = \frac{1}{\mid c_j \mid (\mid c_j \mid -1)} \sum_{\overline{x} \in c_j} \sum_{\overline{y} \neq \overline{x} \in c_j} sim(\overline{x}, \overline{y})$$

Considering $\overline{s}(c_j) = \sum_{\overline{x} \in c_j} \overline{x}$**, then:**

$$\overline{s}(c_j) \cdot \overline{s}(c_j) = \sum_{\overline{x} \in c_j} \sum_{\overline{y} \in c_j} \overline{x} \cdot \overline{y} = \mid c_j \mid (\mid c_j \mid -1) S(c_j) + \sum_{\overline{x} \in c_j} \overline{x} \cdot \overline{x} = \mid c_j \mid (\mid c_j \mid -1) S(c_j) + \mid c_j \mid$$

Therefore:

$$S(c_j) = \frac{\overline{s}(c_j) \cdot \overline{s}(c_j) - \mid c_j \mid}{\mid c_j \mid (\mid c_j \mid -1)}$$

and

$$S(c_i \cup c_j) = \frac{(\overline{s}(c_i) + \overline{s}(c_j)) \cdot (\overline{s}(c_i) + \overline{s}(c_j)) - (\mid c_i \mid + \mid c_j \mid)}{(\mid c_i \mid + \mid c_j \mid)(\mid c_i \mid + \mid c_j \mid -1)}$$

which requires constant time for computing.

Application:

Clustering Improves Language Modeling

[Brown et al., 1992], [Manning & Schuetze, 1992], pages 509–512

Using cross-entropy ($-\frac{1}{N} log P(w_1, \ldots, w_N)$) and bottom-up clustering, Brown obtained a cluster-based language model which didn't prove better than the word-based model.
But the linear interpolation of the two models was better than both!

Example of 3 clusters obtained by Brown:

- **plan, letter, request, memo, case, question, charge, statement, draft**
- **day, year, week, month, quarter, half**
- **evaluation, assessment, analysis, understanding, opinion, conversation, discussion**

Note that the words in these clusters have similar syntactic and semantic properties.

2. Non-hierarchical Clustering

As already mentioned, an initial partition of selected seeds (one seed for each cluster), is then iteratively refined.

Stopping criteria (examples):

– group-average similarity

– mutual information between adjiacent clusters

– the likelyhood of data, given the clusters

– determine the number of clusters, using (for example) the Minimum Description Length (MDL) principle.

Note: The initial centers for clusters can be computed by applying a hierarchical clustering algo on a subset of the objects to be clustered (especially in the case of ill-behaved sets).

An Example of Non-hierarchical Clustering

The Estimation of the Means of k Gausseans — the Problem

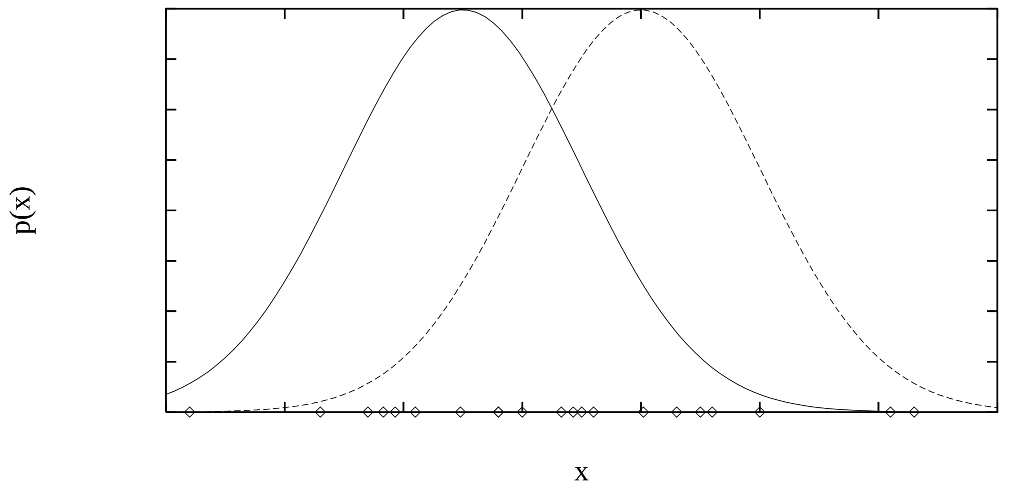
Given

- D, a set of instances from X generated (using uniform probability) by a mixture of k Gaussian distributions;
- the unknown means $\langle \mu_1, \ldots, \mu_k \rangle$ of the k Gaussians;
- ○ to simplify the presentation, all Gaussians are assumed to have the same variance σ^2, and they are selected with equal probability;
- we don't know which x_i was generated by which Gaussian;

determine

- h, the ML estimates of $\langle \mu_1, \ldots, \mu_k \rangle$, i.e. $\mathrm{argmax}_h\, P(D \mid h)$.

Generating Data from a Mixture of k Gaussians



Each instance x is generated by

1. **Choosing one of the k Gaussians (having the same variance σ^2), with uniform probability;**
2. **Generating randomly an instance according to that Gaussian.**

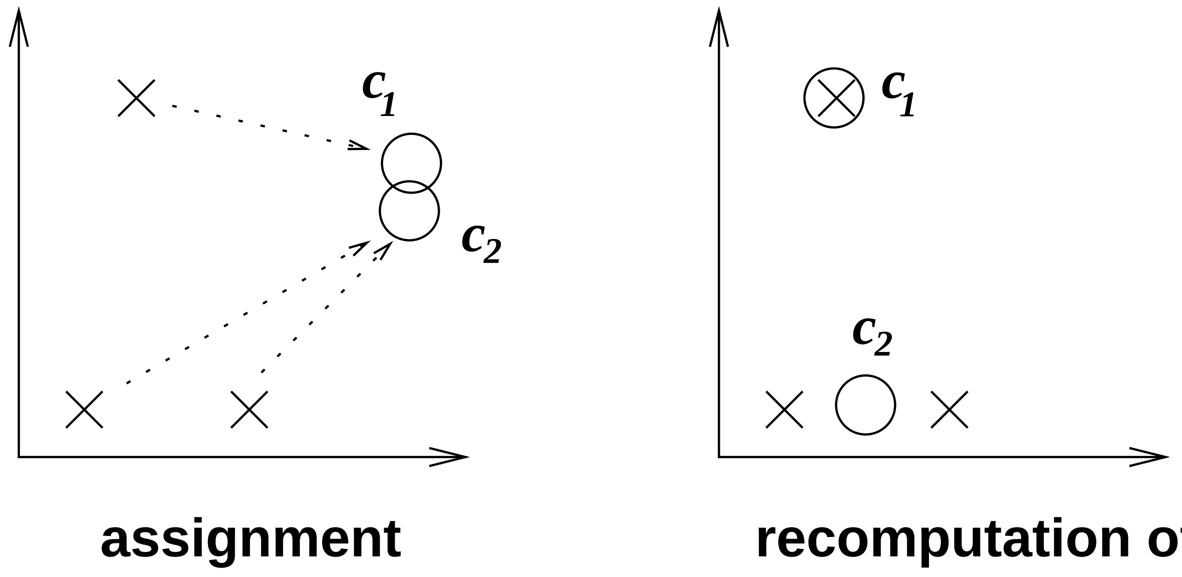
Remark: For $k = 1$ we have already shown that the ML hypothesis is the one that minimizes the sum of squared errors:

$$\mu_{ML} = \operatorname*{argmin}_{\mu} \sum_{i=1}^{m} (x_i - \mu)^2 = \frac{1}{m} \sum_{i=1}^{m} x_i$$

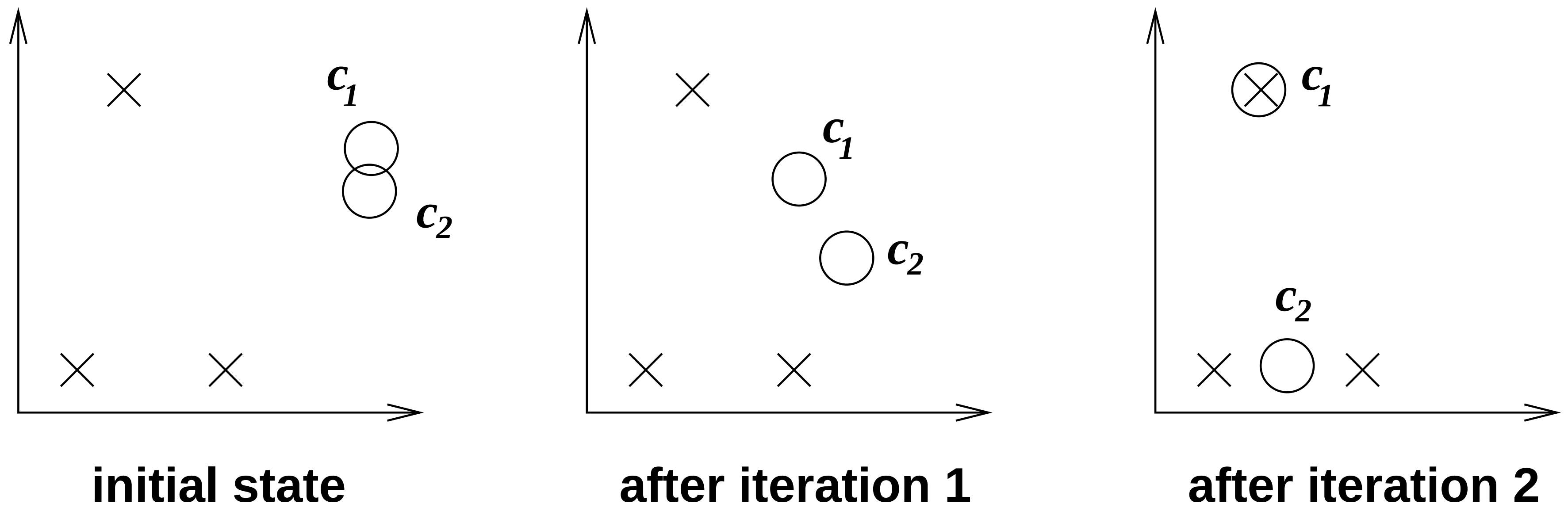
2.1 k-Means Clustering Algorithm

```
Given a set $X = \{x_1, \ldots, x_n\} \subseteq \mathcal{R}^m$
        a distance measure $d$ on $\mathcal{R}^m$
        a function for computing the mean $\mu : \mathcal{P}(\mathcal{R}^m) \rightarrow \mathcal{R}^m$
Select (arbitrarily) $k$ initial centers $f_1, \ldots, f_k$ in $\mathcal{R}^m$
while the stopping criterion is not true
    for all clusters $c_j$ do $c_j = \{x_i \mid \forall f_l \; d(x_i, f_j) \leq d(x_i, f_l)\}$ end
    for all means $f_j$ do $f_j \leftarrow \mu_j$ end
end
```

Illustrating the k-Means Clustering Algorithm



2.2. EM for k-Means Clustering



Remark: The EM algorithm for estimating the means of k Gausseans can be seen as a "soft" version of the previous ("hard") clustering algorithm

Using EM for k-Means Clustering

Notations:

Going back to our initial example ($k = 2$), and

referring to the formulation we used define the to *k-means problem*,

think of the full description of each instance as $y_i =< x_i, z_{i1}, z_{i2} >$, where

- z_{ij} **is 1 if** x_i **generated by** j**th Gaussian**
- x_i **is observable,** z_{ij} **is unobservable**

EM for Estimating k Means: Algorithm Overview

Initial step: **Pick at random $h = \langle \mu_1, \mu_2 \rangle$, then iterate:**

Expectation step: **Assuming that the current hypothesis $h = \langle \mu_1, \mu_2 \rangle$ holds, calculate the expected value $E[z_{ij}]$ of each hidden variable z_{ij}**

$$E[z_{ij}] = \frac{p(x = x_i | \mu = \mu_j)}{\sum_{l=1}^{2} p(x = x_i | \mu = \mu_l)} = \frac{e^{-\frac{1}{2\sigma^2}(x_i - \mu_j)^2}}{\sum_{l=1}^{2} e^{-\frac{1}{2\sigma^2}(x_i - \mu_l)^2}}$$

Maximization step: **Calculate a new ML hypothesis $h' = \langle \mu'_1, \mu'_2 \rangle$, assuming the value taken on by each hidden variable z_{ij} is its expected value $E[z_{ij}]$ calculated above.**

$$\mu'_j \leftarrow \frac{\sum_{i=1}^{m} E[z_{ij}]\ x_i}{\sum_{i=1}^{m} E[z_{ij}]} = \frac{1}{m} \sum_{i=1}^{m} E[z_{ij}]\ x_i$$

Replace $h =< \mu_1, \mu_2 >$ by $h' =< \mu'_1, \mu'_2 >$.

EM for Estimating k Means: Calculus (I)

$$p(y_i|h') = p(x_i, z_{i1}, \ldots, z_{ik}|h') = \frac{1}{\sqrt{2\pi\sigma^2}}\; e^{-\frac{1}{2\sigma^2}\sum_{j=1}^{k} z_{ij}(x_i-\mu'_j)^2}$$

$$\begin{aligned}\ln P(Y|h') &= \ln \prod_{i=1}^{m} p(y_i|h') = \sum_{i=1}^{m} \ln p(y_i|h') \\ &= \sum_{i=1}^{m}(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}\sum_{j=1}^{k} z_{ij}(x_i-\mu'_j)^2)\end{aligned}$$

$$E[\ln P(Y|h')] = \sum_{i=1}^{m}(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}\sum_{j=1}^{k} E[z_{ij}](x_i-\mu'_j)^2)$$

EM for Estimating k Means: Calculus (II)

Expectation step: **(already shown on a previous slide)**

$$E[z_{ij}] \;=\; \frac{p(x = x_i|\mu = \mu_j)}{\sum_{l=1}^{k} p(x = x_i|\mu = \mu_l)} = \frac{e^{-\frac{1}{2\sigma^2}(x_i-\mu_j)^2}}{\sum_{l=1}^{k} e^{-\frac{1}{2\sigma^2}(x_i-\mu_l)^2}}$$

Maximization step:

$$\begin{aligned}\operatorname*{argmax}_{h'} P(Y|h') &= \operatorname*{argmax}_{h'} \sum_{i=1}^{m} (\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{j=1}^{k} E[z_{ij}](x_i - \mu'_j)^2) \\ &= \operatorname*{argmin}_{h'} \sum_{i=1}^{m} \sum_{j=1}^{k} E[z_{ij}](x_i - \mu'_j)^2\end{aligned}$$

$$\mu_j \leftarrow \frac{\sum_{i=1}^{m} E[z_{ij}]\ x_i}{\sum_{i=1}^{m} E[z_{ij}]} = \frac{1}{m} \sum_{i=1}^{m} E[z_{ij}]\ x_i$$

EM for Estimating k Means: Justification

It can be shown (Baum et al. 1970) that after each iteration $P(D \mid h)$ increases, unless it is a local maximum. Therefore the previously defined EM algorithm

- **converges to a (local) maximum likelihood hypothesis h,**
- **by providing iterative estimates of the hidden variables z_{ij}.**

In fact, as shown by the detailed calculus on the next slides, EM finds a local maximum of $E[\ln P(Y|h)]$, where

- **Y is complete set of (observable plus unobservable) variables/data**
- **the expected value of $\ln P(Y|h)$ is taken over possible values of unobserved variables in Y.**