UE: Machine Learning Fundamentals

Part II : Unsupervised, Semi-supervised Learning

Massih-Reza Amini

http://ama.liglab.fr/~amini/Cours/ML/ML.html

Université Grenoble Alpes
Laboratoire d'Informatique de Grenoble
Massih-Reza.Amini@imag.fr

Clustering

- The aim of clustering is to identify disjoint groups of observations within a given collection.
  - $\Rightarrow$ The aim is to find homogenous groups, by assembling observations that are close one to another, and separating the best those that are different
- Let G be a partition found over the collection $\mathcal{C}$ of N observations. An element of G is called *group* (or *cluster*). A group, G_k, where $1 \leq k \leq |G|$, corresponds to a subset of observations in $\mathcal{C}$.
- A representative of a group G_k, generally its center of gravity $\mathbf{r}_k$, is called prototype.

Classification vs. Clustering

- In **classification**: we have pairs of examples constituted by observations and their associated class labels $(\mathbf{x}, y) \in \mathbb{R}^d \times \{1, \ldots, K\}$.
  - The class information is provided by an expert and the aim is to find a prediction function $f : \mathbb{R}^d \to \mathcal{Y}$ that makes the association between the inputs and the outputs following the ERM or the SRM principle
- In **clustering**: the class information does not exist and the aim is to find homogeneous clusters or groups reflecting the relationship between observations.
  - The main hypothesis here is that this relationship can be found with the disposition of examples in the characteristic space,
  - The exact number of groups for a problem is very difficult to be found and it is generally fixed before hand to some arbitrary value,
  - The partitioning is usually done iteratively and it mainly depends on the initialization.

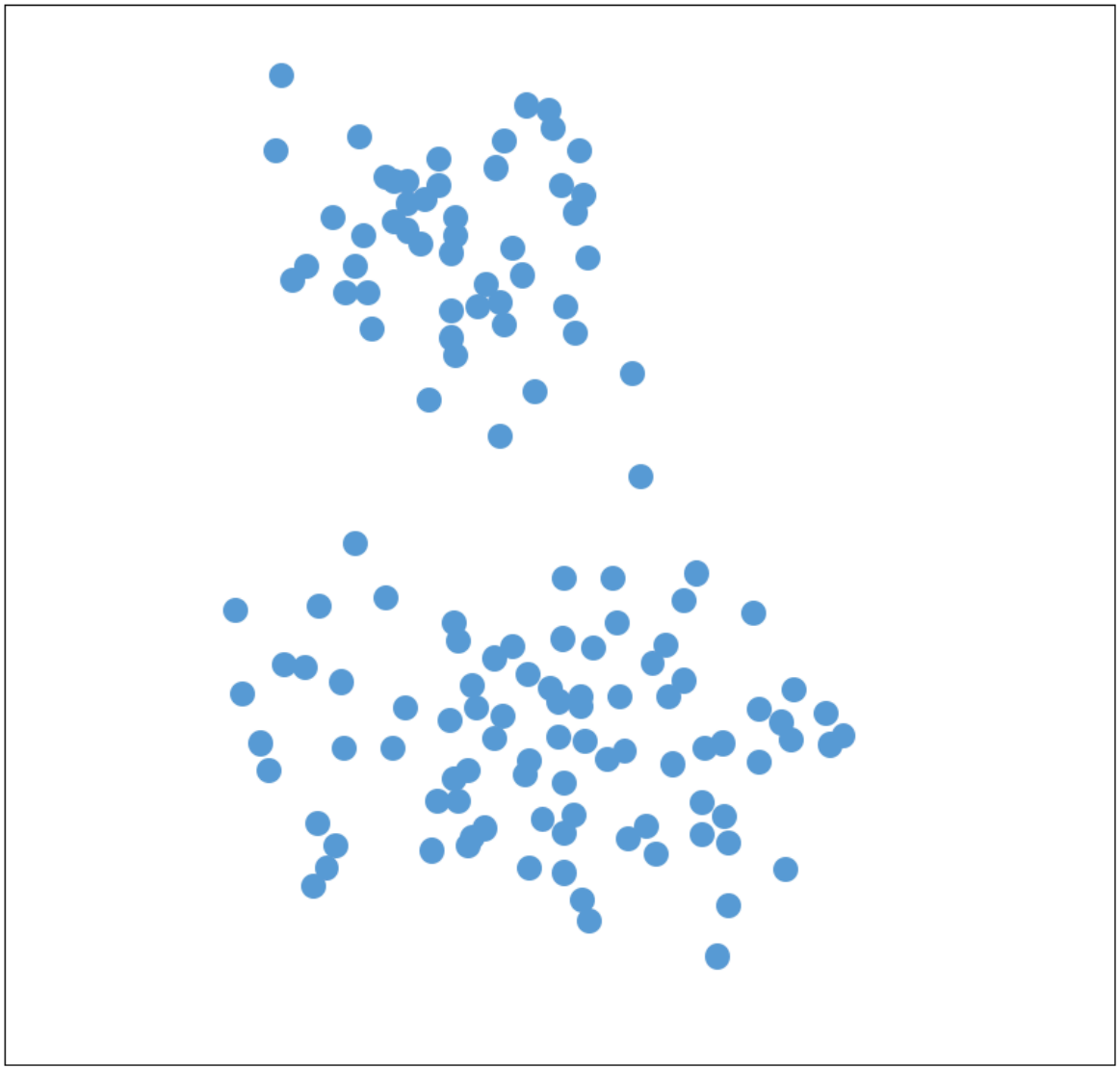
K-means algorithm [MacQueen, 1967]

- The K-means algorithm tends to find the partition for which the average distance between different groups is minimised:

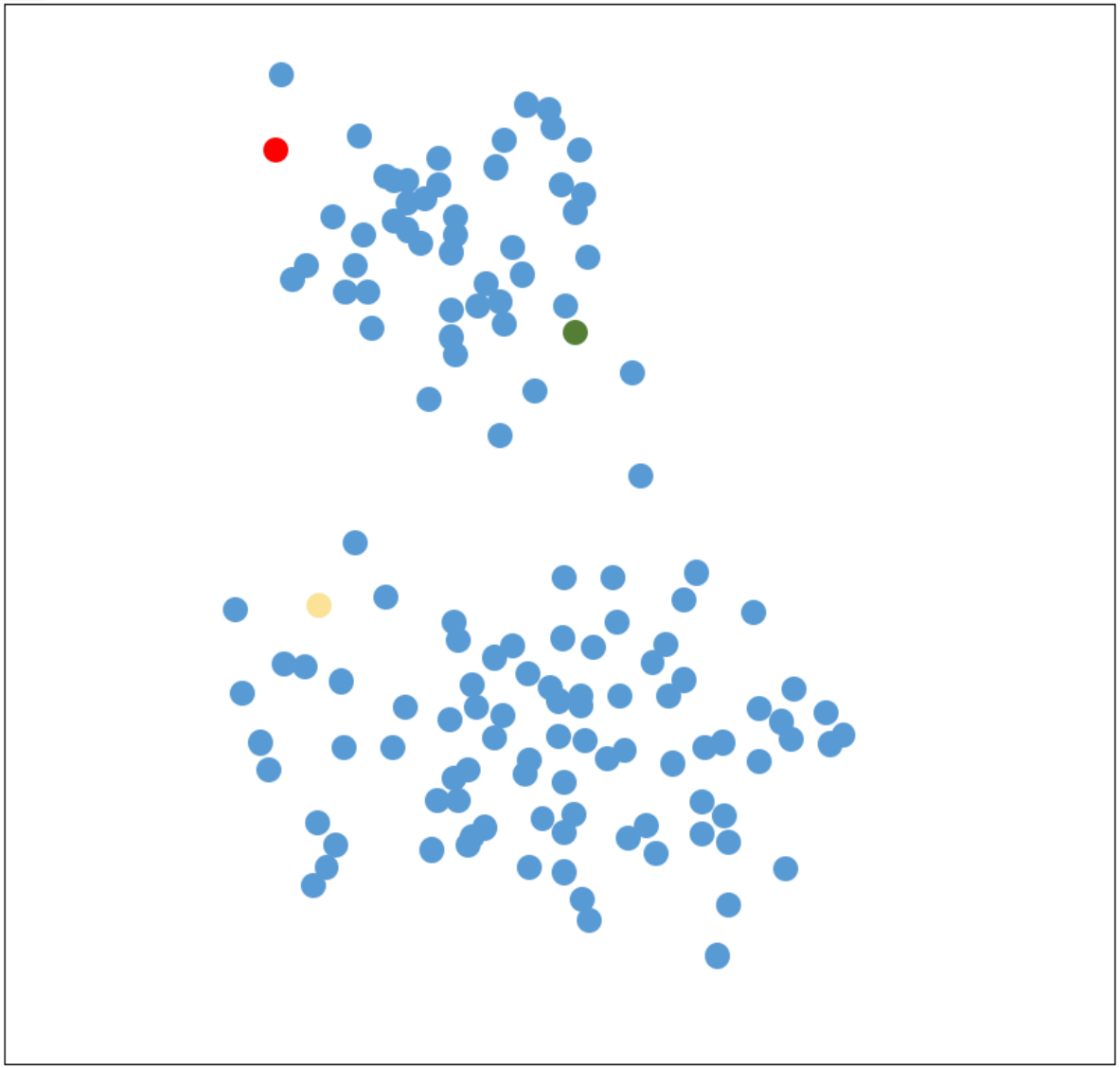
$$\text{argmin}_G \left(\sum_{k=1}^{K} \sum_{d \in G_k} ||\mathbf{x} - \mathbf{r}_k||_2^2 \right)$$

- From a given set of centroids, the algorithm then iteratively
  - affects each observation to the centroid to which it is the closest, resulting in new clusters;
  - estimates new centroids for the clusters that have been found.

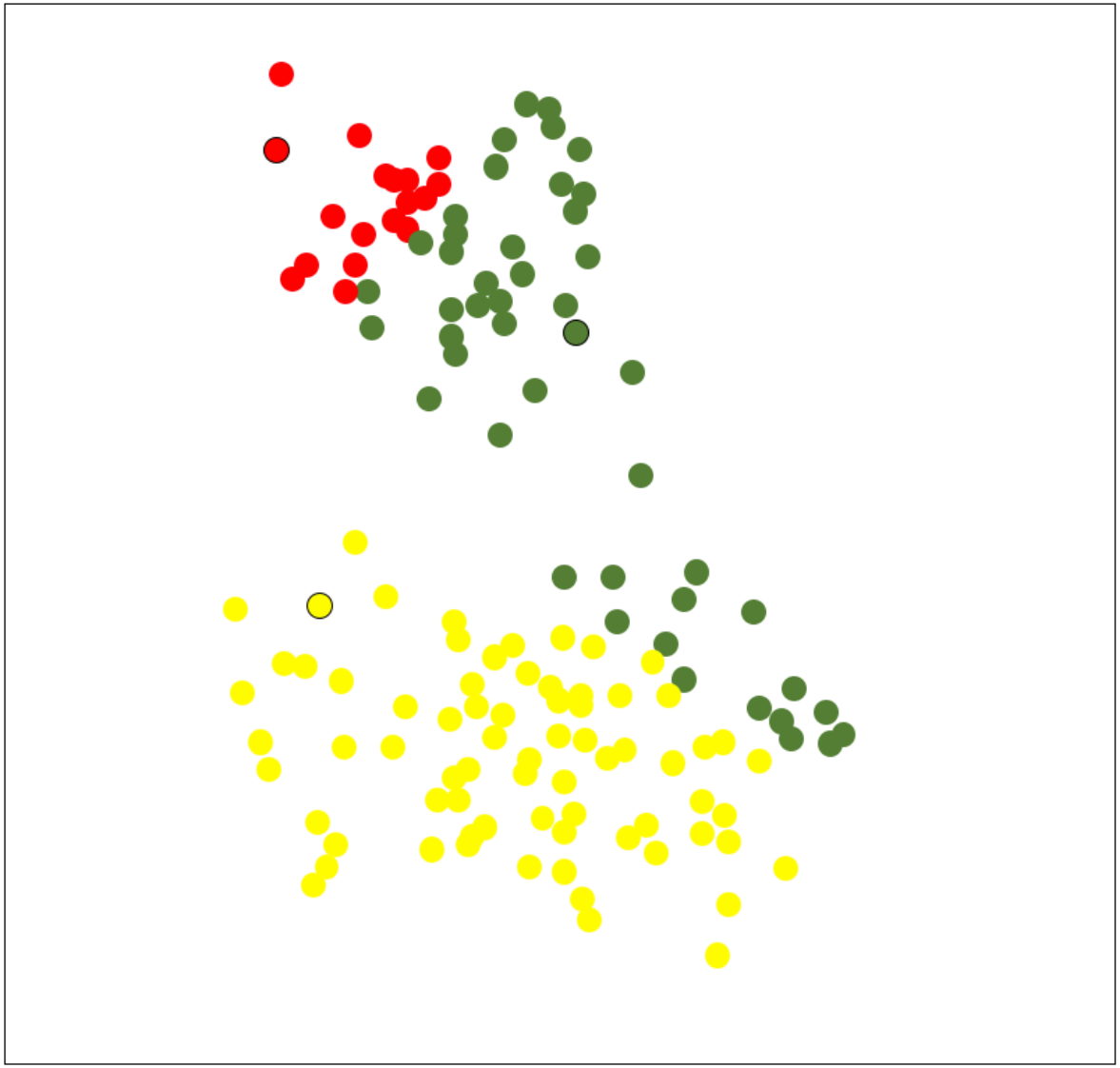
Clustering with K-means



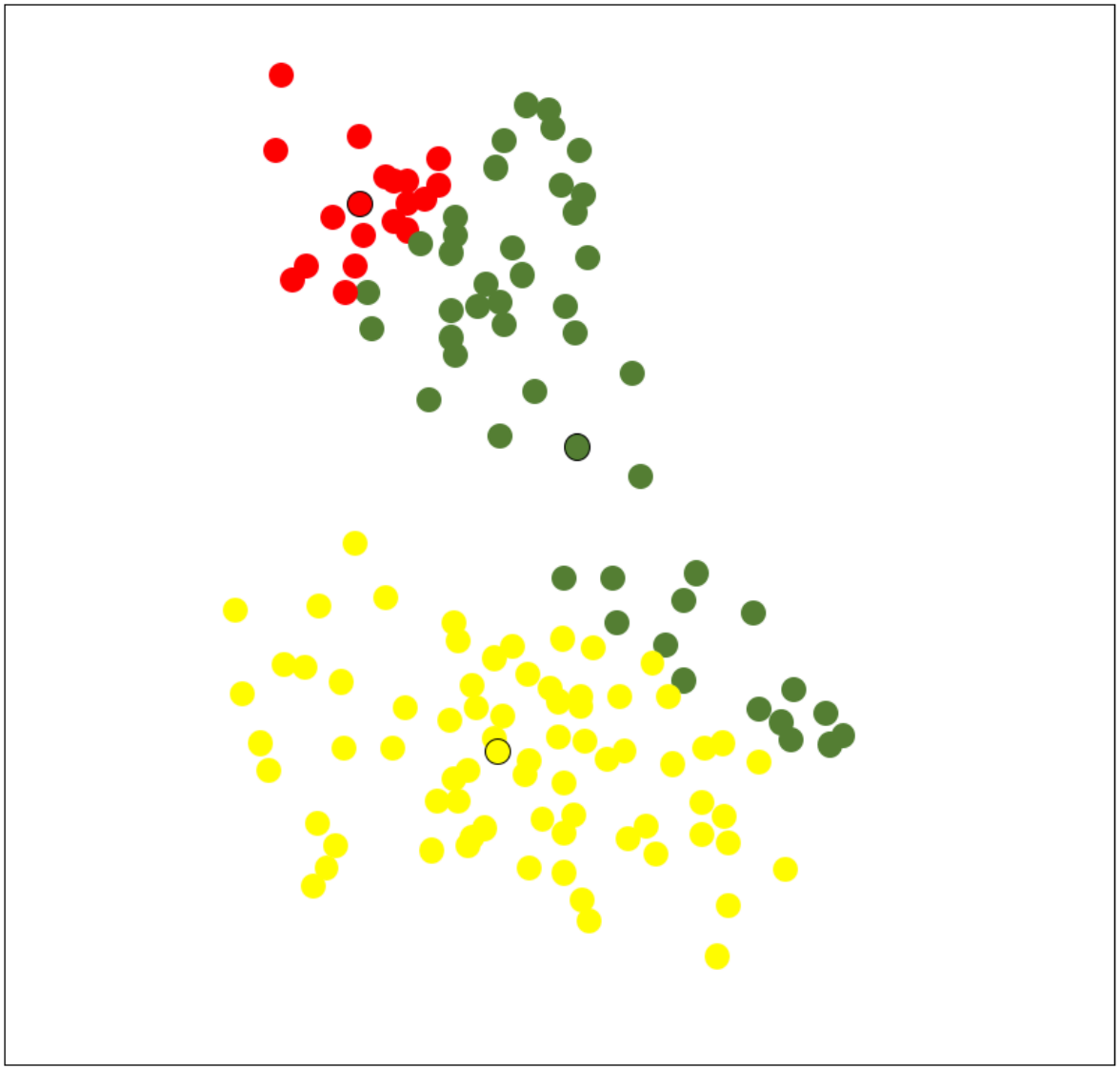
Clustering with K-means



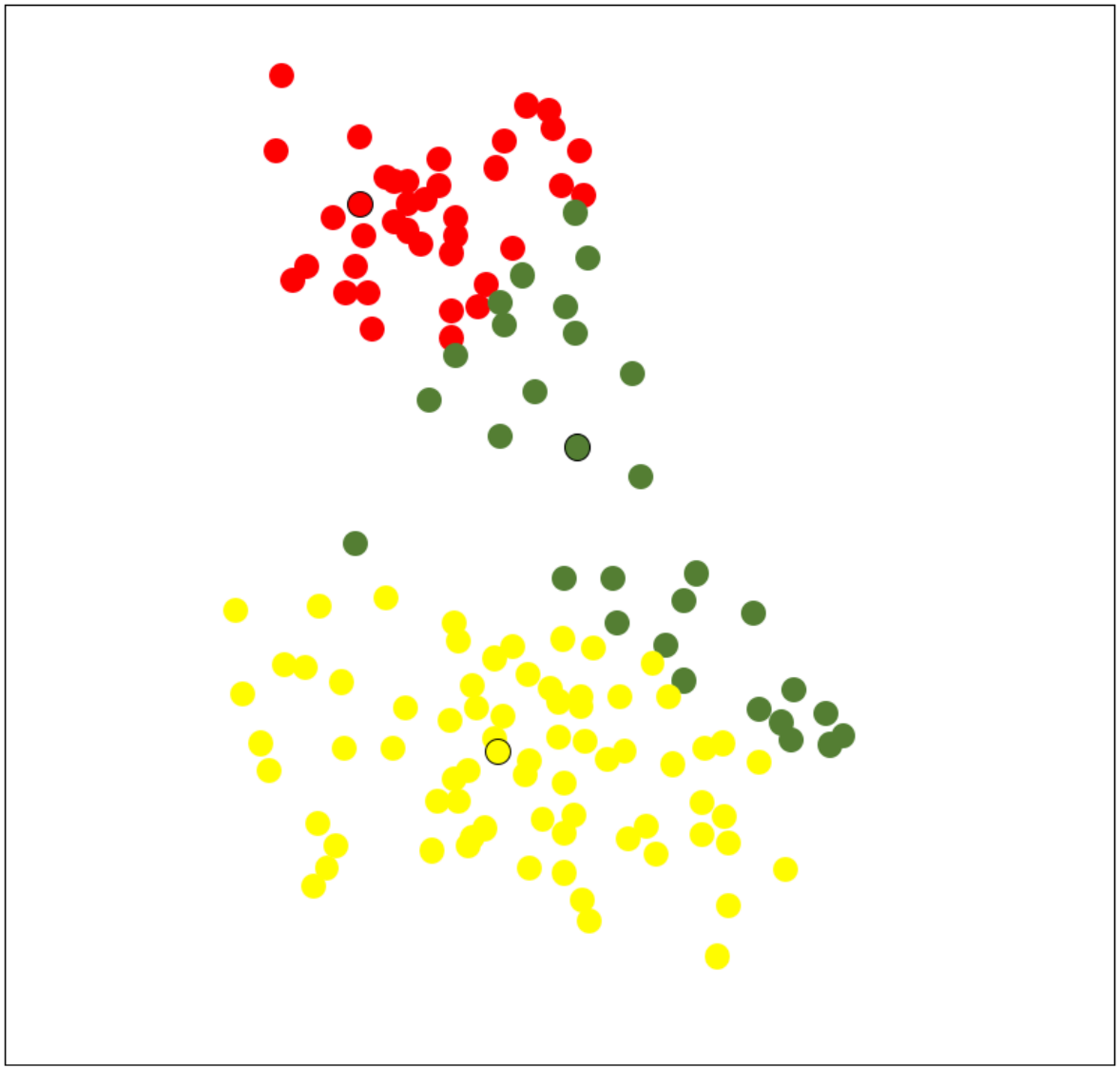
Clustering with K-means



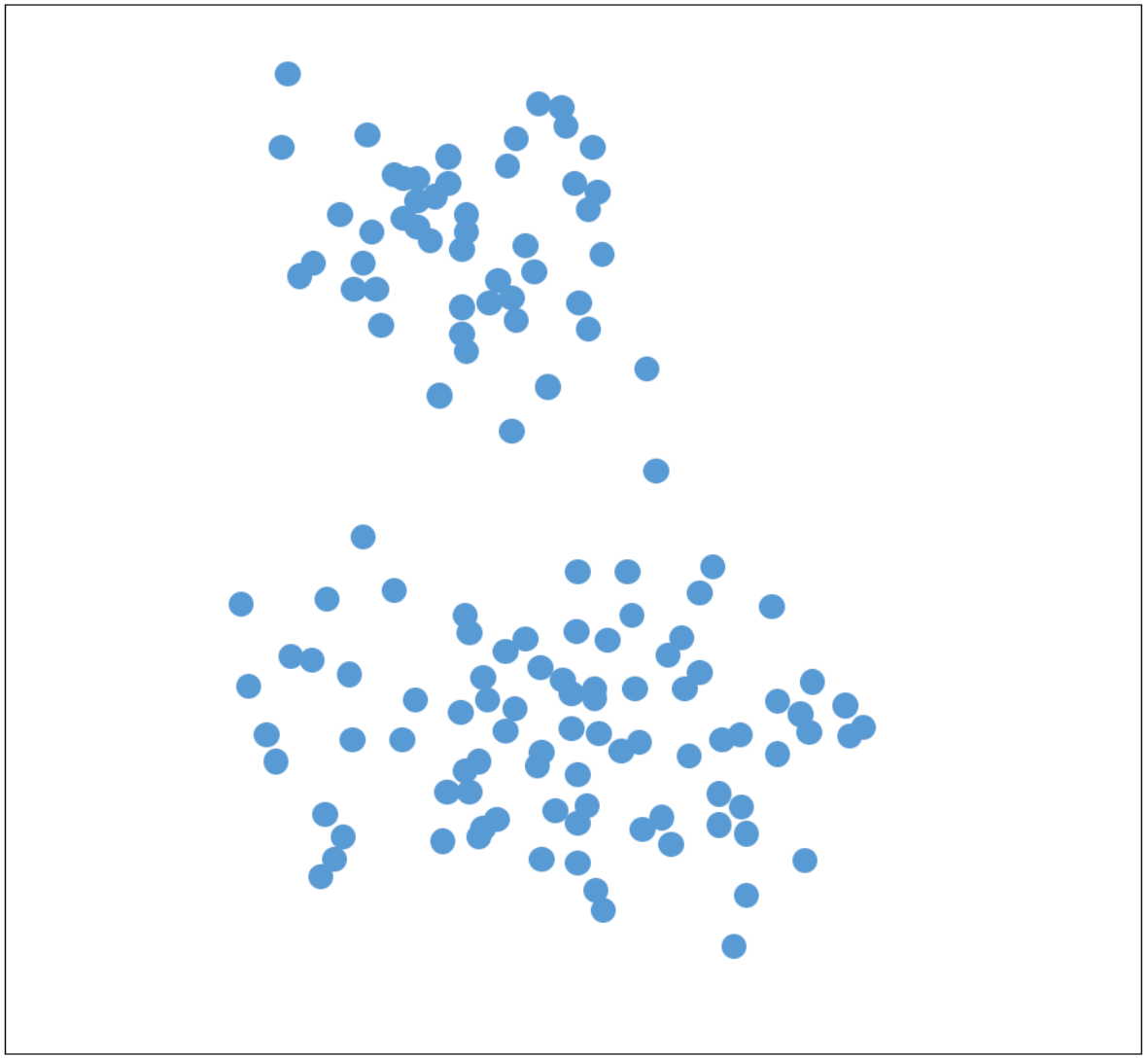
Clustering with K-means



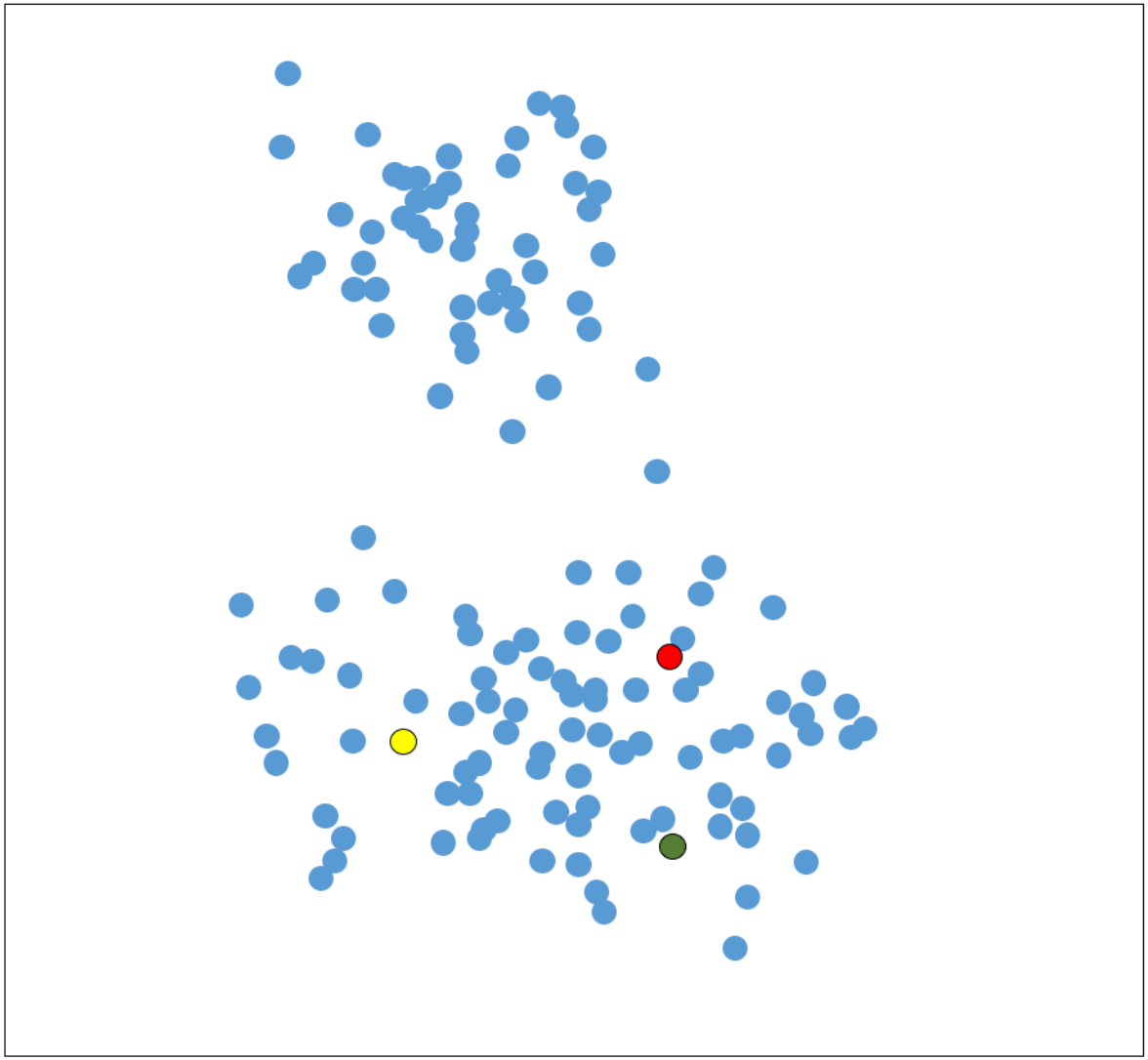
Clustering with K-means



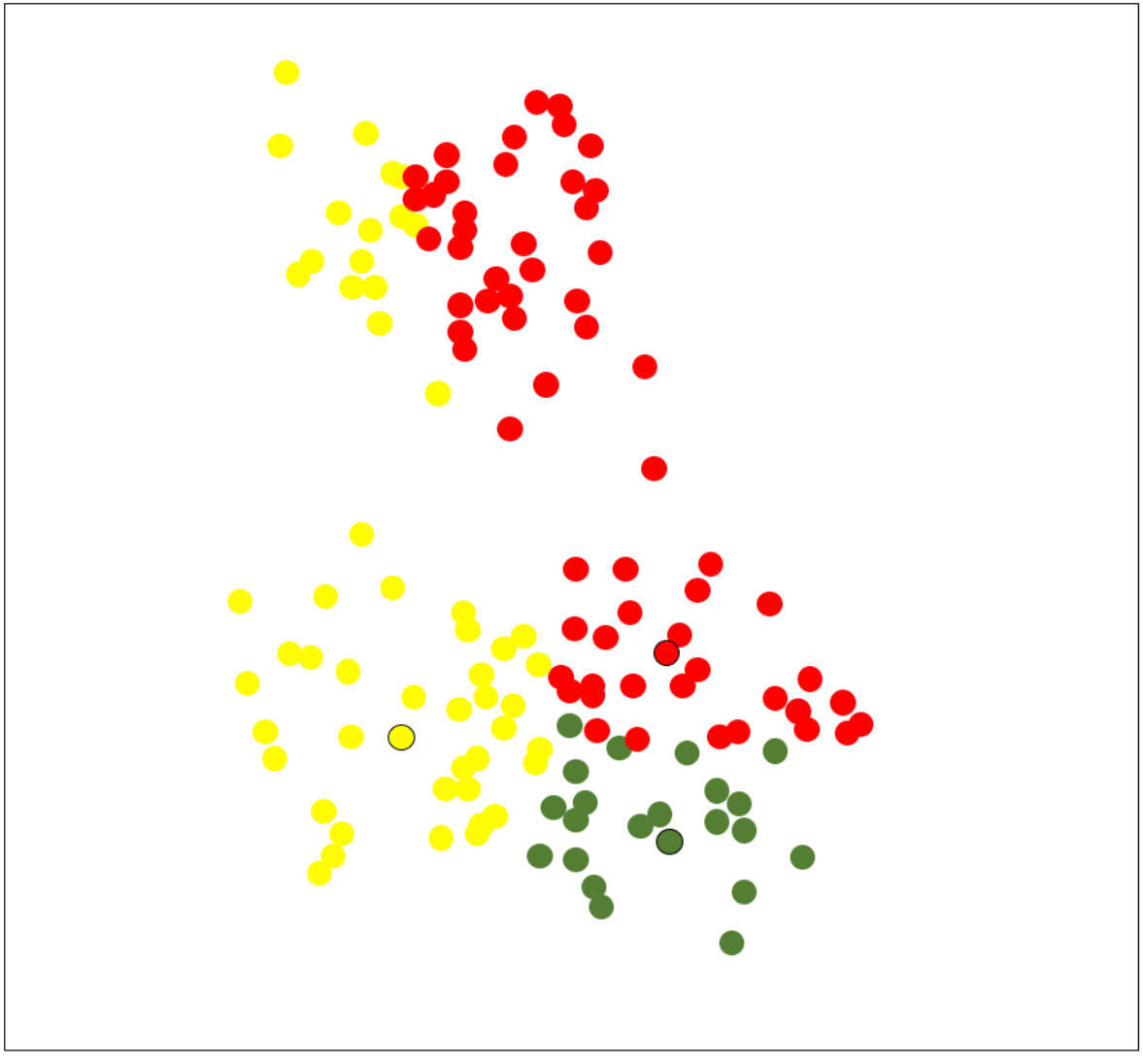
But also ...



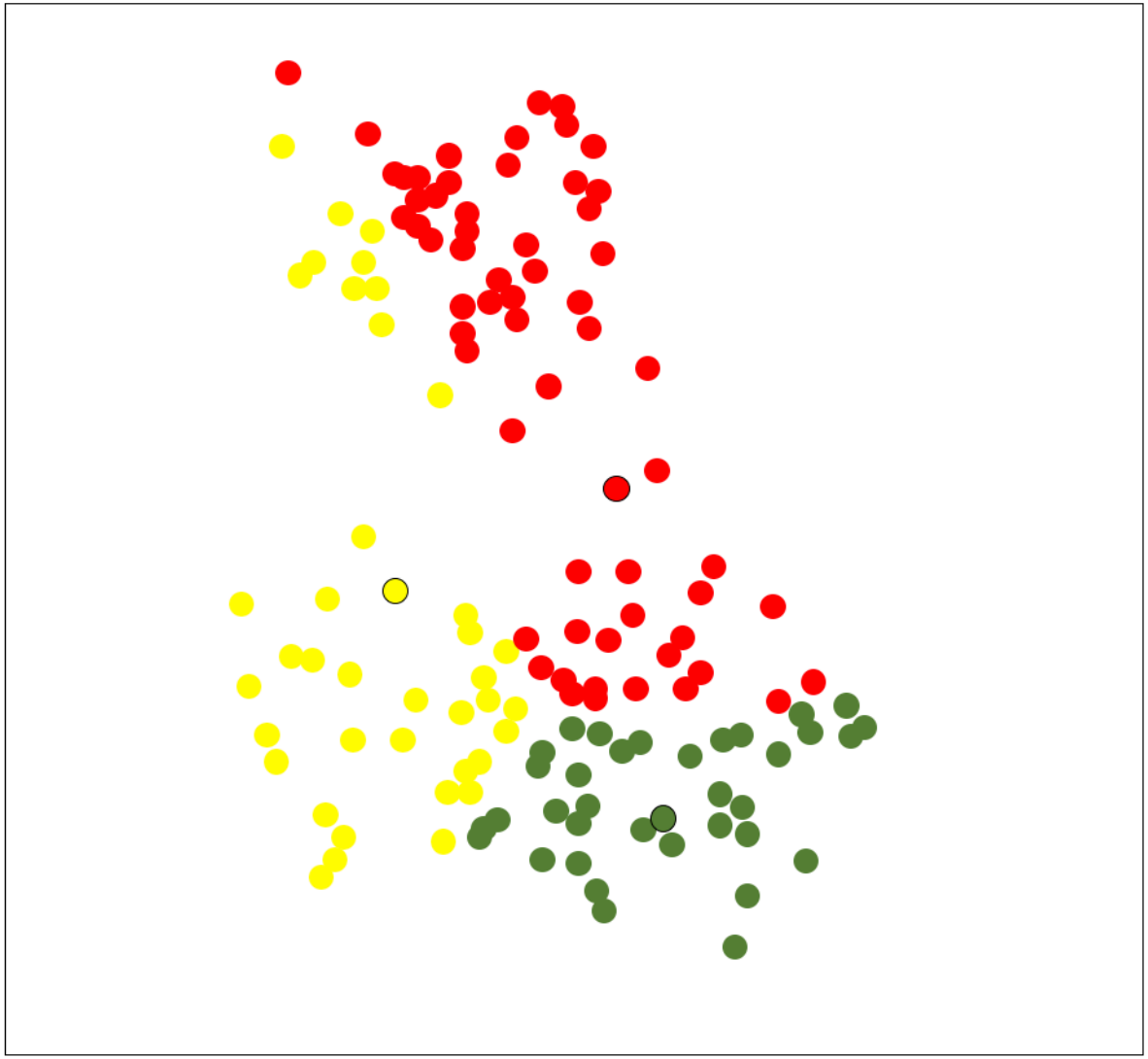
But also ...



But also ...



But also ...



Different forms of clustering

There are two main forms of clustering:

1. *Flat* partitioning, where groups are supposed to be independent one from another. The user then chooses a number of clusters and a threshold over the similarity measure.
2. *Hierarchical* partitioning, where the groups are structured in the form of a taxonomy, which in general is a binary tree (each group has two siblings).

Hierarchical partitioning

- The hierarchical tends to construct a tree and it can be realized
  - in *bottom-up* manner, by creating a tree from the observations (agglomerative techniques), or *top-down*, by creating a tree from its root (divisives techniques).
- Hierarchical methods are purely determinists and do not require that a number of groups to be fixed before hand.

Hierarchical partitioning

- The hierarchical tends to construct a tree and it can be realized
  - in *bottom-up* manner, by creating a tree from the observations (agglomerative techniques), or *top-down*, by creating a tree from its root (divisives techniques).
- Hierarchical methods are purely determinists and do not require that a number of groups to be fixed before hand.
- In opposite, their complexity is in general quadratique in the number of observations (N) !

Steps of clustering

Clustering is an iterative process including the following steps:

1. Choose a similarity measure and eventually compute a similarity matrix.
2. Clustering.
    a. Choose a family of partitioning methods.
    b. Choose an algorithm within that family.
3. Validate the obtained groups.
4. Return to step 2, by modifying the parameters of the clustering algorithm or the family of the partitioning family.

Similarity measures

There exists several similarity measures or distance, the most common ones are:

- *Jaccard* measure, which estimates the proportion of common termes within two documents. In the case where the feature characteristics are between 0 and 1, this measure takes the form:

$$\text{sim}_{\text{Jaccard}}(\mathbf{x}, \mathbf{x}') = \frac{\sum_{i=1}^{d} x_i x'_i}{\sum_{i=1}^{d} x_i + x'_i - x_i x'_i}$$

- *Dice* coefficient takes the form:

$$\text{sim}_{\text{Dice}}(\mathbf{x}, \mathbf{x}') = \frac{\sum_{i=1}^{d} x_i x'_i}{\sum_{i=1}^{d} x_i^2 + (x'_i)^2}$$

Similarity measures

- *cosine* similarity, writes:

$$\text{sim}_{\cos}(\mathbf{x}, \mathbf{x}') = \frac{\sum_{i=1}^{d} x_i x'_i}{\sqrt{\sum_{i=1}^{d} x_i^2}\sqrt{\sum_{i=1}^{d} (x'_i)^2}}$$

- *Euclidean* distance is given by:

$$\text{dist}_{\text{eucl}}(\mathbf{x}, \mathbf{x}') = ||\mathbf{x} - \mathbf{x}'||_2 = \sqrt{\sum_{i=1}^{d} (x_i - x'_i)^2}$$

This distance is then transformed into a similarity measure, by using for example its opposite.

Mixture models

- With the probabilistic approaches, we suppose that each group G_k is generated by a probability density of parameters θ_k
- Following the formula of total probabilities, an observation $\mathbf{x}$ is then supposed to be generated with a probability

$$P(\mathbf{x}, \Theta) = \sum_{k=1}^{K} \underbrace{P(y = k)}_{\pi_k} P(\mathbf{x} \mid y = k, \theta_k)$$

  where $\Theta = \{\pi_k, \theta_k; k \in \{1, \ldots, K\}\}$ are the parameters of the mixture.
- The aim is then to find the parameters Θ with which the mixture models fits the best the observations

Mixture models (2)

- If we have a collection of N observations, $\mathbf{x}_{1:N}$, the log-likelihood writes

$$\mathcal{L}_M(\Theta) = \sum_{i=1}^{N} \ln \left[\sum_{k=1}^{K} \pi_k P(\mathbf{x}_i \mid y = k, \theta_k) \right]$$

- The aim is then to find the parameters Θ^* that maximize this criterion

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}}\, \mathcal{L}_M(\Theta)$$

- The direct maximisation of this criterion is impossible because it implies a sum of a logarithm of a sum.

Mixture models (3)

- We use then iterative methods for its maximisation (e.g. the EM algorithm).
- Once the optimal parameters of the mixture are found, each document is then assigned to a group following the Bayesian decision rule:

$$\mathbf{x} \in G_k \Leftrightarrow P(y = k \mid \mathbf{x}, \Theta^*) = \underset{\ell}{\operatorname{argmax}}\, P(y = \ell \mid \mathbf{x}, \Theta^*)$$

where

$$\begin{aligned} \forall \ell \in \{1, \ldots, K\}, P(y = \ell \mid \mathbf{x}, \Theta^*) &= \frac{\pi_\ell^* P(\mathbf{x} \mid y = \ell, \theta_k^*)}{P(\mathbf{x}, \Theta^*)} \\ &\propto \pi_\ell^* P(\mathbf{x} \mid y = \ell, \theta_k^*) \end{aligned}$$

EM algorithm [Dempster et al., 1977]

- The idea behind the algorithm is to introduce hidden random variables Z such that if Z were known, the value of parameters maximizing the likelihood would be simple to be find:

$$\mathcal{L}_M(\Theta) = \ln \sum_Z P(\mathbf{x}_{1:N} \mid Z, \Theta) P(Z \mid \Theta)$$

- by denoting the current estimates of the parameters at time t by $\Theta^{(t)}$, the next iteration $t+1$ consists in finding the new parameters Θ that maximize $\mathcal{L}_M(\Theta) - \mathcal{L}_M(\Theta^{(t)})$

$$\mathcal{L}_M(\Theta) - \mathcal{L}_M(\Theta^{(t)}) = \ln \sum_Z P(Z \mid \mathbf{x}_{1:N}, \Theta^{(t)}) \frac{P(\mathbf{x}_{1:N} \mid Z, \Theta) P(Z \mid \Theta)}{P(Z \mid \mathbf{x}_{1:N}, \Theta^{(t)}) P(\mathbf{x}_{1:N} \mid \Theta^{(t)})}$$

EM algorithm [Dempster et al., 1977]

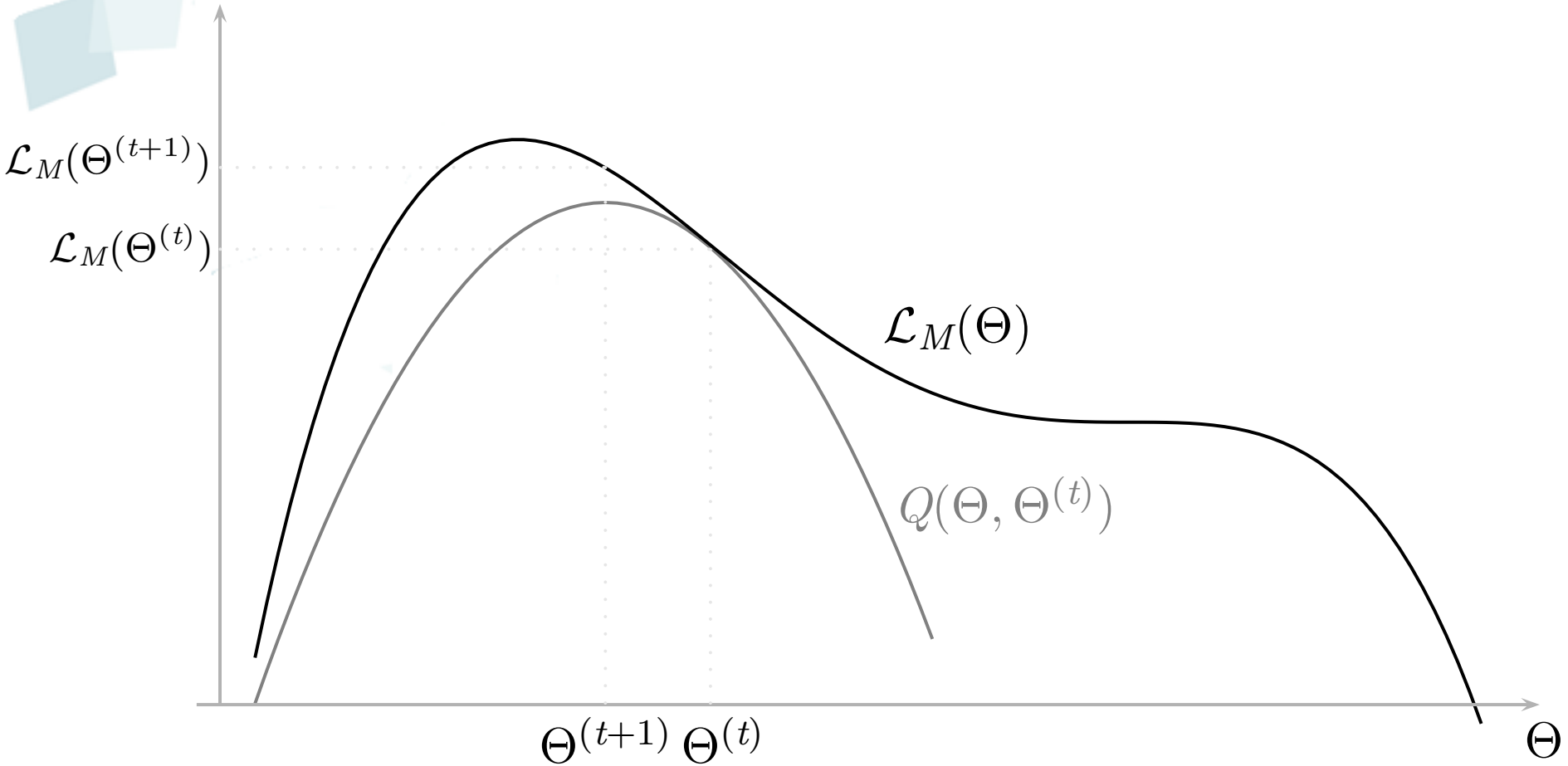
- From the Jensen inequality and the concavity of the logarithm it comes:

$$\mathcal{L}_M(\Theta) - \mathcal{L}_M(\Theta^{(t)}) \geq \sum_Z P(Z \mid \mathbf{x}_{1:N}, \Theta^{(t)}) \ln \frac{P(\mathbf{x}_{1:N} \mid Z, \Theta) P(Z \mid \Theta)}{P(\mathbf{x}_{1:N} \mid \Theta^{(t)}) P(Z \mid \mathbf{x}_{1:N}, \Theta^{(t)})}$$

- Let

$$Q(\Theta, \Theta^{(t)}) = \mathcal{L}_M(\Theta^{(t)}) + \sum_Z P(Z \mid \mathbf{x}_{1:N}, \Theta^{(t)}) \ln \frac{P(\mathbf{x}_{1:N} \mid Z, \Theta) P(Z \mid \Theta)}{P(\mathbf{x}_{1:N} \mid \Theta^{(t)}) P(Z \mid \mathbf{x}_{1:N}, \Theta^{(t)})}$$

EM algorithm [Dempster et al., 1977]



EM algorithm [Dempster et al., 1977]

- At iteration $t+1$, we look for parameters Θ that maximise $Q(\Theta, \Theta^{(t)})$:

$$\Theta^{(t+1)} = \underset{\Theta}{\operatorname{argmax}}\, \mathbb{E}_{Z|\mathbf{d}_{1:N}} \left[\ln P(\mathbf{d}_{1:N}, Z \mid \Theta) \mid \Theta^{(t)} \right]$$

- The EM algorithm is an iterative

Algorithm 1 The EM algorithm

1: Input: A collection $\mathbf{x}_{1:N} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$
2: Initialize randomly the parameters $\Theta^{(0)}$
3: **for each** $t \geq 0$ **do**
4: **E**-step: Estimate $\mathbb{E}_{Z|\mathbf{d}_{1:N}} \left[\ln P(\mathbf{d}_{1:N}, Z \mid \Theta) \mid \Theta^{(t)} \right]$
5: **M**-step: Find new parameters $\Theta^{(t+1)}$ that maximise $Q(\Theta, \Theta^{(t)})$
6: **end for each**

EM algorithm [Dempster et al., 1977]

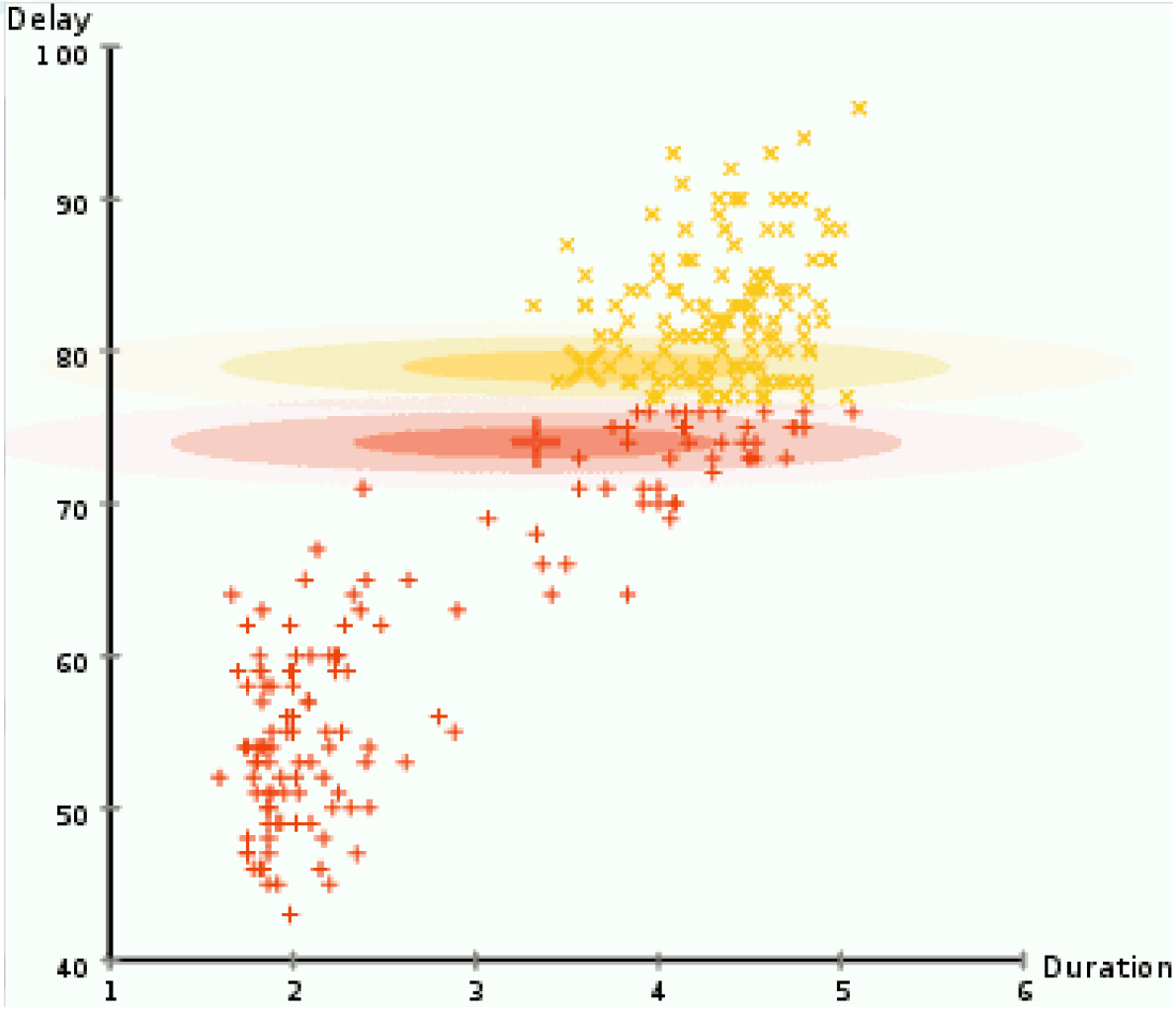


Figure from

https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm

EM algorithm [Dempster et al., 1977]

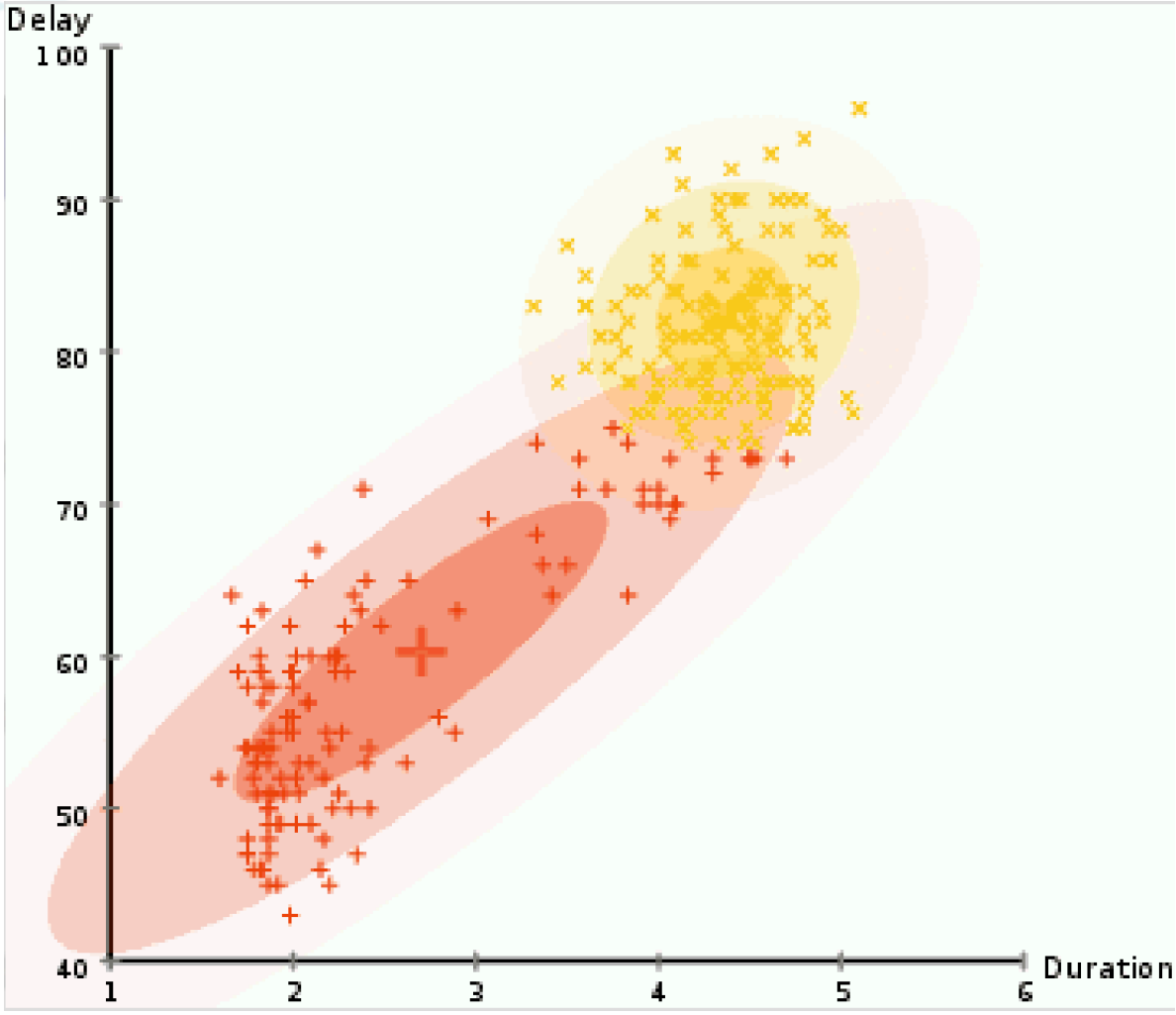


Figure from

https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm

EM algorithm [Dempster et al., 1977]

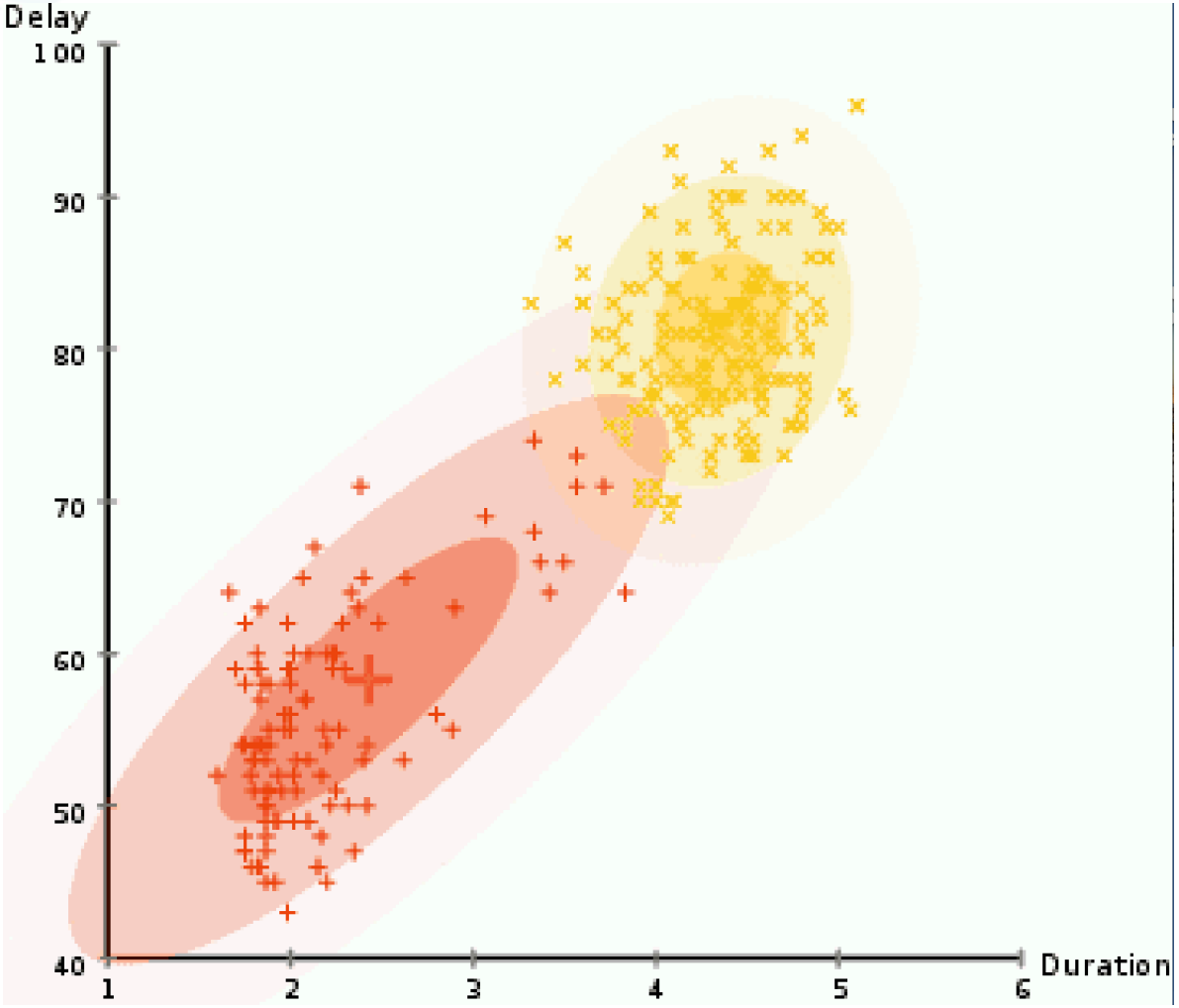


Figure from

https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm

EM algorithm [Dempster et al., 1977]

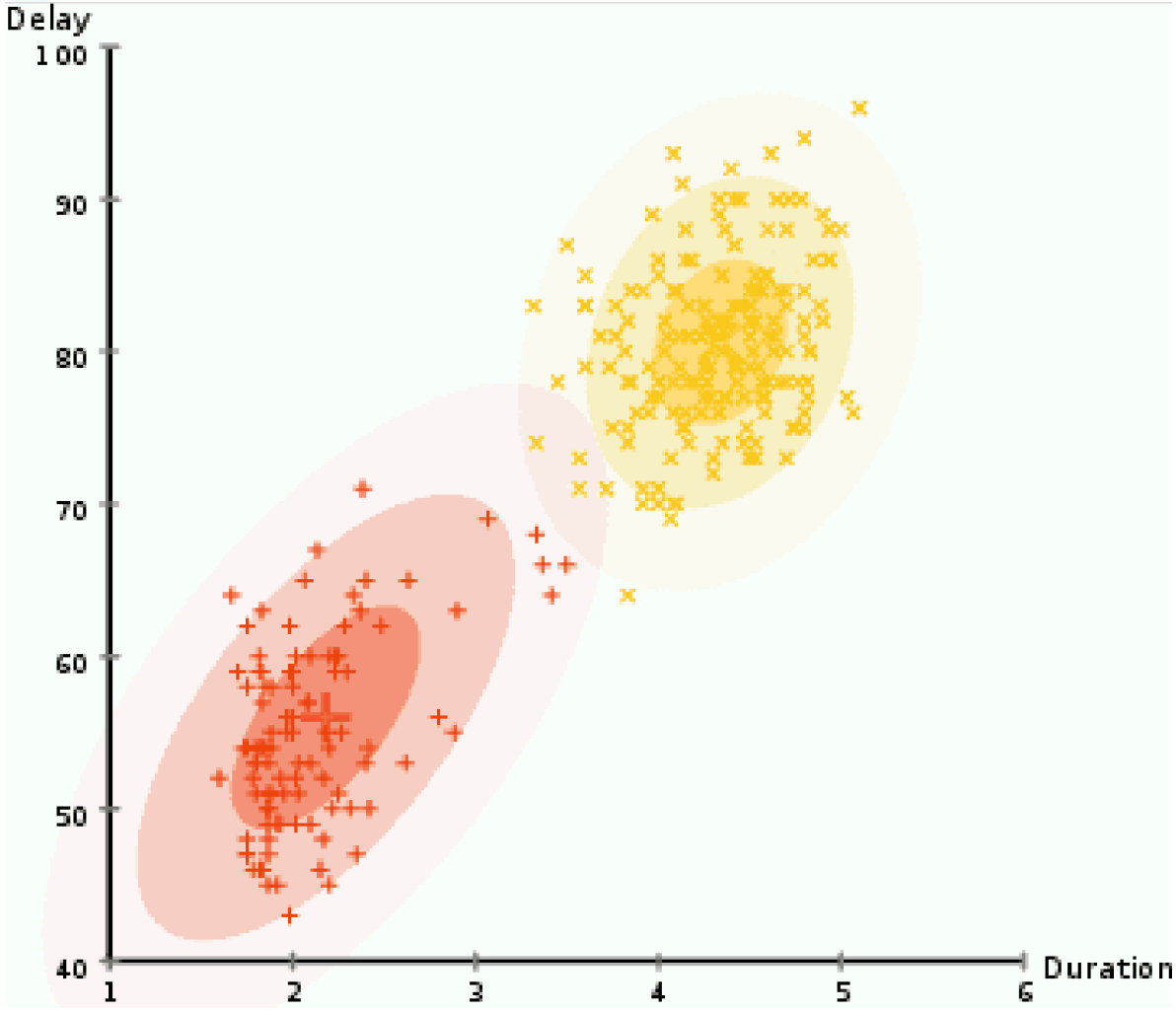


Figure from

https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm

EM algorithm [Dempster et al., 1977]

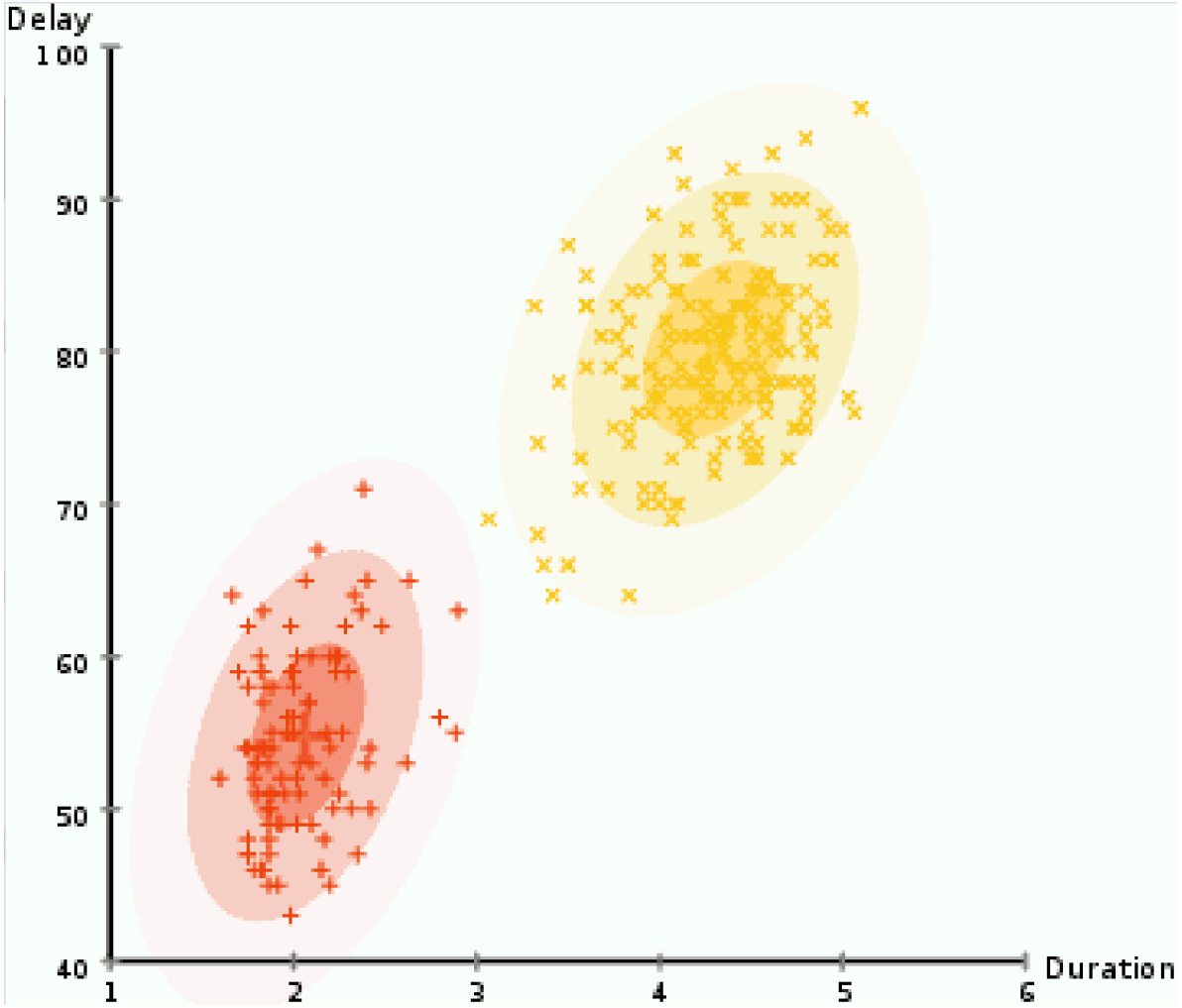


Figure from

https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm

EM algorithm [Dempster et al., 1977]

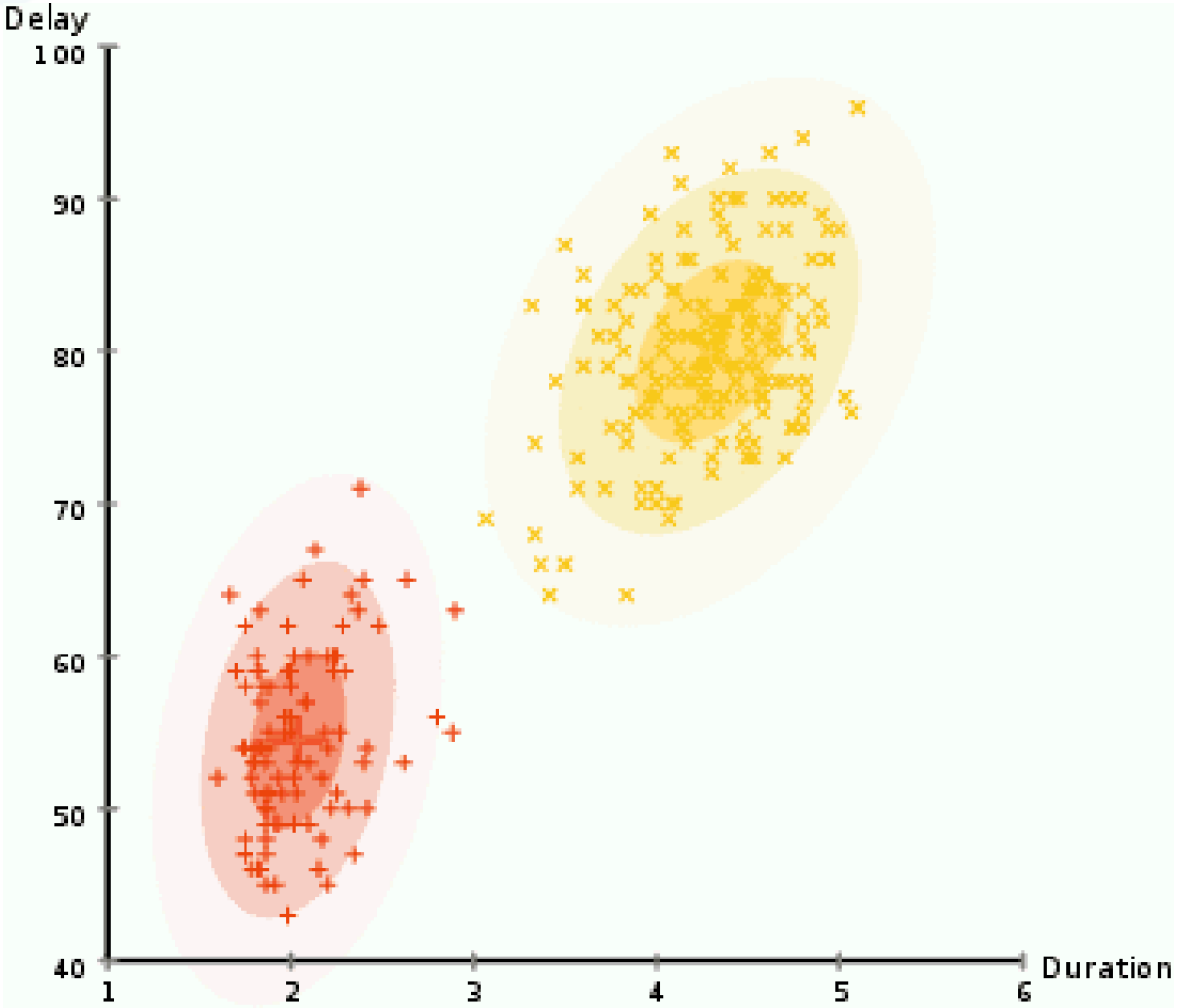


Figure from

https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm

CEM algorithm [Celeux et al. 91]

We suppose that

- Each group $k \in \{1, ..., K\}$ is generated by a distribution of probabilities of parameters θ_k,
- observations are supposed to be identically and independently distributed according to a probability distribution,
- each observation $\mathbf{x}_i \in \mathcal{C}$ belongs to one and only one group, we define a indicator cluster vector $\mathbf{t}_i = (t_{i1}, \ldots, t_{iK})$

$$\mathbf{x}_i \in G_\ell \Leftrightarrow y_i = \ell \Leftrightarrow t_{ik} = \begin{cases} 1, & \text{if } k = \ell, \\ 0, & \text{otherwise.} \end{cases}$$

The aim is to find the parameters $\Theta = \{\theta_k; k \in \{1, \ldots, K\}\}$ qui that maximizes the complete log-likelihood

$$\mathcal{V}(\mathcal{C}, \pi, \Theta, G) = \prod_{i=1}^{N} P(\mathbf{x}_i, y_i = \ell, \theta_k) = \prod_{i=1}^{N} \prod_{k=1}^{K} P(\mathbf{x}_i, y_i = k, \theta_k)^{t_{ik}}$$

Objectif

In general the parameters Θ are those that maximize

$$\begin{aligned}\mathcal{L}(\mathcal{C}, \Theta, G) &= \sum_{i=1}^{N}\sum_{k=1}^{K} t_{ik} \log P(\mathbf{x}_i, y_i = k, \theta_k) \\ &= \sum_{i=1}^{N}\sum_{k=1}^{K} t_{ik} \log \underbrace{P(y_i = k)}_{\pi_k} P(\mathbf{x}_i \mid y_i = k, \theta_k)\end{aligned}$$

The maximization can be carried out using the classification

EM (CEM) algorithm.

CEM algorithm [Celeux et al. 91]

Begin with an initial partition $G^{(0)}$.

$t \leftarrow 0$

while $\mathcal{L}(\mathcal{C}, \Theta^{(t+1)}, G^{(t+1)}) - \mathcal{L}(\mathcal{C}, \Theta^{(t)}, G^{(t)}) > \epsilon$ **do**

E-step Estimate the posterior probabilities using the current parameters $\Theta^{(t)}$:

$$\forall \ell = \{1, \ldots, K\} \mathbb{E}[t_{i\ell} \mid \mathbf{x}_i, G^{(t)}, \Theta^{(t)}] = \frac{\pi_\ell^{(t)} P(\mathbf{x}_i \mid G_\ell^{(t)}, \theta_\ell^{(t)})}{\sum_{k=1}^{K} \pi_k^{(t)} P(\mathbf{x}_i \mid G_k^{(t)}, \theta_k^{(t)})}$$

C-step Assign to each example $\mathbf{x}_i$ its partition, the one for which the posterior probability is maximum. Note $G^{(t+1)}$ this new partition

M-step Estimate the new parameters $\Theta^{(t+1)}$ qui maximisent $\mathcal{L}(\mathcal{C}, \Theta^{(t)}, G^{(t+1)})$

$t \leftarrow t + 1$

end while

CEM algorithm (convergence)

The algorithm converges to a local maxima of the completer log-likelihood.

- At the **C**-step we choose the new partition $G^{(t+1)}$ using the current set of parameters $\Theta^{(t)}$, according the Bayesian decision rule

$$\mathcal{L}(\mathcal{C}, \Theta^{(t)}, G^{(t+1)}) \geq \mathcal{L}(\mathcal{C}, \Theta^{(t)}, G^{(t)})$$

- At the **M**-step new parameters are found $\Theta^{(t+1)}$ by maximising $\mathcal{L}(\mathcal{C}, \Theta^{(t)}, G^{(t+1)})$:

$$\mathcal{L}(\mathcal{C}, \Theta^{(t+1)}, G^{(t+1)}) \geq \mathcal{L}(\mathcal{C}, \Theta^{(t)}, G^{(t+1)})$$

- At each iteration t we have :

$$\mathcal{L}(\mathcal{C}, \Theta^{(t+1)}, G^{(t+1)}) \geq \mathcal{L}(\mathcal{C}, \Theta^{(t)}, G^{(t)})$$

As there is a finite number of partitions, the iterations between these two steps is guaranteed to converge.

Study case: document clustering

- Documents are usually represented using the Vector Space Model (VSM) proposed By Salton;
  - In this case, the feature characteristics of a document translate the presence of the terms of the vocabulary $\mathcal{V} = (t_1, \ldots, t_V)$ in that document.
  - If these features are based on term frequencies, a document d is then represented by a vector of dimension V:

$$\mathbf{d} = (\mathrm{tf}_{1,d}, \ldots, \mathrm{tf}_{V,d})$$

- In the case where, the presence of the terms in a document is supposed to be independent one from another. The probability distributions are Multinomials

$$\forall \ell \in \{1, \ldots, K\}, P(\mathbf{d} \mid y = \ell) = \frac{\mathrm{tf}_d!}{\mathrm{tf}_{1,d}! \ldots \mathrm{tf}_{V,d}!} \prod_{j=1}^{V} \theta_{j|\ell}^{\mathrm{tf}_{j,d}}$$

where, $tf_d = \mathrm{tf}_{1,d} + \ldots + \mathrm{tf}_{V,d}$

Study case: document clustering

- The parameters of the Mixture model are then

$$\Theta = \left\{\theta_{j|k}; j \in \{1, \ldots, V\}, k \in \{1, \ldots, K\}; \pi_k; j \in \{1, \ldots, K\}\right\}$$

- By neglecting the multinomial terms, the optimization of the complete log-likelihood over a document collection of N documents $\mathcal{C} = \{d_1, \ldots, d_N\}$ writes

$$\max_{\Theta} \sum_{i=1}^{N} \sum_{k=1}^{K} t_{ik} \left(\ln \pi_k + \sum_{j=1}^{V} \mathrm{tf}_{j,d} \ln \theta_{j|k} \right)$$

$$u.c. \sum_{k=1}^{K} \pi_k = 1, \forall k, \sum_{j=1}^{V} \theta_{j|k} = 1$$

Study case: document clustering

- The maximization of the complete log-likelihood with respect to model parameters is then carried out by these esitmates

$$\forall j, \forall k, \theta_{j|k} = \frac{\sum_{i=1}^{N} t_{ik} \mathrm{tf}_{j,d_i}}{\sum_{j=1}^{V} \sum_{i=1}^{N} t_{ik} \mathrm{tf}_{j,d_i}}$$

$$\forall k, \pi_k = \frac{\sum_{i=1}^{N} t_{ik}}{N}$$

Evaluation

- The results of clustering can be evaluated using a labeled training set.
- The two common measures are *purity* and *Normalised Mutual Information*.
- The purity measure tends to quantify the ability of the clustering method to regroupe the observations of the same class into the same partitions. Let G be the partition found and C the set of classes found over G. The purity measure is then defined by:

$$\text{pure}(G, C) = \frac{1}{N} \sum_{k} \max_{l} |G_k \cap C_l|$$

Evaluation

- The Normalised Mutual Information is defined by:

$$\mathrm{IMN}(G, C) = \frac{2 \times I(G, C)}{H(G) + H(C)}$$

where I is the mutual information and H the entropy. These two quantities can be computed as:

$$\begin{aligned} I(G, C) &= \sum_k \sum_l P(G_k \cap C_l) \log \frac{P(G_k \cap C_l)}{P(G_k) P(C_l)} \\ &= \sum_k \sum_l \frac{|G_k \cap C_l|}{N} \log \frac{N |G_k \cap C_l|}{|G_k||C_l|} \end{aligned}$$

and:

$$\begin{aligned} H(G) &= -\sum_k P(G_k) \log P(G_k) \\ &= -\sum_k \frac{|G_k|}{N} \log \frac{|G_k|}{N} \end{aligned} \tag{1}$$

NMI is equal to 1 if the two sets G and C are identical

Semi-supervised Learning

Semi-supervised Learning

- International Data Corporation (IDC) projects that in 2020 there will be 40 zettabytes (ZB) (40 trillion GB) of digital data available i.e. *as many digital bits as there are stars in the universe*[1]

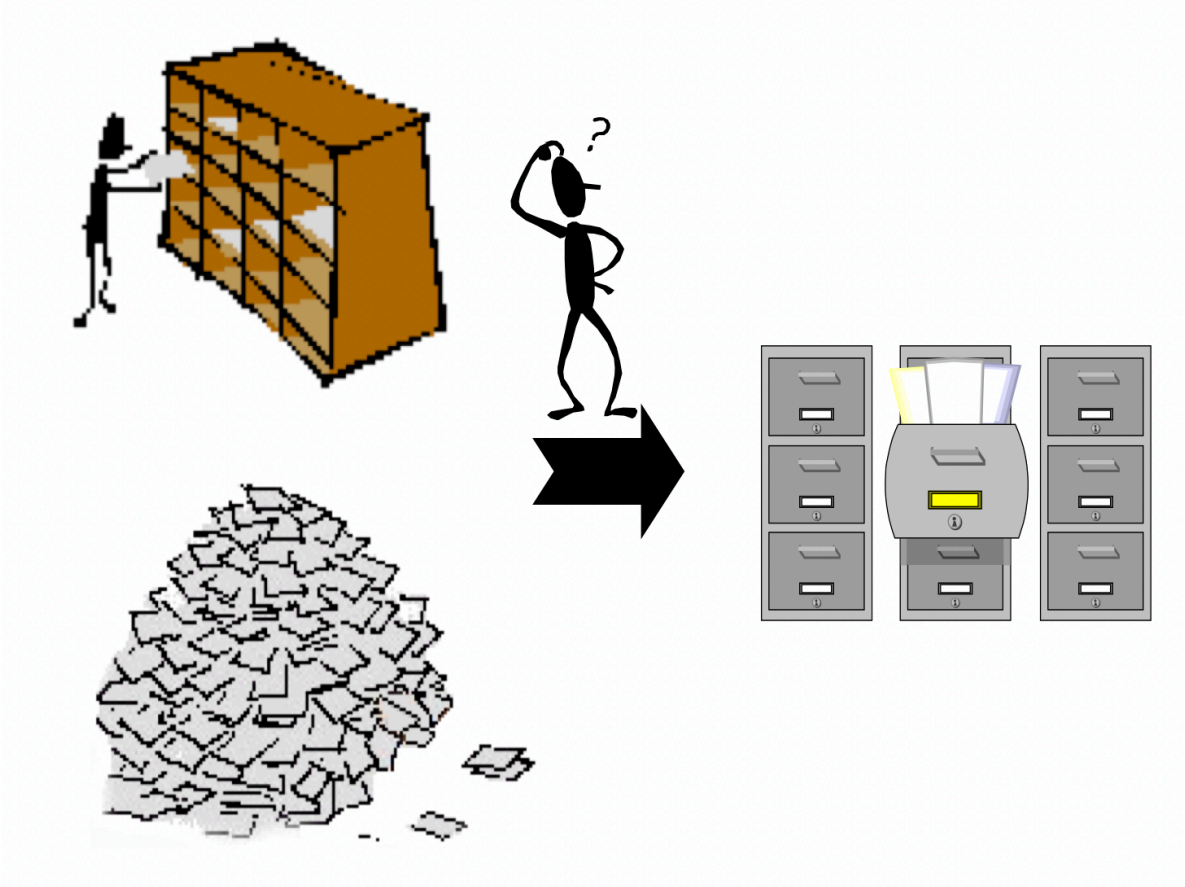
- The labeling of these data becomes prohibitive and unrealistic in many cases,

- The question is then how to learn prediction functions as in supervised learning?

[1] `http://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm`

Semi-supervised Learning

- Semi-supervised learning techniques aim at enhancing supervised models, by respecting the structure of unlabeled data [Amini, 2015].



Formally

We consider an input space $\mathcal{X} \subseteq \mathbb{R}^d$ and an output space $\mathcal{Y}$.

We suppose to have m pairs of examples $\mathcal{S} = \{(\mathbf{x}_i, y_i), i \in \{1, \ldots, m\}\}$ generated `i.i.d` from a probability distribution $\mathcal{D}$; along with u observations $\mathcal{U} = \{\mathbf{x}_i; i \in \{m+1, \ldots, m+u\}\}$ also generated `i.i.d` from a marginal $\mathcal{D}_{\mathcal{X}}$, where generally $u >> m$.

Aim: Construct a prediction function $f : \mathcal{X} \to \mathcal{Y}$ which predicts an output y for a given new $\mathbf{x}$ with a minimum probability of error.

Hypotheses

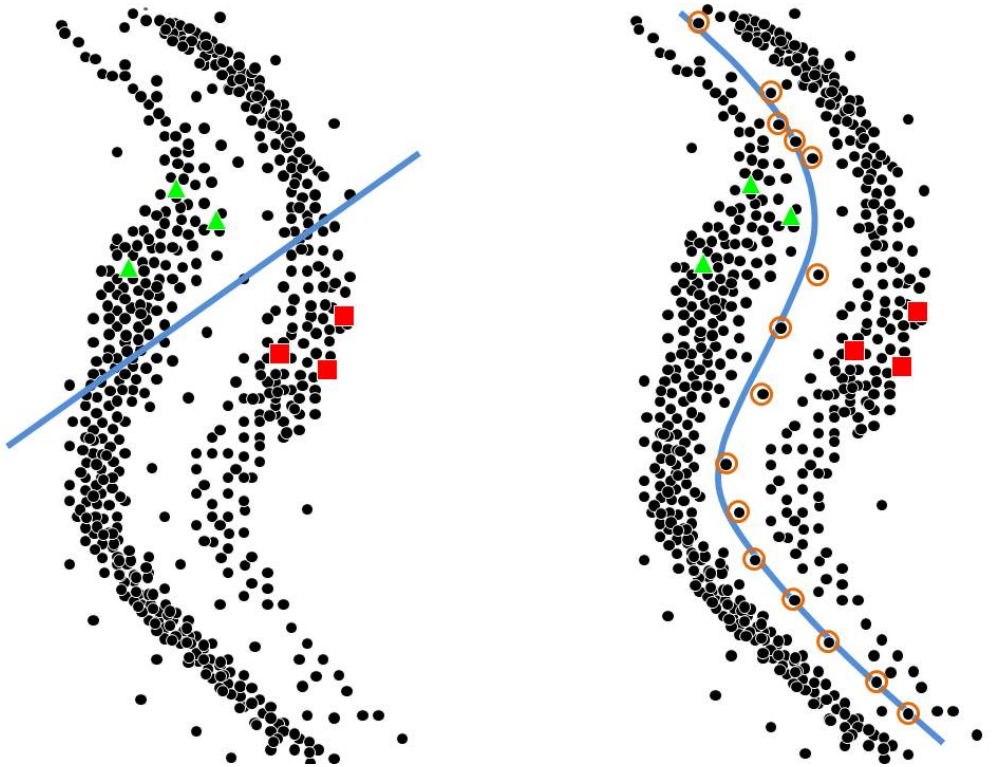
- **smoothness** assumption: If two observations $\mathbf{x}_1$ and $\mathbf{x}_2$ in a high-density region are close, then their corresponding outputs y_1 and y_2 should be close as well.

Hypotheses

- **smoothness** assumption: If two observations $\mathbf{x}_1$ and $\mathbf{x}_2$ in a high-density region are close, then their corresponding outputs y_1 and y_2 should be close as well.
  - This assumption gave place to a more powerful assumption called **Low density separation**, stipulating that the decision boundary should lie in a low-density region.

Hypotheses

- **smoothness** assumption: If two observations $\mathbf{x}_1$ and $\mathbf{x}_2$ in a high-density region are close, then their corresponding outputs y_1 and y_2 should be close as well.
  - This assumption gave place to a more powerful assumption called **Low density separation**, stipulating that the decision boundary should lie in a low-density region.

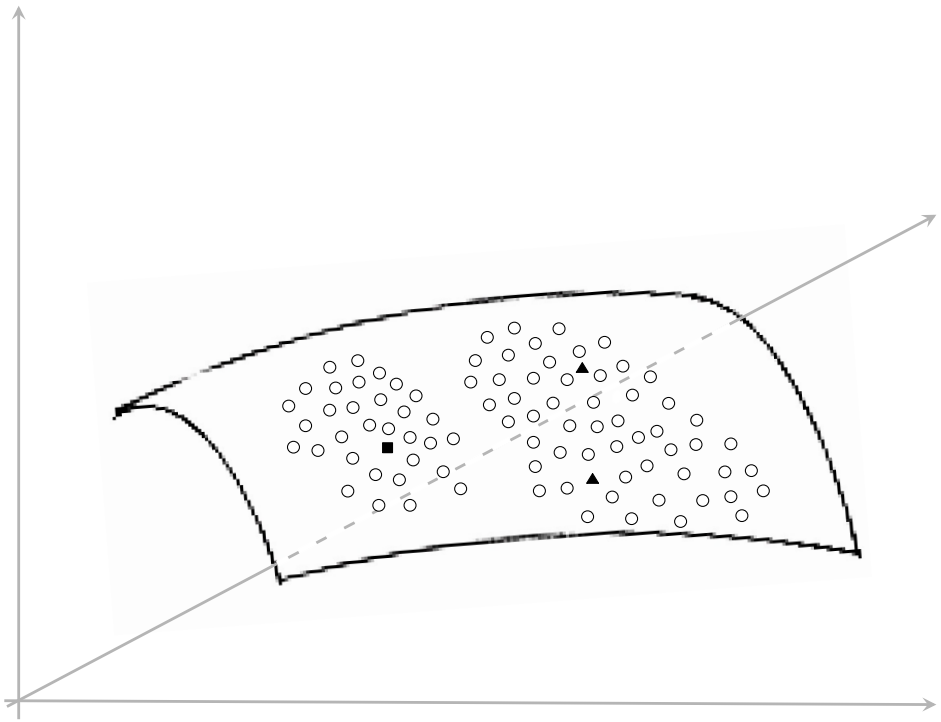


Hypotheses

- **Manifold** assumption: Observations are roughly contained in a low dimensional manifold.
  - The assumption aims to avoid the curse of dimensionality, as it assumes that learning can be performed in a more meaningful low-dimensional space.

Hypotheses

- **Manifold** assumption: Observations are roughly contained in a low dimensional manifold.
  - The assumption aims to avoid the curse of dimensionality, as it assumes that learning can be performed in a more meaningful low-dimensional space.



Generative Approaches

- Under the generative approach, it is assumed that each observation $\mathbf{x}$ is drawn from a mixture of K groups or classes in proportions $\pi_1, ..., \pi_c$, respectively, where

$$\sum_{k=1}^{c} \pi_k = 1 \text{ and } \forall k, \pi_k \geq 0$$

  Further the classes of the labeled examples are known.
- For each labeled example $(\mathbf{x}_i, y_i)$ in $\mathcal{S}$, let $t_i = \{t_{i\ell}\}_\ell$ be the indicator vector class associated to $\mathbf{x}_i$.

$$\forall i \in \mathcal{S}, \forall k, y_i = k \Leftrightarrow t_{ik} = 1 \text{ and } \forall \ell \neq k, t_{i\ell} = 0$$

- During training, unlabeled samples will be given tentative labels. Let $\widetilde{y}$ and $\widetilde{t}$ denote respectively the class label and the class indicator vector of an unlabeled observation $\mathbf{x}$ estimated with a learning system.

Generative Approaches

- Generative models are designed under the smoothness assumption and there are two main approaches : maximum likelihood (ML) and classification maximum likelihood (CML). For both approaches, observations are supposed to be generated via a mixture density:

$$P(x, \Theta) = \sum_{k=1}^{K} \pi_k P(\mathbf{x} \mid y = k, \theta_k)$$

- [McLachlan, 1992] has extended CML and CEM for generative algorithms to the case where both labeled and unlabeled data are used for learning.
- In this context, the indicator vector class for labeled data are known whereas they are estimated for unlabeled data, and the CML writes

$$\mathcal{L}_c(C, \Theta, G) = \sum_{i=1}^{m} \sum_{k=1}^{K} t_{ik} \log P(x_i, y = k, \Theta) + \sum_{i=m+1}^{m+u} \sum_{k=1}^{K} \widetilde{t}_{ik} \log P(x_i, \widetilde{y} = k, \Theta)$$

Semi-supervised CEM

The density probabilities $P(\mathbf{x} \mid y = k, \theta_k^{(0)})$ are respectively estimated on the K classes from the labeled data $\mathcal{S}$, and $C^{(0)}$ is defined accordingly.

while $\mathcal{L}_c(C, \Theta^{(t+1)}, G^{(t+1)}) - \mathcal{L}(C, \Theta^{(t)}, G^{(t)}) > \epsilon$ **do**

E-step: Estimate the posterior class probability that each unlabeled example $\mathbf{x}_i$ belongs to $C_k^{(j)}$:

$$\forall \mathbf{x}_i \in \mathcal{U}, \forall k, \mathbb{E}[\tilde{t}_{ik}^{(j)} \mid \mathbf{x}_i; C^{(j)}, \Theta^{(j)}] = \frac{\pi_k^{(j)} P(\mathbf{x}_i \mid y = k, \theta_k^{(j)})}{P(\mathbf{x}, \Theta^{(j)})}$$

C-step: Assign each $\mathbf{x}_i \in \mathcal{U}$ to the cluster $C_k^{(j+1)}$ with maximal posterior probability according to $\mathbb{E}[\tilde{t} \mid \mathbf{x}]$. Let $C^{(j+1)}$ be the new partition.

M-step: Estimate the new parameters $\Theta^{(j+1)}$ which maximize $L_c(C^{(j+1)}, \Theta^{(j)})$ for semi-supervised learning

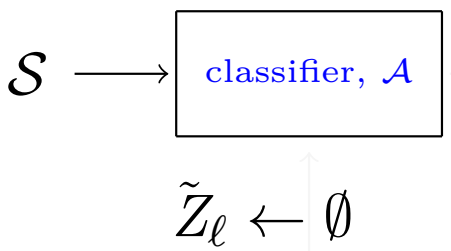
end while

Discriminant Approaches - Self-Training

- Discriminant models based on the low density separation assumption, and the most popular one is the self-training algorithm.

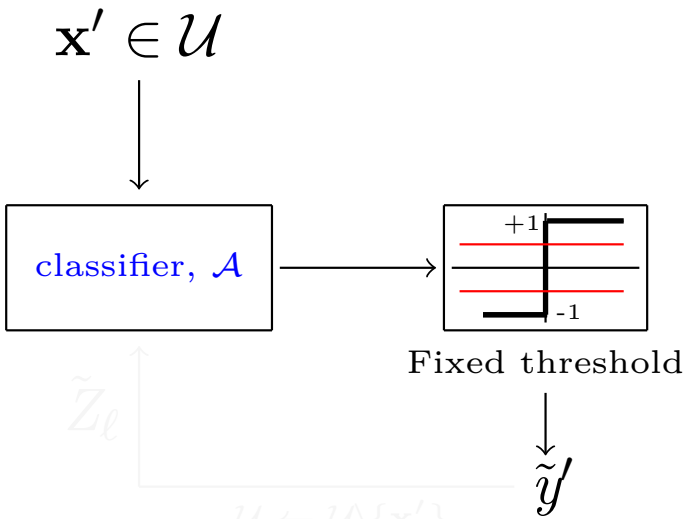
Discriminant Approaches - Self-Training

- Discriminant models based on the low density separation assumption, and the most popular one is the self-training algorithm.



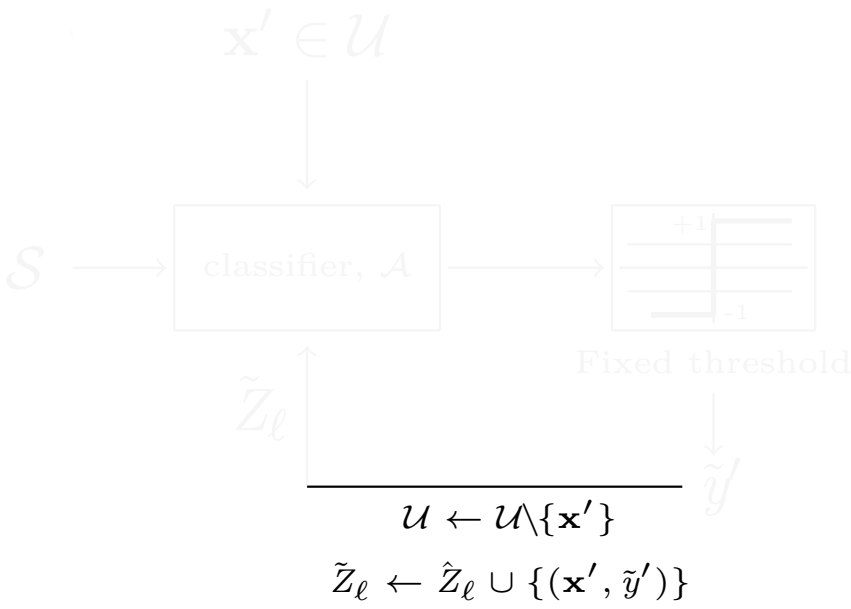
Discriminant Approaches - Self-Training

- Discriminant models based on the low density separation assumption, and the most popular one is the self-training algorithm.



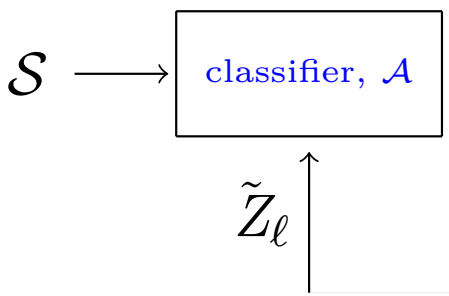
Discriminant Approaches - Self-Training

- Discriminant models based on the low density separation assumption, and the most popular one is the self-training algorithm.



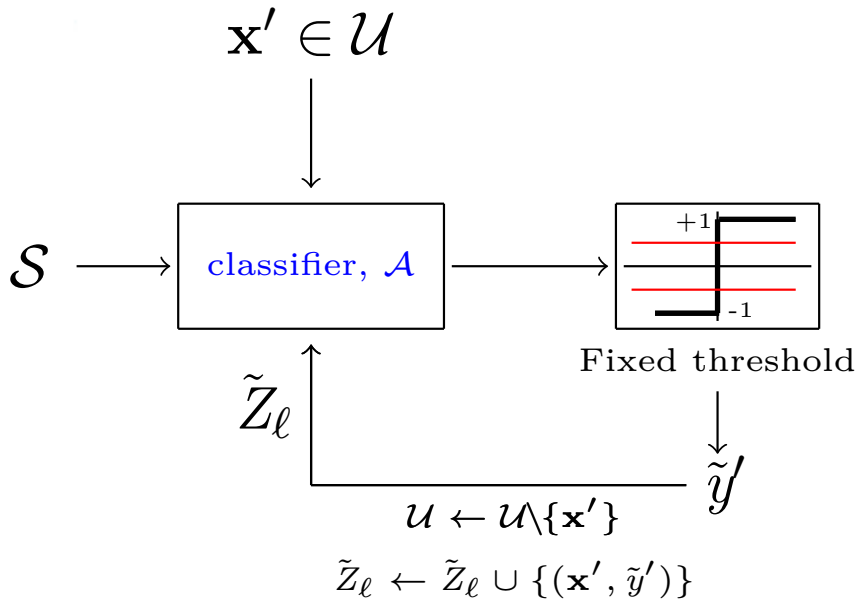
Discriminant Approaches - Self-Training

- Discriminant models based on the low density separation assumption, and the most popular one is the self-training algorithm.



Discriminant Approaches - Self-Training

- Discriminant models based on the low density separation assumption, and the most popular one is the self-training algorithm.



- SLA is a discriminant instance of the CEM algorithm.

References

M.-R. Amini, Nicolas Usunier
Learning with partially labeled and interdependent data.
Springer
2015.

G. Celeux, G. Govaert.
A classification EM algorithm for clustering and two stochastic versions.
Research Report-1364, INRIA, 1991.

A.P. Dempster, N.M. Laird, D.B. Rubin (1977).
Maximum Likelihood from Incomplete Data via the EM Algorithm.
Journal of the Royal Statistical Society, Series B. 39, (1): 1–38, 1977

T. Hofmann
Probabilistic Latent Semantic Indexing,
Proceedings of the 22Nd Annual International ACM SIGIR Conference,
pp. 50-57, 1999.

References

G.J. McLachlan
Discriminant Analysis and Statistical Pattern Recognition.
Wiley Interscience,
1992.

J. B. MacQueen
Some Methods for classification and Analysis of Multivariate Observations,
Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, (1): 281–297,
1967