MTAT.03.227 Machine Learning

Expectation-maximisation algorithm

Sven Laur
University of Tartu

Quick recap of hard clustering

Model. A hard-clustering algorithm is based on a probabilistic model

$$\mathrm{p}[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n | \boldsymbol{z}, \boldsymbol{\Theta}]$$

where $\boldsymbol{z}$ denotes *unknown* labels and $\boldsymbol{\Theta}$ captures all model parameters.

Optimisation task. The aim of hard-clustering algorithm is to minimise

$$F(\boldsymbol{z}, \boldsymbol{\Theta}) = -\log \mathrm{p}[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n | \boldsymbol{z}, \boldsymbol{\Theta}] \ .$$

If all data points are independent form each other then we can simplify

$$F(\boldsymbol{z}, \boldsymbol{\Theta}) = -\sum_{i=1}^{n} \log \mathrm{p}[\boldsymbol{x}_i | z_i, \boldsymbol{\Theta}_{z_i}]$$

where $\boldsymbol{\Theta}_{z_i}$ denotes parameters for the z_ith data source.

Two-step minimisation algorithm

M1 step. Find a new labelling $\boldsymbol{z}$ that minimises $F(\boldsymbol{z}, \boldsymbol{\Theta})$ for fixed $\boldsymbol{\Theta}$. Due the form of $F(\boldsymbol{z}, \boldsymbol{\Theta})$ the label can be sought for each data point separately:

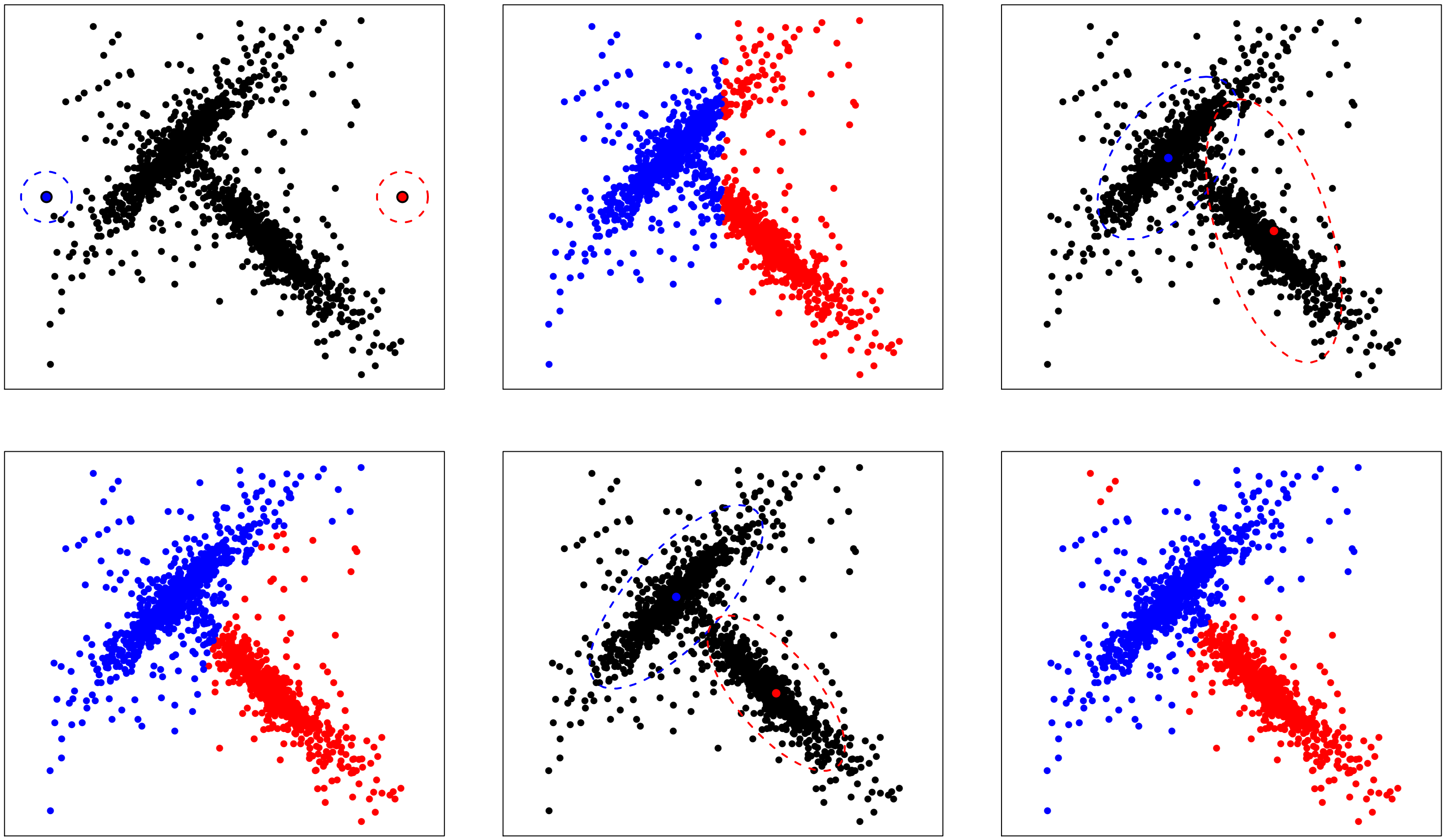
$$z_i = \operatorname*{argmax}_{z \in \{1,\ldots,k\}} \mathrm{p}[\boldsymbol{x}_i | z_i = z, \boldsymbol{\Theta}_z] \ .$$

M2 step. Find parameters $\boldsymbol{\Theta}$ that minimise $F(\boldsymbol{z}, \boldsymbol{\Theta})$ for fixed $\boldsymbol{z}$. Again, parameters can separately be sought for each data source:

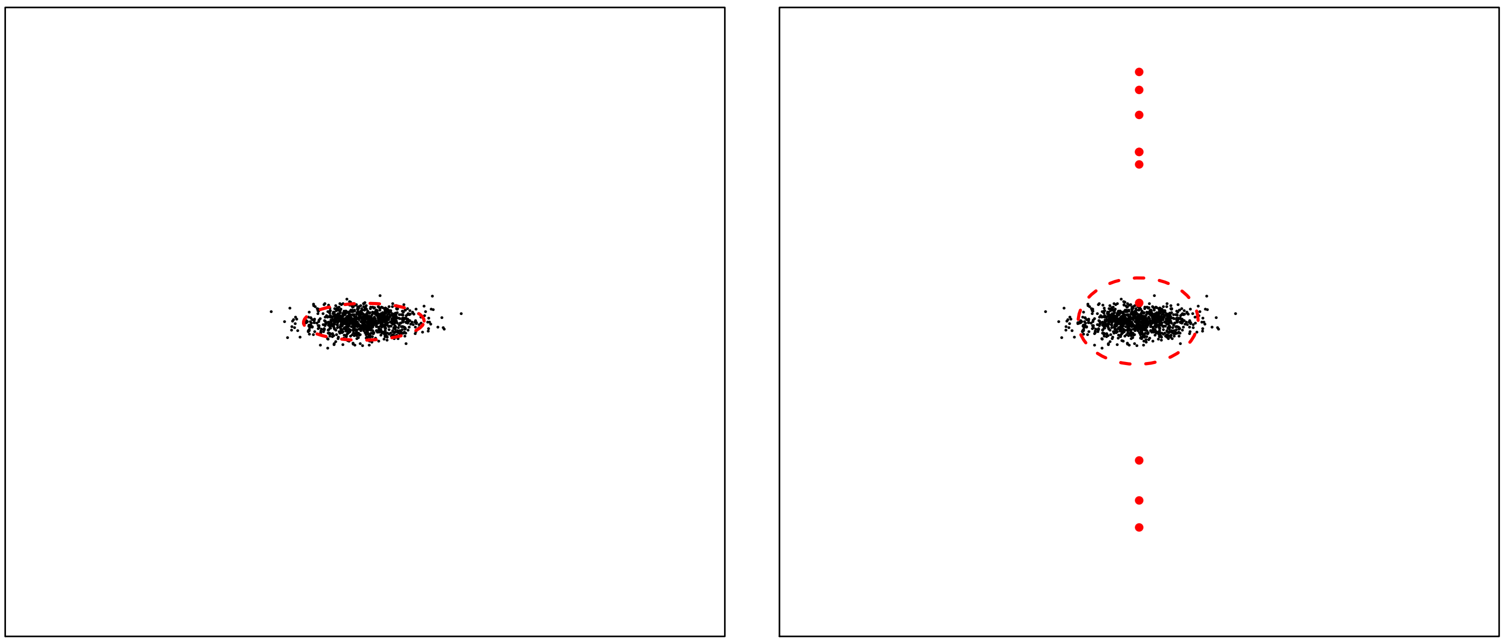
$$\boldsymbol{\Theta}_j^* = \operatorname*{argmax}_{\boldsymbol{\Theta}_j} \sum_{i \in \mathcal{I}_j} \log \mathrm{p}[\boldsymbol{x}_i | z_i = j, \boldsymbol{\Theta}_j]$$

where $\mathcal{I}_j = \{i : z_i = j\}$ denotes elements coming form the jth data source.

Illustrative example



Hard clustering is brittle against outliers



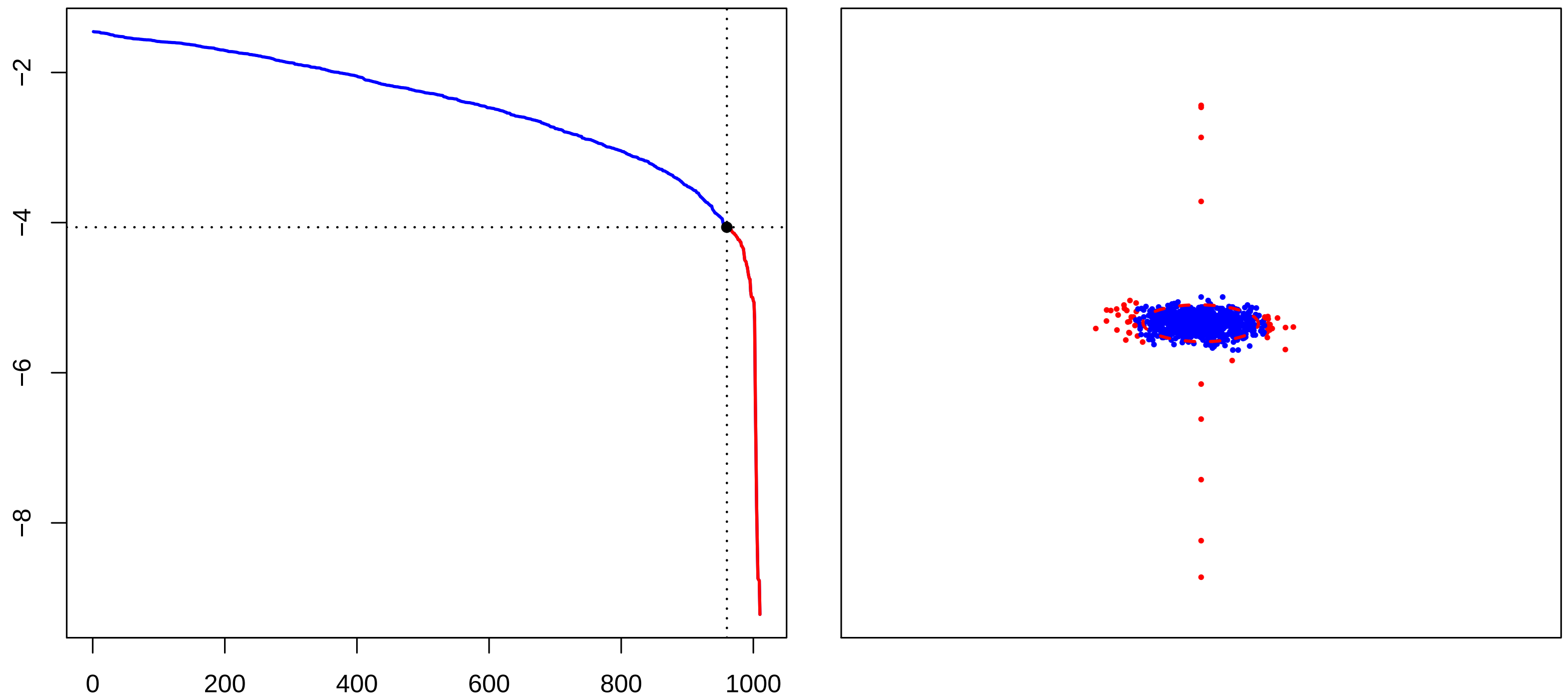
A few data points not belonging to the cluster can completely offset the shape estimation procedure. We need to reduce the impact of outliers.

Quick fix

Assuming that current parameter estimates are not far off, we can make parameter fitting more robust by throwing out improbable points.

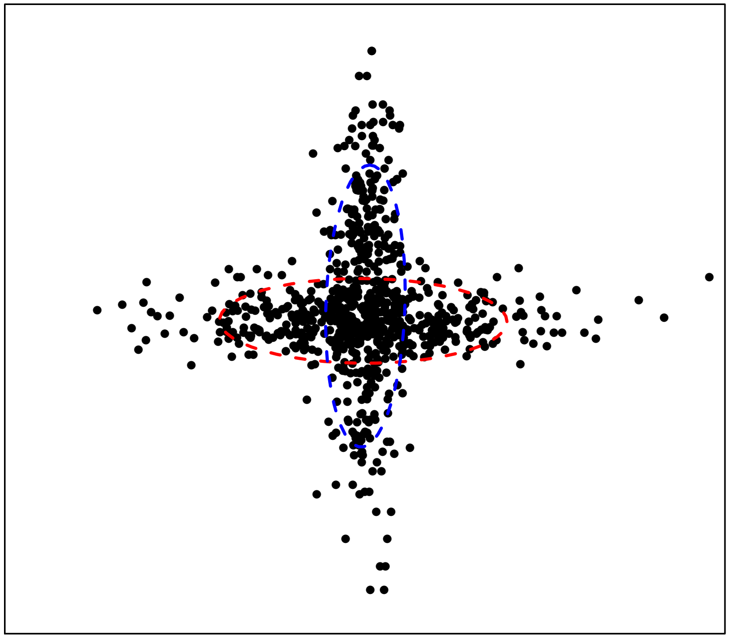
- ▷ Compute likelihood $\mathrm{p}[\boldsymbol{x}_i|\boldsymbol{\Theta}_j]$ for each data point $\boldsymbol{x}_i$ in the cluster
- ▷ Throw out 5-10% cluster points with lowest likelihood scores
- ▷ Model parameters based on the reduced dataset

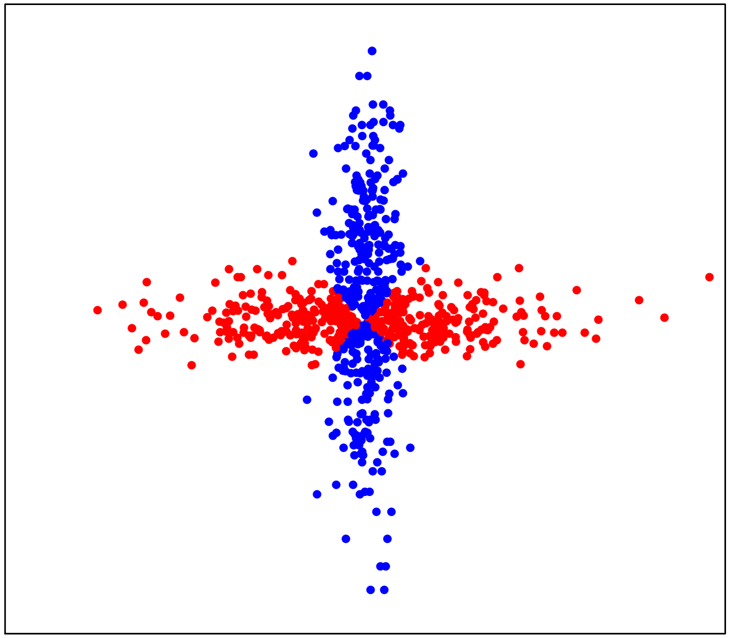
Illustrative example



Left pane shows ordered log likelihood of points and corresponding cut-off point. The right pane shows how the outlier elimination alter parameters.

Choice of labels can be ambiguous





Sometimes the likelihoods for different sources are almost equal

- ▷ Label assignment is almost arbitrary
- ▷ Data points that belong to the cluster are not counted

Fractional weights as an alternative

Data points should have different weighs based on the plausibility

- ▷ Potential outliers should have low weights to limit their impact
- ▷ Ambiguous points should impact the parameters of both clusters
- ▷ We should not prefer some data points to others in the algorithm

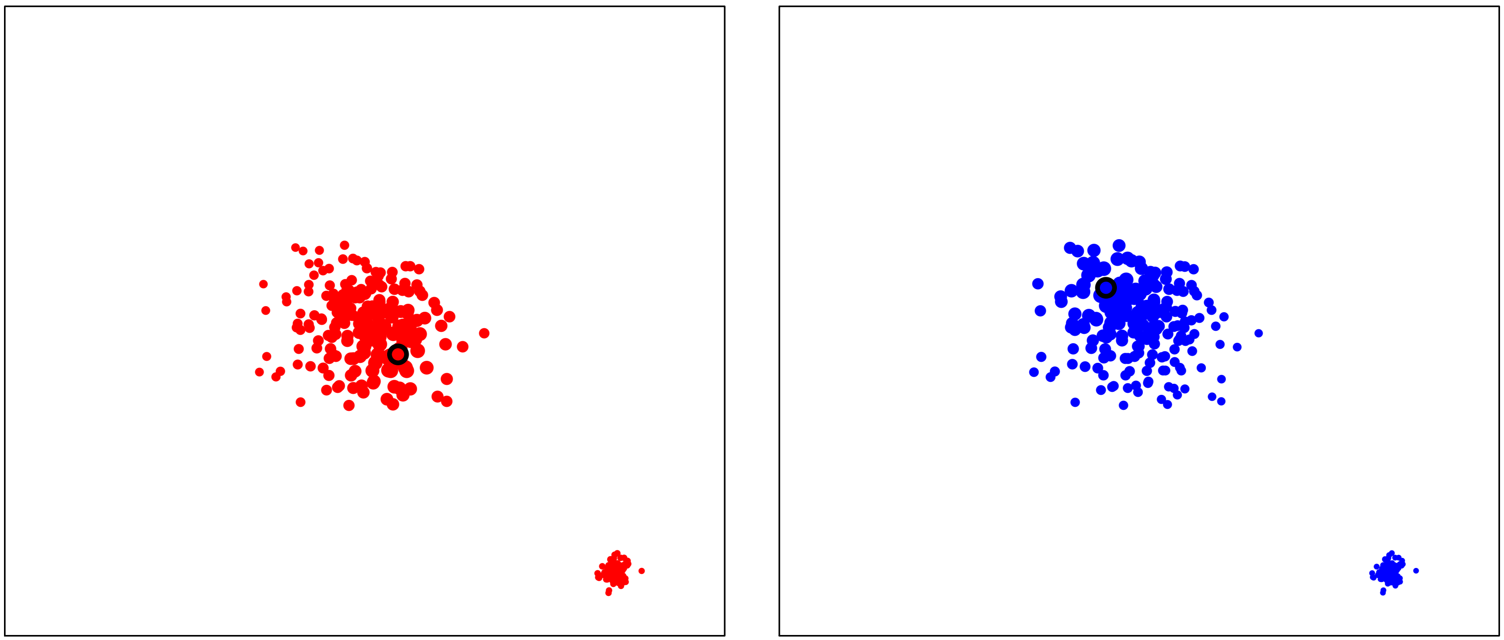
First try. The most obvious choice for the weights are likelihoods

$$w_{ij} = \mathrm{p}[\boldsymbol{x}_i | z_i = j, \boldsymbol{\Theta}_j]$$

but some data points have very low likelihoods for all clusters

- ▷ These data points would be largely ignored by the algorithm

The weighting scheme hides data



The overall contribution of cluster points in south-east direction is small for both clusters and thus the cluster is never found by the the algorithm.

Fractional weights as an alternative

Data points should have different weighs based on the plausibility

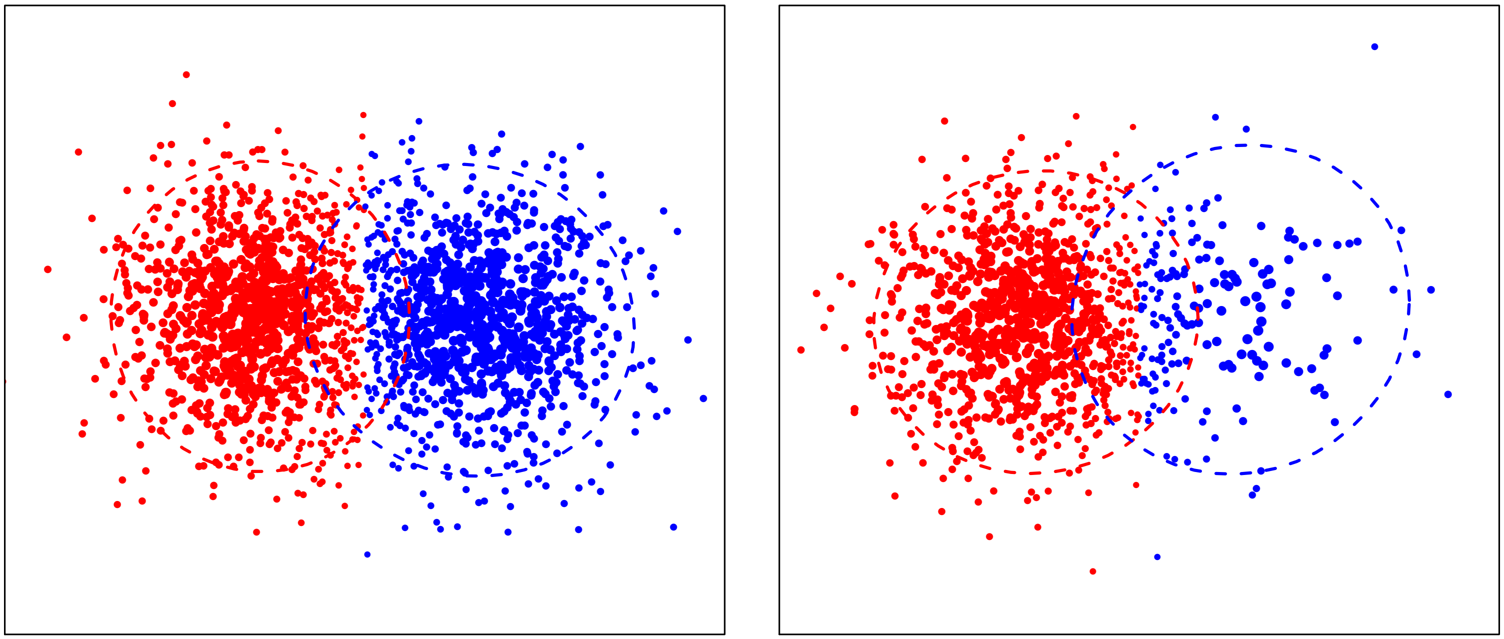
- ▷ Potential outliers should have low weights to limit their impact
- ▷ Ambiguous points should impact the parameters of both clusters
- ▷ We should not prefer some data points to others in the algorithm

Second try. We can normalise the weights so that the sum up to one

$$w_{ij} = \frac{\mathrm{p}[\boldsymbol{x}_i|z_i = j, \boldsymbol{\Theta}_j]}{\mathrm{p}[\boldsymbol{x}_i|z_i = 1, \boldsymbol{\Theta}_1] + \cdots + \mathrm{p}[\boldsymbol{x}_i|z_i = k, \boldsymbol{\Theta}_k]}$$

This weighting does not work if getting data points from some cluster is much more probable than from the other clusters.

The weighting scheme ignores frequency



If the elements from one data source are rare then the elements form the ambiguous area are more likely to originate from the abundant source.

Fractional weights as an alternative

Data points should have different weighs based on the plausibility

- ▷ Potential outliers should have low weights to limit their impact
- ▷ Ambiguous points should impact the parameters of both clusters
- ▷ We should not prefer some data points to others in the algorithm

Standard solution. We must take mixture proportions into account

$$w_{ij} = \frac{\lambda_j \cdot \mathrm{p}[\boldsymbol{x}_i | z_i = j, \boldsymbol{\Theta}_j]}{\lambda_1 \cdot \mathrm{p}[\boldsymbol{x}_i | z_i = 1, \boldsymbol{\Theta}_1] + \cdots + \lambda_k \cdot \mathrm{p}[\boldsymbol{x}_i | z_i = k, \boldsymbol{\Theta}_k]}$$

$$= \frac{\mathrm{p}[\boldsymbol{x}_i, z_i = j | \boldsymbol{\Theta}]}{\mathrm{p}[\boldsymbol{x}_i | \boldsymbol{\Theta}]} = \mathrm{p}[z_i = j | \boldsymbol{x}_i, \boldsymbol{\Theta}]$$

and thus the weight w_j is equal to label probability given $\boldsymbol{x}_i$ and $\boldsymbol{\Theta}$.

How to handle fractional weights?

Naive implementation of fractional weights

Under the independence assumption hard-clustering minimises

$$F(\boldsymbol{z}, \boldsymbol{\Theta}) = -\sum_{i=1}^{n} \log \mathrm{p}[\boldsymbol{x}_i | z_i, \boldsymbol{\Theta}_{z_i}]$$

To implement weights with precision $\frac{1}{\ell}$ we duplicate each data point ℓ times and modify label assignment step **M1**. The step **M2** remains same.

E1 step

- ▷ Find weights for each $w_{ij} = \Pr[z_i = j | \boldsymbol{x}_i, \boldsymbol{\Theta}]$
- ▷ Compute integer counts $c_{ij} = \lfloor w_{ij} \cdot \ell \rceil$ so that they add up to ℓ.
- ▷ For each data point $\boldsymbol{x}_i$ assign c_{ij} copies to the cluster j.

Implementation without data duplication

E1 step. Compute fractional weights $w_{ij} = \mathrm{p}[z_i = j|\boldsymbol{x}_i, \boldsymbol{\Theta}]$ and assign proportional number of point instances $\hat{w}_{ij}$ to each cluster.

M2 step. Find parameters $\boldsymbol{\Theta}$ that minimise

$$F_*(\boldsymbol{w}, \boldsymbol{\Theta}) = -\sum_{i=1}^{n}\sum_{j=1}^{k} w_{ij} \log \mathrm{p}[\boldsymbol{x}_i|z_i = j, \boldsymbol{\Theta}]$$

Convergence

- ▷ M2 step clearly reduces the objective function.
- ▷ It is ~~not~~ evident that E1 step does not increase the objective function.
- ▷ We need another way to establish covergence.

Update steps for Gaussian mixture model

E1 step. Compute fractional weights $w_{ij} = \mathrm{p}[z_i = j|\boldsymbol{x}_i, \boldsymbol{\Theta}]$ for each point

$$w_{ij} = \frac{\lambda_j \cdot \mathrm{p}[\boldsymbol{x}_i|z_i = j, \boldsymbol{\Theta}_j]}{\lambda_1 \cdot \mathrm{p}[\boldsymbol{x}_i|z_i = 1, \boldsymbol{\Theta}_1] + \cdots + \lambda_k \cdot \mathrm{p}[\boldsymbol{x}_i|z_i = k, \boldsymbol{\Theta}_k]}$$

M2 step. Find parameters $\boldsymbol{\Theta}$ that minimise $F_*(\boldsymbol{w}, \boldsymbol{\Theta})$ for fixed $\boldsymbol{w}$ where $\boldsymbol{w}$ denotes fractional multiplicity of labels:

$$n_j = \sum_{i=1}^{n} w_{ij} \qquad \boldsymbol{\mu_j} = \frac{1}{n_j} \cdot \sum_{i=1}^{n} w_{ij}\boldsymbol{x}_i$$

$$\lambda_j = \frac{n_j}{n_1 + \cdots + n_k} \qquad \Sigma_j = \frac{1}{n_j} \cdot X^t \mathrm{diag}(\boldsymbol{w}_{*j}) X$$

Why does this algorithm work?

Hard-clustering is non-optimal in theory

Assume that we can compute all likelihoods $\mathrm{p}[\boldsymbol{x}_1, \boldsymbol{x}_2 | \boldsymbol{z}, \mathcal{M}_1]$ for two models

Model	z_1	z_2	p
$\mathcal{M}_1$	0	0	0.18
$\mathcal{M}_1$	0	1	0.17
$\mathcal{M}_1$	1	0	0.14
$\mathcal{M}_1$	1	1	0.15
			0.64

Model	z_1	z_2	p
$\mathcal{M}_2$	0	0	**0.24**
$\mathcal{M}_2$	0	1	0.04
$\mathcal{M}_2$	1	0	0.04
$\mathcal{M}_2$	1	1	0.04
			0.36

Then the hard clustering algorithm chooses the model $\mathcal{M}_2$ although the overall plausibility of $\mathcal{M}_1$ is much higher if we are neutral.

Fix. We should choose the model with the highest posterior probability

$$\mathrm{p}[\boldsymbol{\Theta} | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] = \sum_{\boldsymbol{z}} \mathrm{p}[\boldsymbol{\Theta}, \boldsymbol{z} | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]$$

Further analysis

Now note that

$$\mathrm{p}[\mathbf{\Theta}, \boldsymbol{z} | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] = \frac{\mathrm{p}[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{z} | \mathbf{\Theta}] \cdot \mathrm{p}[\mathbf{\Theta}]}{\mathrm{p}[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]}$$

and thus if we do *not have preferences over models* we get

$$\mathrm{p}[\mathbf{\Theta}, \boldsymbol{z} | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] = c(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \cdot \mathrm{p}[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{z} | \mathbf{\Theta}]$$

Hence, we can solve the optimisation task

$$P(\mathbf{\Theta}) = \sum_{\boldsymbol{z}} \mathrm{p}[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{z} | \mathbf{\Theta}] \to \max$$

Decomposition into individual draws

The maximisation task is hard, since $P(\boldsymbol{\Theta})$ has no nice form

$$P(\boldsymbol{\Theta}) = \sum_{\boldsymbol{z}} \prod_{i=1}^{n} \Pr\left[z_i|\boldsymbol{\Theta}\right] \cdot \mathrm{p}[\boldsymbol{x}_i|\boldsymbol{z}_i, \boldsymbol{\Theta}_{z_i}]$$

$$= \prod_{i=1}^{n} \sum_{j=1}^{k} \Pr\left[z_i = j|\boldsymbol{\Theta}\right] \cdot \mathrm{p}[\boldsymbol{x}_i|\boldsymbol{z}_i = j, \boldsymbol{\Theta}_j]$$

Even the logarithm trick does not help

$$\log P(\boldsymbol{\Theta}) = \sum_{i=1}^{n} \log\left(\sum_{j=1}^{k} \Pr\left[z_i = j|\boldsymbol{\Theta}\right] \cdot \mathrm{p}[\boldsymbol{x}_i|\boldsymbol{z}_i = j, \boldsymbol{\Theta}_j]\right)$$

Heuristic minimisation

Lets assign an arbitrary probability distribution over labels $q(\boldsymbol{z})$ then we can formally provide a lower bound to the log-likelihood

$$\log \mathrm{p}[\boldsymbol{\Theta}|\boldsymbol{x}_1, \ldots \boldsymbol{x}_n] = \log \left(\sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \frac{\mathrm{p}[\boldsymbol{\Theta}, \boldsymbol{z}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]}{q(\boldsymbol{z})} \right)$$

$$\geq \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log \left(\frac{\mathrm{p}[\boldsymbol{\Theta}, \boldsymbol{z}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]}{q(\boldsymbol{z})} \right) = F(\boldsymbol{q}, \boldsymbol{\Theta})$$

Now if we manage to find $q(\boldsymbol{z})$ and $\boldsymbol{\Theta}$ that maximise $F(\boldsymbol{q}, \boldsymbol{\Theta})$ we have located the set of model parameters where the sought log-likelihood $P(\boldsymbol{\Theta})$ must be quite high.

Further analysis

The new minimisation target further decomposes into two separate parts

$$F(\boldsymbol{q}, \boldsymbol{\Theta}) = -\sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log q(\boldsymbol{z}) + \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log \left(\mathrm{p}[\boldsymbol{z}, \boldsymbol{\Theta} | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \right)$$

and we can use two-step minimisation algorithm

E step. Fix $\boldsymbol{\Theta}$ and find optimal weights $q(\boldsymbol{z})$ for all labels

M step. Fix all weights $q(\boldsymbol{z})$ and find parameters $\boldsymbol{\Theta}$ maximising $F(\boldsymbol{q}, \boldsymbol{\Theta})$

What are optimal weights?

It is well known fact that the term

$$\sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log\left(\frac{\mathrm{p}[\boldsymbol{z}, \boldsymbol{\Theta}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]}{q(\boldsymbol{z})}\right)$$

is maximised only if

$$q(\boldsymbol{z}) = C \cdot \mathrm{p}[\boldsymbol{z}, \boldsymbol{\Theta}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] = C_* \cdot \mathrm{p}[\boldsymbol{z}|\boldsymbol{\Theta}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]$$

Since the probabilities of labels decompose into the product so does $q(\boldsymbol{z})$

$$q(\boldsymbol{z}) = \prod_{i=1}^{n} \mathrm{p}[z_i|\boldsymbol{\Theta}_{z_i}, \boldsymbol{x}_i] = \prod_{i=1}^{n} w_{iz_i}$$

where the weights are the same as were in our *practical algorithm*.

What are optimal parameters?

For fixed weights $q(\boldsymbol{z})$ the first term in $F(\boldsymbol{q}, \boldsymbol{\Theta})$ is constant and thus optimal set of parameters can be found by maximising

$$\sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log\left(\mathrm{p}[\boldsymbol{z}, \boldsymbol{\Theta} | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]\right)$$

The latter is equivalent to the following maximisation task

$$\sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log\left(\mathrm{p}[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{z} | \boldsymbol{\Theta}]\right)$$

which can be further decomposed into a simpler sum

$$\sum_{i=1}^{n} \sum_{\boldsymbol{z} \in \mathcal{Z}} q(\boldsymbol{z}) \cdot \log\left(\mathrm{p}[\boldsymbol{x}_i, z_i | \boldsymbol{\Theta}]\right) = \sum_{i=1}^{n} \sum_{\boldsymbol{z} \in \mathcal{Z}} \prod_{\ell=1}^{n} w_{\ell z_\ell} \cdot \log\left(\mathrm{p}[\boldsymbol{x}_i, z_i | \boldsymbol{\Theta}]\right) \quad .$$

Lower-bound regularly coincides with probability

Observe the second factor in the sum after E-step to get further insight

$$
\begin{aligned}
F(\boldsymbol{q}, \boldsymbol{\Theta}) &= \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log\left(\frac{\mathrm{p}[\boldsymbol{\Theta}, \boldsymbol{z}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]}{q(\boldsymbol{z})}\right) \\
&= \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log\left(\frac{\mathrm{p}[\boldsymbol{\Theta}, \boldsymbol{z}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]}{C_* \cdot \mathrm{p}[\boldsymbol{z}|\boldsymbol{\Theta}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]}\right) \\
&= const + \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log\left(\frac{\mathrm{p}[\boldsymbol{\Theta}, \boldsymbol{z}|\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]}{\mathrm{p}[\boldsymbol{z}|\boldsymbol{\Theta}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]}\right) \\
&= const + \sum_{\boldsymbol{z}} q(\boldsymbol{z}) \cdot \log\left(\mathrm{p}[\boldsymbol{\Theta}|\boldsymbol{z}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]\right) \\
&= const + \boxed{\log\left(\mathrm{p}[\boldsymbol{\Theta}|\boldsymbol{z}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]\right)}
\end{aligned}
$$

Connection with the hard-clustering algorithm

The hard-clustering algorithm assigns zero-one probabilities to different labelings $\boldsymbol{z}$ which can be formalised as products of zero-one weights w_{ij}. Thus the parameters are still chosen by maximising

$$\sum_{i=1}^{n} \sum_{\boldsymbol{z} \in \mathcal{Z}} \prod_{\ell=1}^{n} w_{\ell z_\ell} \cdot \log\left(\mathrm{p}[\boldsymbol{x}_i, z_i | \boldsymbol{\Theta}]\right) \quad .$$

Soft-clustering (EM-algorithm) is just a more general maximisation task.

Maximisation task can be converted back to hard-clustering problem by rounding fractional weights to integers.

- ▷ This leads to the dataset with repeated data samples.
- ▷ We can still use the update steps for hard clustering.