Probabilistic Inference and Learning
Lecture 18
Clustering

Philipp Hennig

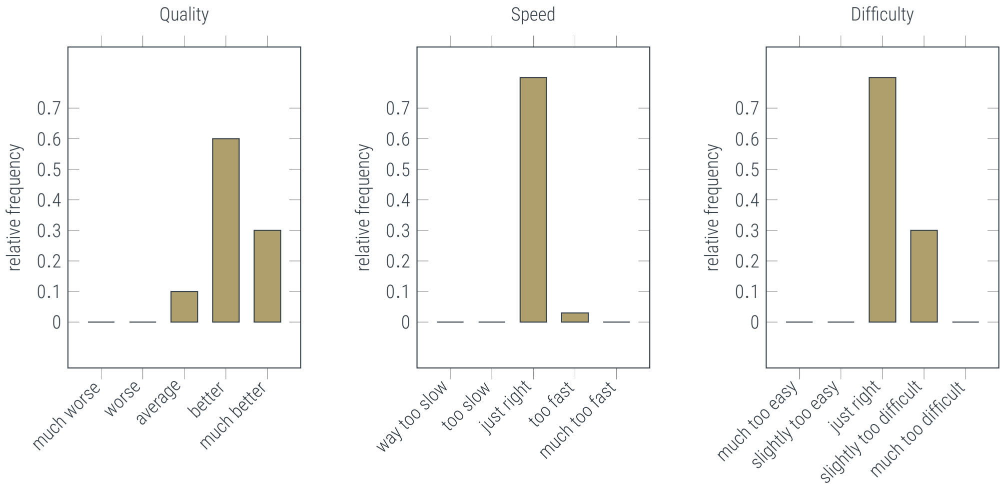
19 December 2018

Eberhard Karls Universität Tübingen

Faculty of Science
Department of Computer Science
Chair for the Methods of Machine Learning

Last Lecture: Debrief

Feedback dashboard



Last Lecture: Debrief

Detailed Feedback

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Things you did not like:

- **Can we have an example exam**???
- it's too cold!
- please repeat questions from the audience
- don't force interactivity
- While you're right about ψ |psʌi|, for *psychology*, the **only admissible** pronounciation is |sʌɪ'kɑlədʒi|

Things you did not understand:

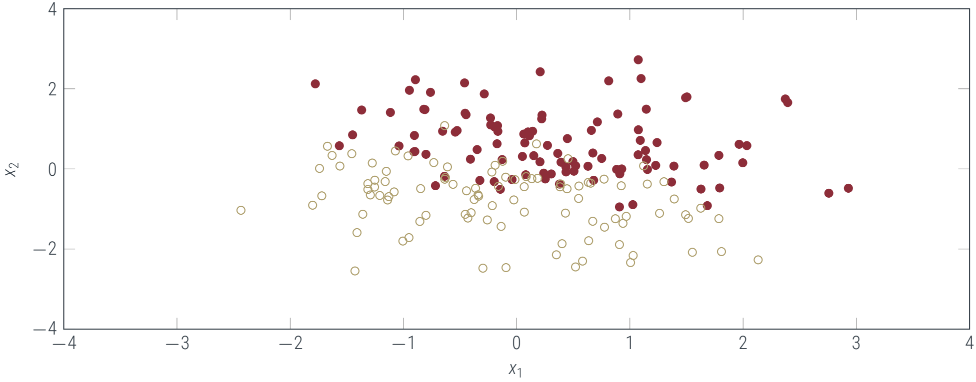
- the arg max part was too fast
- why is this algorithm presented if it is not generally useful?

Things you enjoyed:

- intro and outlook

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

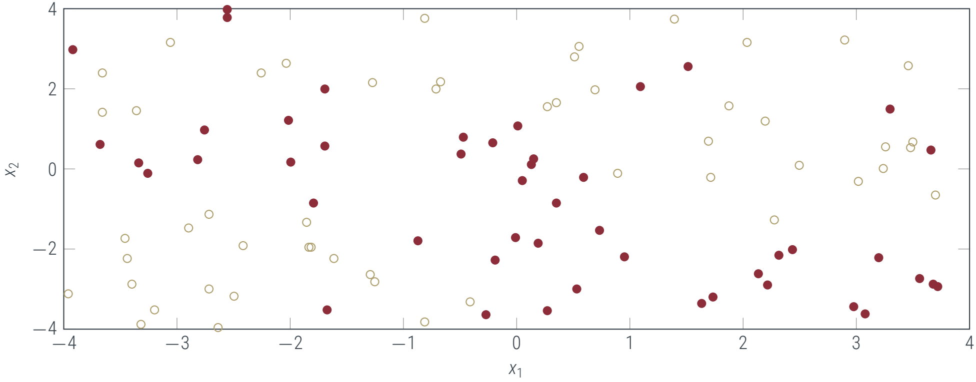
a **supervised** problem that can be solved **discriminatively** in a *linear* fashion



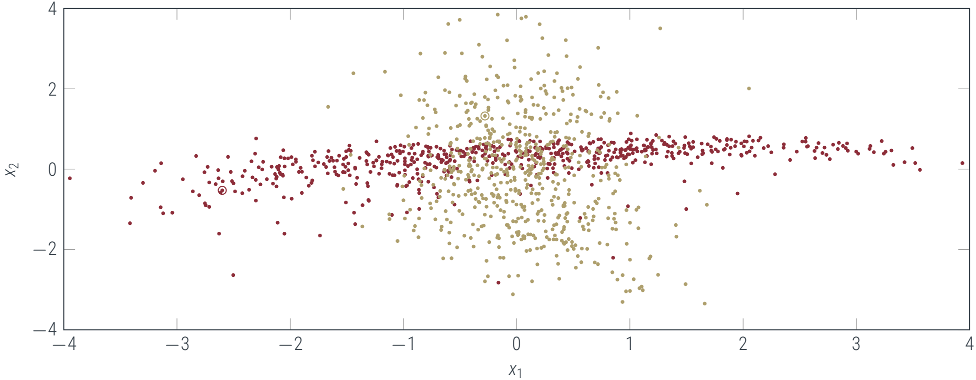
A Typography of Machine Learning Problems

Unsupervised, Supervised, Generative, Discriminative

a **supervised** problem that can be solved **discriminatively** in a *nonlinear* fashion



a **supervised** problem that can be solved **generatively** (in a Gaussian fashion?)



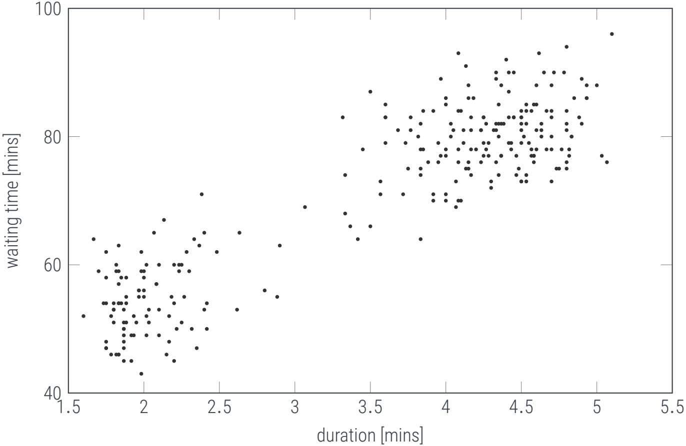
A Typography of Machine Learning Problems

Unsupervised, Supervised, Generative, Discriminative

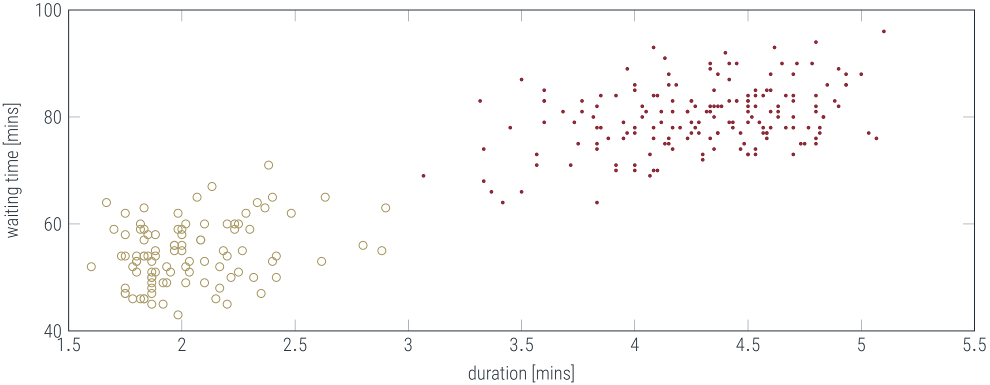
an **unsupervised** problem

`https://www.stat.cmu.edu/ larry/all-of-statistics/=data/faithful.dat`

Azzalini, A. and Bowman, A. W. (1990). *A look at some data on the Old Faithful geyser.* Applied Statistics 39, 357-365.



a **clustering** of the **unsupervised** problem



A Typography of Machine Learning Problems

nb: this list is not complete!

Task types

Supervised given **input-output pairs** $[x_i \in \mathbb{X}, y_i \in \mathbb{Y}]_{i=1,\dots,n} = (X_{\text{train}}, Y_{\text{train}})$, predict $y_{\text{test}}(x_{\text{test}})$

- **Regression** $\mathbb{Y} = \mathbb{R}^d$
- **Classification** $\mathbb{Y} \subset \mathbb{N} = \sigma(\mathbb{R}^d)$
- **Structured Output** $\mathbb{Y} \simeq f(\mathbb{R}^d)$
- **Time Series** $\mathbb{X} = \mathbb{R}$

Unsupervised given collection $[x_i \in \mathbb{X}]_{i=1,\dots,n}$

- **Generative Modelling** assume $x_i \sim p$. Make more $x_j \sim p$
- **Clustering** assign a class $c_i \in [1, \dots, C]$ for each x_i (why?)

Note: there are many more task types and sub-types (semi-supervised, dimensionality reduction, matrix factorization, causal inference, …)

We will see that **Clustering** is a subtype of (or even the same thing as?) Generative Modelling.
Clustering is also primarily a way to reduce dimensionality/complexity;
it should be used carefully if the goal is to “discover” structure.

One of the oldest Clustering Methods

in the dark days of the 20th century

Hugo Steinhaus [1887–1972]

- born in Jasło (then Austro-Hungary), died in Wrocław
- PhD 1911, Göttingen with David Hilbert
- in hiding during the "third Reich"
- PhD Advisor to Stefan Banach and Marc Kac
- Keeper of the *Scottish Book*
- Steinhaus, H. (1957). *Sur la division des corps matériels en parties*. Bull. Acad. Polon. Sci. 4 (12): 801–804.

Given $\{x_i\}_{i=1,\dots,n}$

Init Set k means $\{m_k\}$ to random values

Assign each datum x_i to its *nearest mean*. One could denote this by an integer variable

$$k_i = \arg\min_k \|m_k - x_i\|^2$$

or by binary responsibilities

$$r_{ki} = \begin{cases} 1 & \text{if } k_i = k \\ 0 & \text{else} \end{cases}$$

Update set the means to the sample mean of each cluster

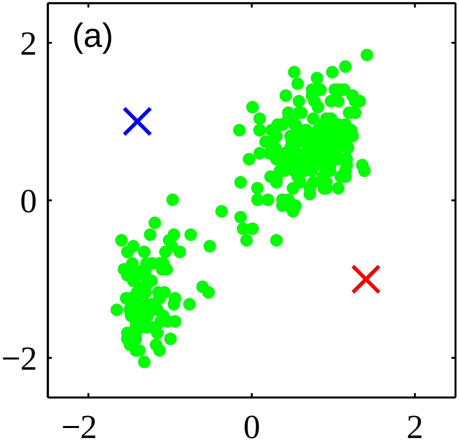
$$m_k \leftarrow \frac{1}{R_k} \sum_i^n r_{ki} x_i \qquad \text{where } R_k := \sum_i r_{ki}$$

Repeat until the assignments do not change

k-Means Clustering

Example on Old Faithful

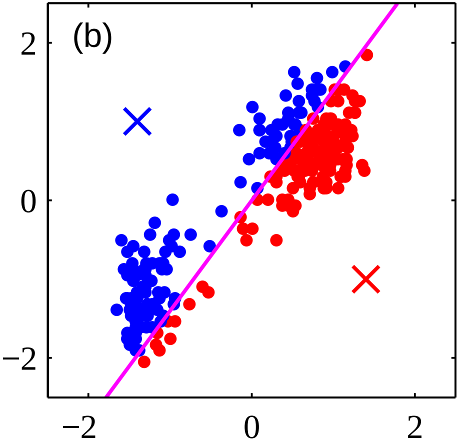
[Figure 9.1 in Bishop, 2006]



k-Means Clustering

Example on Old Faithful

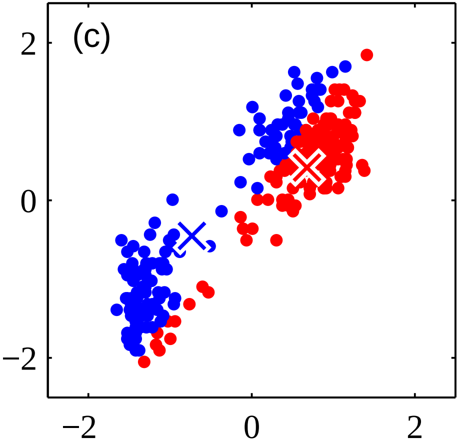
[Figure 9.1 in Bishop, 2006]

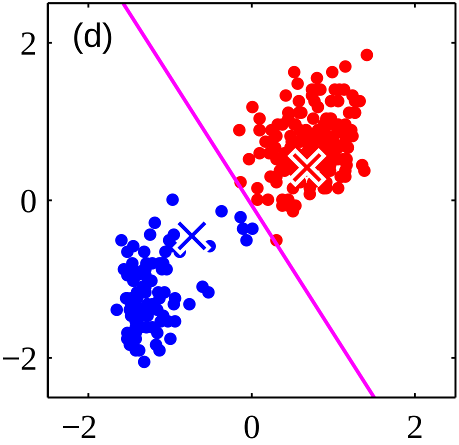


k-Means Clustering

Example on Old Faithful

[Figure 9.1 in Bishop, 2006]

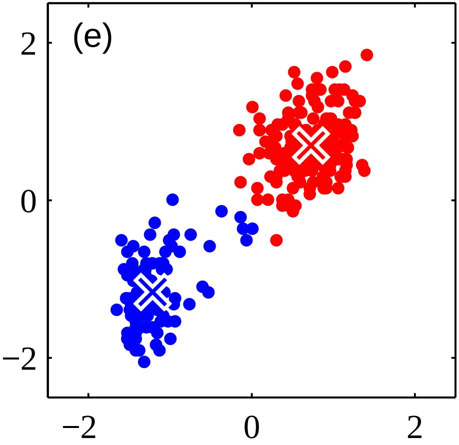



(d)
2
0
−2
−2
0
2

k-Means Clustering

Example on Old Faithful

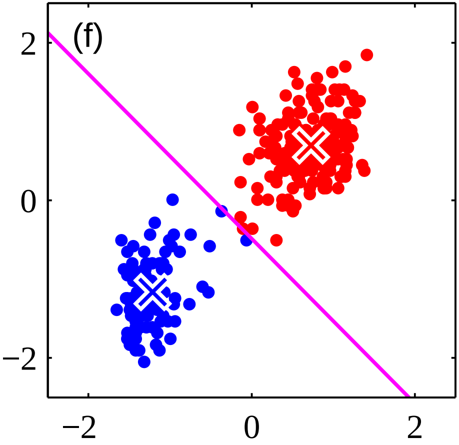
[Figure 9.1 in Bishop, 2006]



k-Means Clustering

Example on Old Faithful

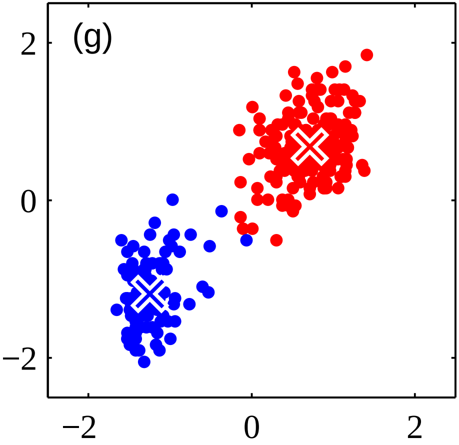
[Figure 9.1 in Bishop, 2006]



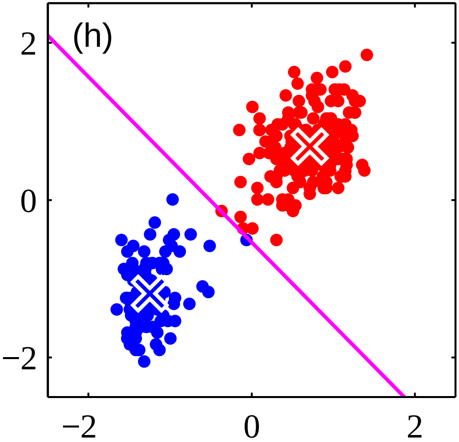
k-Means Clustering

Example on Old Faithful

[Figure 9.1 in Bishop, 2006]



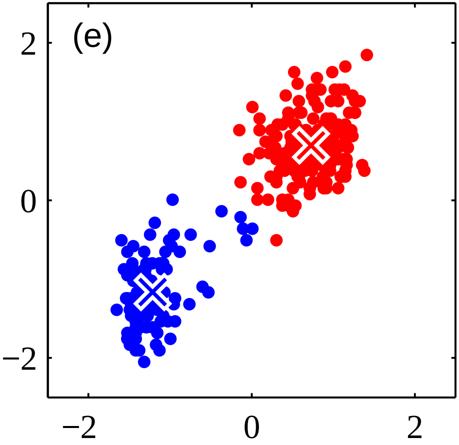
[Figure 9.1 in Bishop, 2006]



k-Means Clustering

Example on Old Faithful

[Figure 9.1 in Bishop, 2006]



k-means always converges

for an interesting reason …

Definition (Lyapunov Function)

In the context of iterative algorithms, a *Lyapunov Function J* is a positive function of the algorithm's state variables that decreases in each step of the algorithm.

The existence of a Lyapunov function means that one can think about the algorithm in question as an optimization routine for J. It also guarantees convergence of the algorithm at a *local* (not necessarily global!) minimum of J

Aleksandr M. Lyapunov
(1857—1918)

k-means always converges …

for an interesting reason …

1 **procedure** k-MEANS$(\boldsymbol{x}, k)$
2 $\quad \boldsymbol{m} \leftarrow \text{RAND}(k)$ // initialize
3 $\quad$ **while** not converged **do**
4 $\quad\quad \boldsymbol{r} \leftarrow \text{FIND}(\min(\|\boldsymbol{m} - x\|^2))$ // set responsibilities
5 $\quad\quad \boldsymbol{m} \leftarrow \boldsymbol{r}\boldsymbol{x} \oslash \boldsymbol{r}\mathbf{1}$ // set means
6 $\quad$ **end while**
7 $\quad$ **return** $\boldsymbol{m}$
8 **end procedure**

Consider $$J(r, m) := \sum_i^n \sum_j^k r_{ik} \|x_i - m_k\|^2$$

- step 4 always decreases J (by definition)
- step 5 always decreases J, because

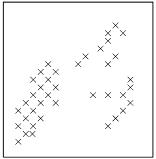
$$\frac{\partial}{\partial m_k} J(r, m) = -2 \sum_i^n r_{ik}(x_i - m_k) = 0 \quad \Rightarrow \quad m_k = \frac{\sum_i r_{ik} x_i}{\sum_i r_{ik}} \qquad \frac{\partial^2 J(r, m)}{\partial m_k^2} = 2 \sum_i r_{ik} > 0$$

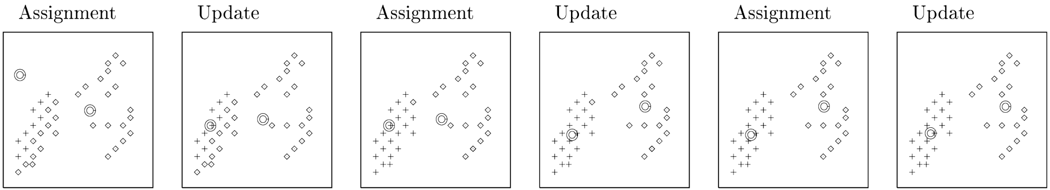
k-means has pathologies

k-means can work well …

figures from DJC MacKay, ITILA, 2003

Data:





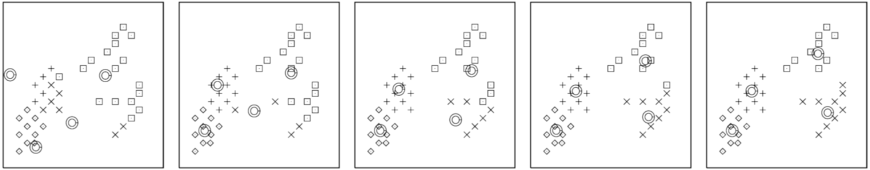
k-means has pathologies

…but it has no way to set *k*

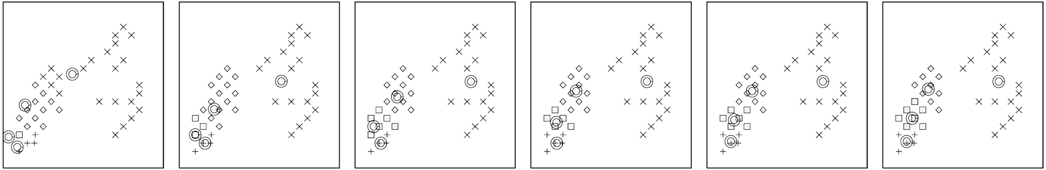
EBERHARD KARLS UNIVERSITÄT TÜBINGEN

figures from DJC MacKay, ITILA, 2003

Run 1



Run 2

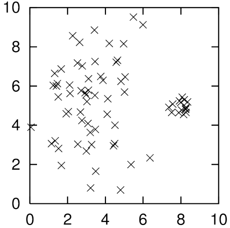


k-means has pathologies

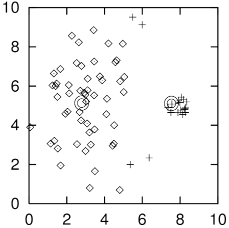
…or to set the *shape* of the clusters

figures from DJC MacKay, ITILA, 2003

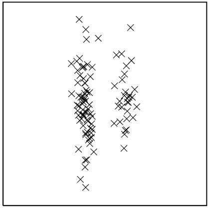
(a)



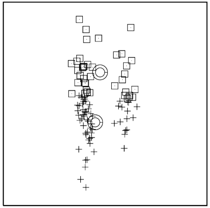
(b)



(a)



(b)



“Soft k-means”

beginning to fix k-means

[presentation from DJC MacKay, ITILA, 2003]

- replace the hard assignments $r_{ik} = \mathbb{I}(\arg\min_k \|m_k - x_i\|^2)$ with the *softmax*

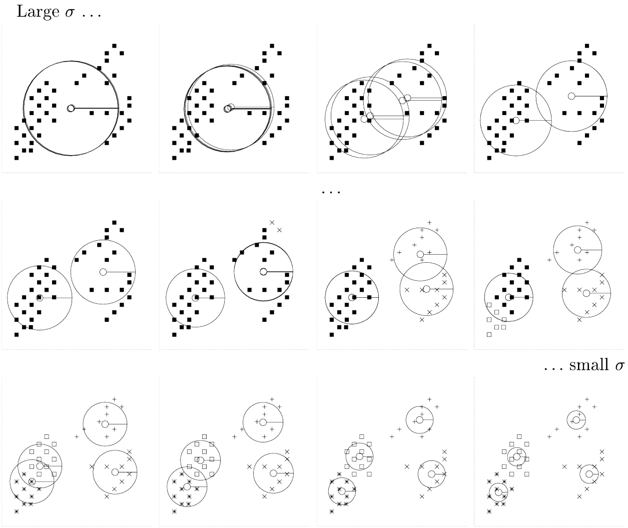
$$r_{ik} = \frac{\exp(-\beta\|m_k - x_i\|^2)}{\sum_{k'} \exp(-\beta\|m_{k'} - x_i\|^2)}$$

- β is the *stiffness*. For $\beta \rightarrow \infty$, get back k-means

Soft k-means fixes *some* of the problems of k-means

shared responsibility allows overlap ($\sigma := \beta^{-1/2}$)

figures from DJC MacKay, ITILA, 2003



- *k*-means is a simple algorithm that always finds **a** stable clustering
- the resulting clusterings can be unintuitive. They do not capture shape of clusters or their number, and are subject to random fluctuations
- **soft *k*-means** can address some of these issues by allowing points to be partly assigned to several clusters at the same time. But it requires tuning the stiffness parameter β

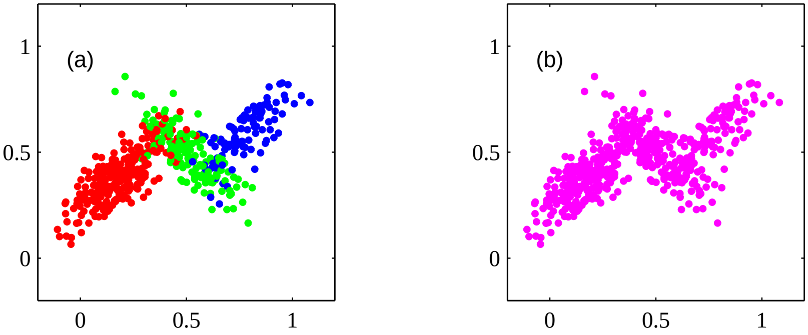
a probabilistic interpretation of *k*-means yields clarity and allows fitting all parameters

Gaussian Mixtures

A generative model for *k*-means

[Figure from Bishop, PRML, 2006]

$$p(x \mid \pi, \mu, \Sigma) = \sum_j^k \pi_j \mathcal{N}(x; \mu_j, \Sigma_j) \qquad \pi_j \in [0,1], \quad \sum_j \pi_j = 1$$

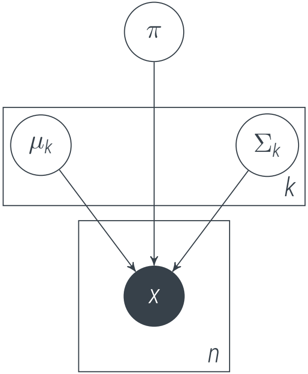


Soft k-means as maximum likelihood

for the Gaussian mixture model

- Given dataset $[x_i]_{i=1,\dots,n}$, want to learn generative model (π, μ, Σ)

$$p(x \mid \pi, \mu, \Sigma) = \prod_{i}^{n} \sum_{j}^{k} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \qquad (\star)$$



Soft k-means as maximum likelihood

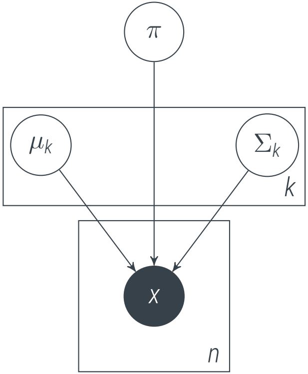
for the Gaussian mixture model

- Given dataset $[x_i]_{i=1,\dots,n}$, want to learn generative model (π, μ, Σ)

$$p(x \mid \pi, \mu, \Sigma) = \prod_i^n \sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \qquad (\star)$$

- Ideally, want Bayesian inference

$$p(\pi, \mu, \Sigma \mid x) = \frac{p(x \mid \pi, \mu, \Sigma) \cdot p(\pi, \mu, \Sigma)}{p(x)}$$



Soft k-means as maximum likelihood

for the Gaussian mixture model

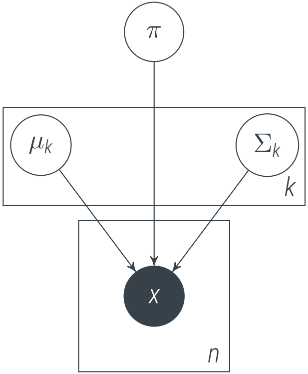
- Given dataset $[x_i]_{i=1,\dots,n}$, want to learn generative model (π, μ, Σ)

$$p(x \mid \pi, \mu, \Sigma) = \prod_i^n \sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \qquad (\star)$$

- Ideally, want Bayesian inference

$$p(\pi, \mu, \Sigma \mid x) = \frac{p(x \mid \pi, \mu, \Sigma) \cdot p(\pi, \mu, \Sigma)}{p(x)}$$

- likelihood is not an exponential family — no obvious conjugate prior



posterior (and likelihood) do not factorize over μ, π, Σ! $\mu \not\perp\!\!\!\perp \pi \mid x$

Soft k-means as maximum likelihood

for the Gaussian mixture model

Let's try to maximize the likelihood ($\star$) for π, μ, Σ (tool 1)

$$\log p(x \mid \pi, \mu, \Sigma) = \sum_i^n \log \left(\sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)$$

To maximize w.r.t. μ set gradient of log likelihood to 0:

$$\nabla_{\mu_j} \log p(x \mid \pi, \mu, \Sigma) = -\frac{1}{2} \sum_i^n \underbrace{\frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}}_{=: r_{ji}} \Sigma_j (x_i - \mu_j)$$

$$\nabla_{\mu_j} \log p = 0 \qquad \Rightarrow \mu_j = \frac{1}{R_j} \sum_i^n r_{ji} x_i \qquad R_j := \sum_i r_{ji}$$

Soft k-means as maximum likelihood

for the Gaussian mixture model

Let's try to maximize the likelihood ($\star$) for π, μ, Σ (tool 1)

$$\log p(x \mid \pi, \mu, \Sigma) = \sum_i^n \log \left(\sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)$$

To maximize w.r.t. Σ set gradient of log likelihood to 0 (note $\partial \log |\Sigma^{-1}| / \partial \Sigma = \Sigma$):

$$\nabla_{\Sigma_j} \log p(x \mid \pi, \mu, \Sigma) = -\frac{1}{2} \sum_i^n \underbrace{\frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}}_{=: r_{ji}} \left((x_i - \mu_j)(x_i - \mu_j)^\intercal - \Sigma_j \right)$$

$$\nabla_{\Sigma_j} \log p = 0 \qquad \Rightarrow \Sigma_j = \frac{1}{R_j} \sum_i^n r_{ji} (x_i - \mu_j)(x_i - \mu_j)^\intercal \qquad R_j := \sum_i r_{ji}$$

Soft k-means as maximum likelihood

for the Gaussian mixture model

Let's try to maximize the likelihood $(\star)$ for π, μ, Σ (tool 1)

$$\log p(x \mid \pi, \mu, \Sigma) = \sum_i^n \log \left(\sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)$$

To maximize w.r.t. π, enforce $\sum_j \pi_j = 1$ by introducing Lagrange multiplier λ and optimize

$$\nabla_{\pi_j} \log p(x \mid \pi, \mu, \Sigma) + \lambda \left(\sum_j \pi_j - 1 \right) = \sum_i^n \frac{\mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)} + \lambda$$

$$0 = \sum_i^n \pi_j \frac{\mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)} + \lambda \pi_j = \sum_i^n r_{ij} + \lambda \pi_j$$

$$\sum_j \pi_j = 1 \Rightarrow \lambda = -N \qquad \Rightarrow \qquad \pi_j = \frac{R_j}{n}$$

The EM Algorithm (for Gaussian mixtures)

Refinement of soft k-means and k-means

If we know the responsibilities r_{ij}, we can optimize μ, π analytically. And if we know μ, π, we can set r_{ij}! Thus

1. initialize μ, π (e.g. random μ, uniform π)
2. Set
$$r_{ij} = \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'}^{k} \pi_{j'} \mathcal{N}(x_i; \mu_{j'}, \Sigma_{j'})}$$
3. Set
$$R_j = \sum_i r_{ji} \qquad \mu_j = \frac{1}{R_j} \sum_i^n r_{ij} x_i \qquad \Sigma_j = \frac{1}{R_j} \sum_i^n r_{ij} (x_i - \mu_j)(x_i - \mu_j)^\intercal \qquad \pi_j = \frac{R_j}{n}$$

- Note that π is essentially given through r_{ij}, thus can be incorporated into the first step

The connection to (soft) k-means

Refinement of soft k-means and k-means with cluster probabilities

Set $\Sigma_j = \beta^{-1} I$ for all $j = 1, \dots, k$

1. initialize μ, π (e.g. random μ, uniform π)
2. Set
$$r_{ij} = \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'}^{k} \pi_{j'} \mathcal{N}(x_i; \mu_{j'}, \Sigma_{j'})} = \frac{R_j \exp(-\beta \|x_i - m_j\|^2)}{\sum_{j'} R_{j'} \exp(-\beta \|x_i - m_{j'}\|^2)}$$
3. Set
$$R_j = \sum_i r_{ji} \qquad \mu_j = \frac{1}{R_j} \sum_i^n r_{ij} x_i \qquad \left(\Sigma_j = \frac{1}{R_j} \sum_i^n r_{ij} (x_i - \mu_j)(x_i - \mu_j)^\intercal \qquad \pi_j = \frac{R_j}{n} \right)$$

the EM algorithm is a refinement of soft k-means

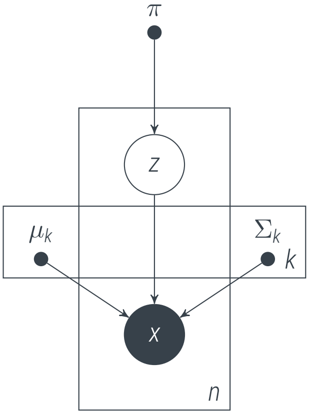
- For $\beta \to \infty$, get back k-means
- What is r_{ij}?

- consider binary $z_j \in \{0; 1\}$ with $\sum_j z_j = 1$ ("one-hot")
- what is $p(x, z)$? Let's write it as $p(x, z) = p(x \mid z)p(z)$ with

$$p(z_j = 1) = \pi_j \qquad \Rightarrow p(z) = \prod_j^k \pi_j^{z_j}$$

$$p(x \mid z_j = 1) = \mathcal{N}(x; \mu_j, \Sigma_j) \qquad \Rightarrow p(x \mid z) = \prod_j^k \mathcal{N}(x \mid \mu_k, \Sigma_j)^{z_k}$$

$$p(x) = \sum_j p(z = j)p(x \mid z = j) = \sum_j^k \pi_j \mathcal{N}(x; \mu_j, \Sigma_j)$$



Joint Generative Model

For the Gaussian Mixture

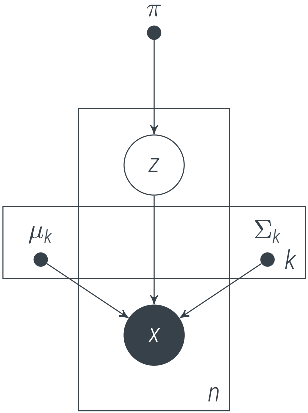
$$p(x, z \mid \pi, \mu, \Sigma) = \prod_i^n \prod_j^k \pi_j^{z_{ij}} \mathcal{N}(x_i; \mu_j, \Sigma_j)^{z_{ij}}$$

$$p(z_{ij} = 1 \mid x_i, \mu, \Sigma) = \frac{p(z_{ij} = 1) p(x_i \mid z_{ij} = 1, \mu_j, \Sigma_j)}{\sum_{j'}^k p(z_{ij'} = 1) p(x_i \mid z_{ij'} = 1, \mu_j, \Sigma_j)}$$

$$= \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}$$

$$= r_{ij}$$

r_{ij} is the marginal posterior probability (**[E]xpectation**) for $z_{ij} = 1$!



Given μ, Σ, have a simple distribution for z. And, given z, μ, Σ show up in a tractable form.

The Expectation Maximization Algorithm

Refinement of soft *k*-means and *k*-means with cluster probabilities

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Set $\Sigma_j = \beta^{-1} I$ for all $j = 1, \dots, k$

1. initialize μ, π (e.g. random μ, uniform π)
2. Compute **EXPECTED** value of z:

$$r_{ij} = \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'}^{k} \pi_{j'} \mathcal{N}(x_i; \mu_{j'}, \Sigma_{j'})} = \frac{R_j \exp(-\beta \|x_i - m_j\|^2)}{\sum_{j'} R_{j'} \exp(-\beta \|x_i - m_{j'}\|^2)}$$

3. **MAXIMIZE** Likelihood

$$R_j = \sum_i r_{ji} \qquad \mu_j = \frac{1}{R_j} \sum_i^n r_{ij} x_i \qquad \left(\Sigma_j = \frac{1}{R_j} \sum_i^n r_{ij} (x_i - \mu_j)(x_i - \mu_j)^\intercal \qquad \pi_j = \frac{R_j}{n} \right)$$

the EM algorithm is an *iterative maximum likelihood* algorithm.

Does it converge?

CODE

`EM_Algorithm.ipynb`

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Summary:

- **Clustering** is a paradigm to learn a generative model for data by mapping it into a low-dimensional discrete space of generating distributions
- classic algorithms like k-**means** do not capture this view, but they implicitly do it anyway
- the probabilistic formulation helps clarify the setting, but also to fix pathologies
- the **EM algorithm** fits a probabilistic model by alternating between
  1. computing the **expectation** of the cluster membership for each datum
  2. **maximizing** the likelihood of the cluster parameters

After Christmas, we will return to EM and find that it is a special case of a more general inference scheme.