Machine Learning

MIRI Master

Lluís A. Belanche

belanche@cs.upc.edu

Soft Computing Research Group
Dept. de Ciències de la Computació (Computer Science)
Universitat Politècnica de Catalunya

Spring Semester 2016-2017

LECTURE 7: Probabilistic clustering (k-means and E-M)

Probabilistic clustering (k-means and E-M)

Outline

1. Introduction
2. The k-means algorithm
3. Choosing the number of clusters
4. Gaussian mixture models (MoGs)
5. The E-M algorithm for MoGs
6. Closing remarks

Probabilistic clustering (k-means and E-M)

Introduction

The goal of **clustering** is to partition a data sample into groups ("*clusters*") in such a way that observations in the same cluster tend to be more similar than observations in different clusters

There are vary many (families of) algorithms in the literature:

Hierarchical bottom-up/top-down: single linkage, average linkage, Ward, ...

Probabilistic use MoGs: k-means and E-M

Possibilistic use memberships: fuzzy c-means clustering

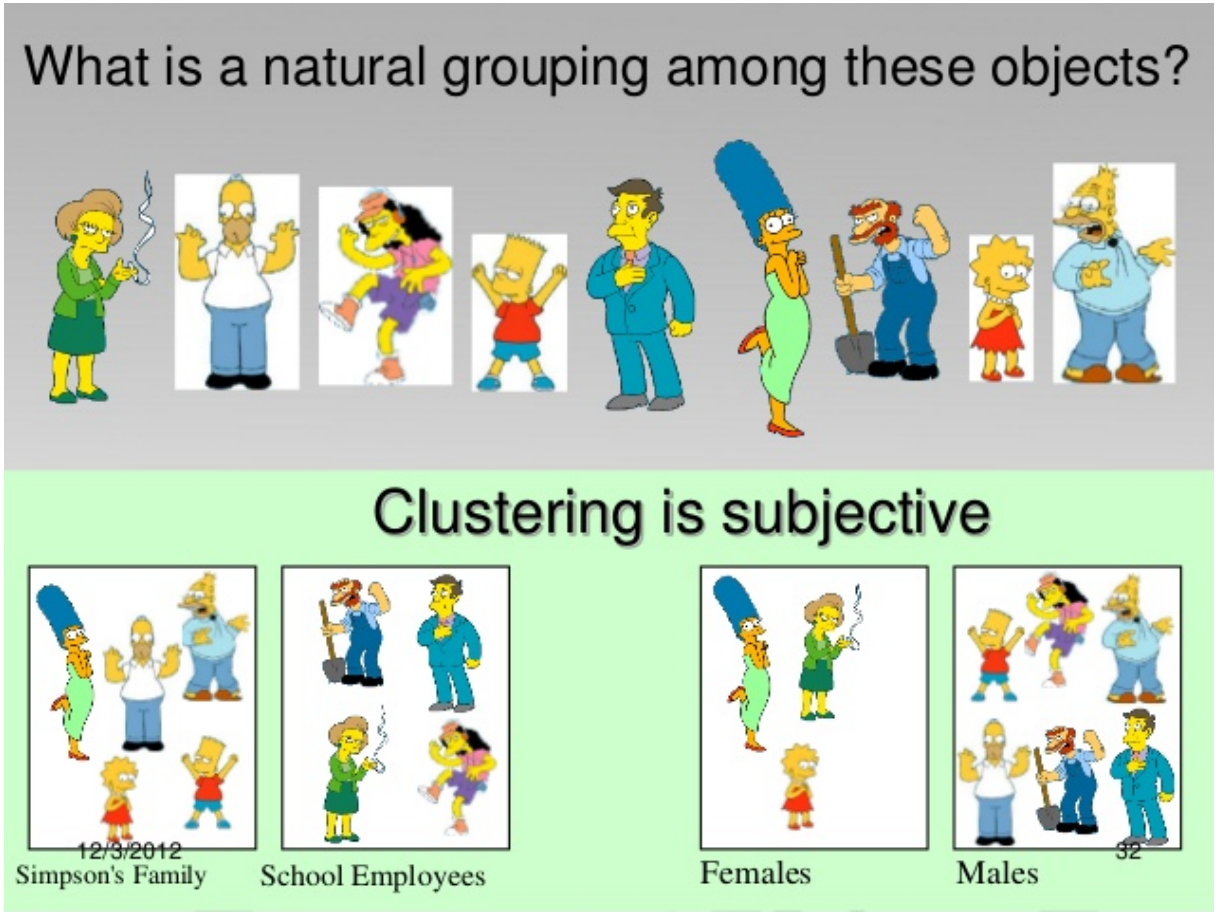
Algorithmic greedy/hill-climbing (swapping elements between clusters, *e.g.* PAM)

Spectral use the spectrum (eigenvalues) of the data similarity matrix to perform dimensionality reduction before clustering in fewer dimensions

Density-based clusters are connected dense regions in the data space (*e.g.* DBSCAN)

Probabilistic clustering (k-means and E-M)

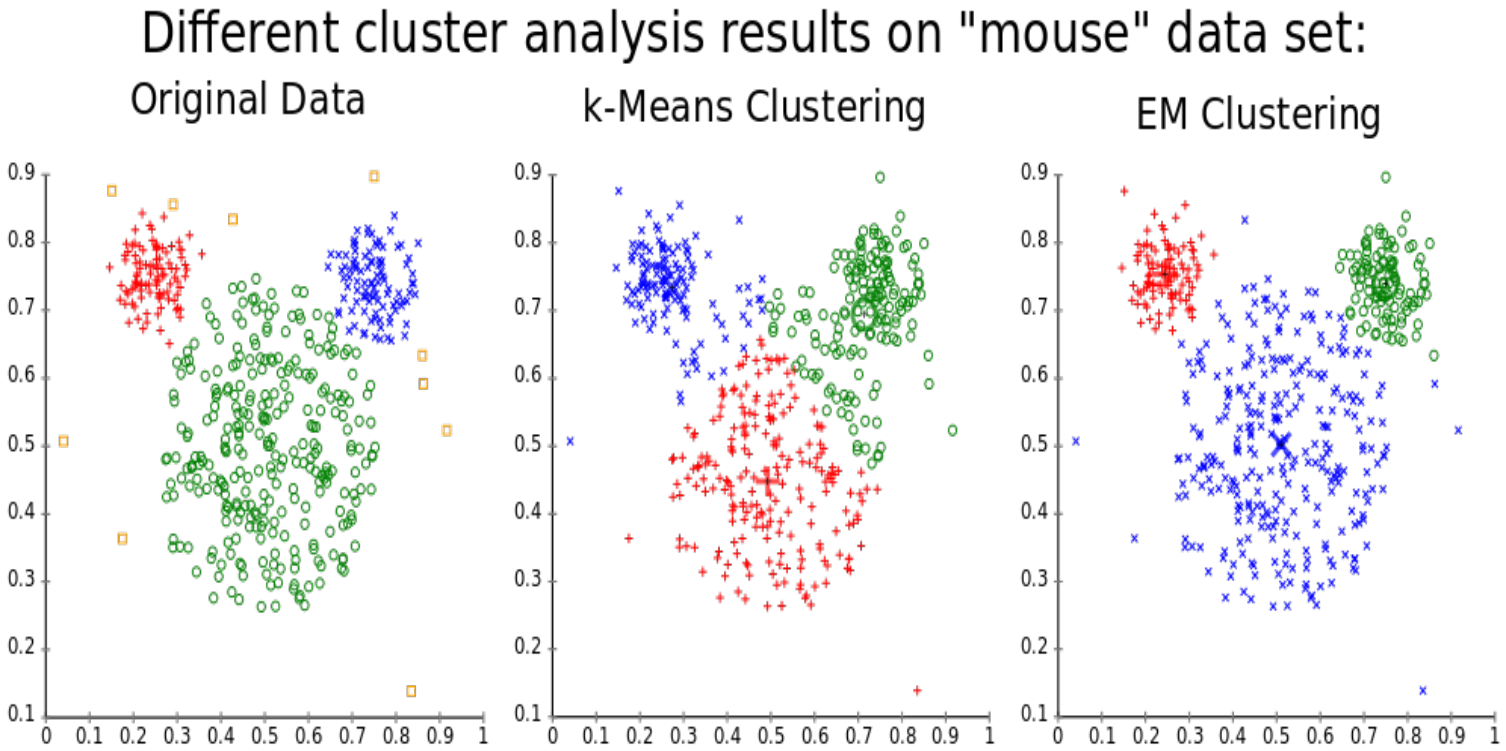
Introduction



Clustering is one of the more subjective ML tasks

Probabilistic clustering (k-means and E-M)

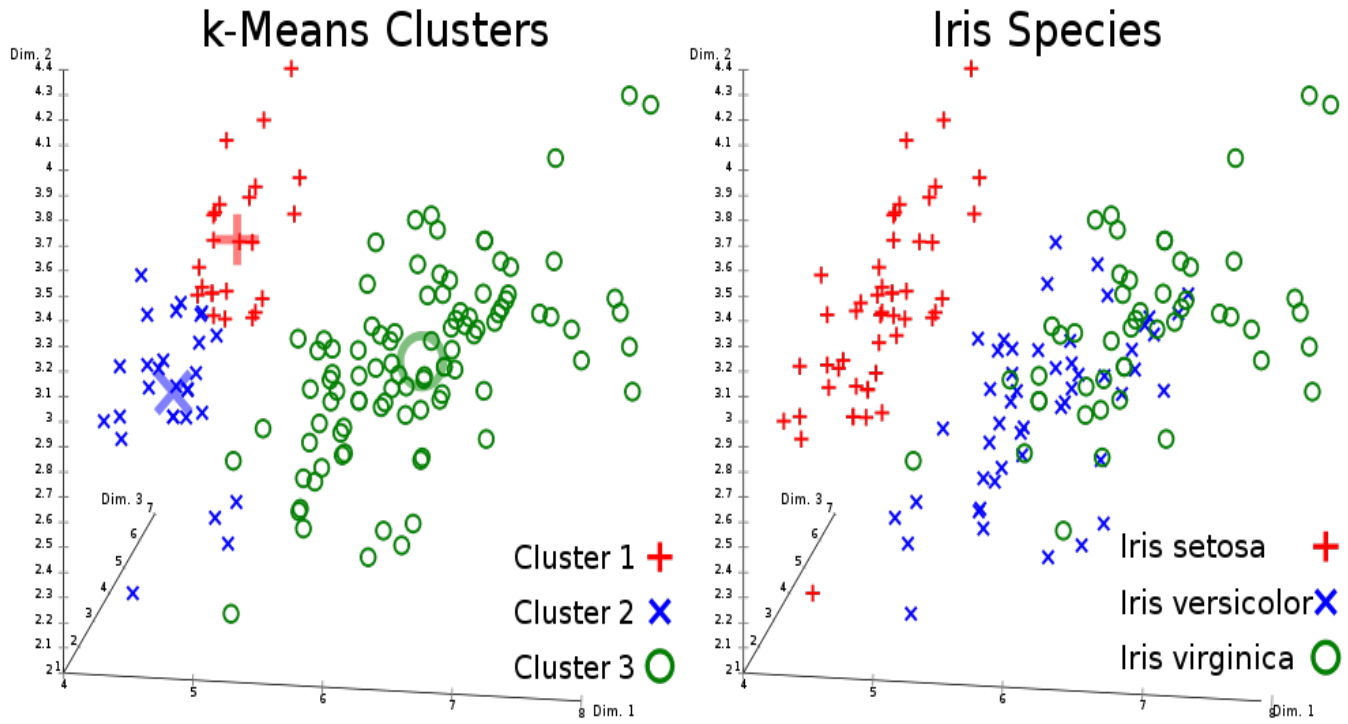
Introduction



Every clustering methods "sees" the data differently

Probabilistic clustering (k-means and E-M)

Introduction



We sometimes expect clusters to correspond to classes

Probabilistic clustering (k-means and E-M)

Introduction

- The number of different clusterings is astronomical:

$$S(N,K) = \frac{1}{K!}\sum_{k=1}^{K}(-1)^{K-k}\binom{K}{k}k^N$$

(number of ways to partition a set of N objects into K nonempty subsets: Stirling numbers of the second kind)

- The sum over the possible K gives $B(N) = \sum_{K=1}^{N} S(N,K)$

(number of partitions of a set with N members: the N-th Bell number)

$S(10,4) = 35{,}105 \qquad S(19,4) \approx 10^{10} \qquad B(71) \approx 4 \cdot 10^{74}$

Probabilistic clustering (k-means and E-M)

The k-means algorithm

Consider the problem of grouping an i.i.d. data sample of N *unlabelled* observations $D = \{\boldsymbol{x}_n\}_{n=1,\ldots,N}$, $\boldsymbol{x}_n \in \mathbb{R}^d$, into K disjoint groups.

Idea: introduce a set of prototypes (aka **centroids**: cluster centers)

$$\mathcal{P} = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K\},\ \boldsymbol{\mu}_k \in \mathbb{R}^d$$

The goal is find $\mathcal{P}$ such that the distances between every $\boldsymbol{x}_n$ and its assigned cluster (i.e. its prototype) are globally minimized

Probabilistic clustering (k-means and E-M)

The k-means algorithm

We introduce a set of indicator variables:

$$r_{nk} := \begin{cases} 1 & \text{if } \boldsymbol{x}_n \text{ is assigned to cluster } k \\ 0 & \text{otherwise} \end{cases}$$

and an objective function:

$$J(\mathcal{P}) := \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2$$

Probabilistic clustering (k-means and E-M)

The k-means algorithm

The goal is to find $\mathcal{P}$ and the $\{r_{nk}\}$ that minimize $J(\mathcal{P})$

bad news: this is an NP-hard problem
illusion: if we have a pocket oracle ... then good news

1. if we asked the oracle for the right prototypes, optimizing for the assignments would be easy!

2. if we asked the oracle for the right assignments, optimizing for the prototypes would be easy!

Probabilistic clustering (k-means and E-M)

The k-means algorithm

1. Initialize the prototypes $\mathcal{P}$

2. **repeat**

   a) (re)compute the assignments $\{r_{nk}\}$ in an optimal way

   b) (re)compute the prototypes $\mathcal{P}$ in an optimal way

3. **until** no further changes in assignments
   (**or** max. number of iterations reached)

Probabilistic clustering (k-means and E-M)

The k-means algorithm

Initialize the prototypes $\mathcal{P}$

The preferred initialization method randomly chooses K observations from the data set and uses these as the initial prototypes $\boldsymbol{\mu}_k$

(Re)compute the assignments $\{r_{nk}\}$ in an optimal way

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_{1 \le j \le K} \|\boldsymbol{x}_n - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{otherwise} \end{cases}$$

Probabilistic clustering (k-means and E-M)

The k-means algorithm

(Re)compute the prototypes $\boldsymbol{\mu}_k$ in an optimal way

The objective function

$$J(\mathcal{P}) = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2$$

is quadratic on the $\boldsymbol{\mu}_k$, therefore

$$\frac{\partial J(\mathcal{P})}{\partial \boldsymbol{\mu}_k} = \sum_{n=1}^{N} r_{nk} \frac{\partial \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2}{\partial \boldsymbol{\mu}_k} = 2 \sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) = 0$$

Probabilistic clustering (k-means and E-M)

The k-means algorithm

which gives:

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} r_{nk}\boldsymbol{x}_n}{\sum_{n=1}^{N} r_{nk}}$$

the average (the centroid!) of those $\boldsymbol{x}_n$ currently assigned to cluster k

Probabilistic clustering (k-means and E-M)

The k-means algorithm

Advantages:

1. Easy to implement and to apply
2. Pretty fast, even on large data sets, can be run many times

Limitations:

1. The loop ends up in a local minimum of $J(\mathcal{P})$
2. We must specify the number of clusters K beforehand
3. Cluster assignments are "hard"
4. Initialization greatly influences the result

Probabilistic clustering (k-means and E-M)

Choosing the number of clusters

The Calinski-Harabasz index is defined as

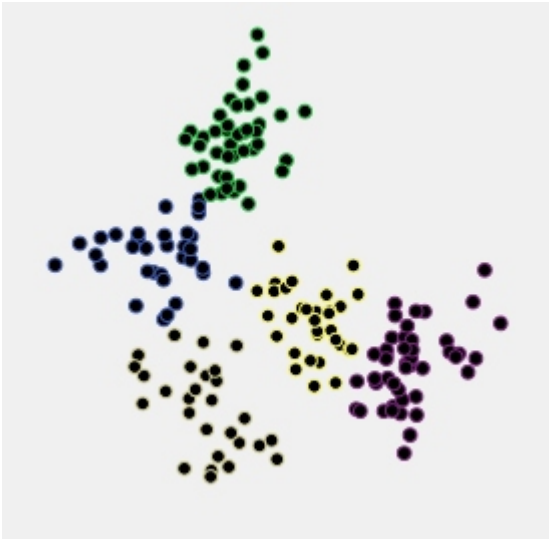
$$\mathrm{CH}(K) := \frac{\sum_{k=1}^{K} N_k \|\boldsymbol{\mu}_k - \bar{\boldsymbol{x}}\|^2}{\sum_{k=1}^{K} \sum_{n=1}^{N} r_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2} \cdot \frac{N-K}{K-1} = \frac{S_B}{S_W} \cdot \frac{N-K}{K-1}$$

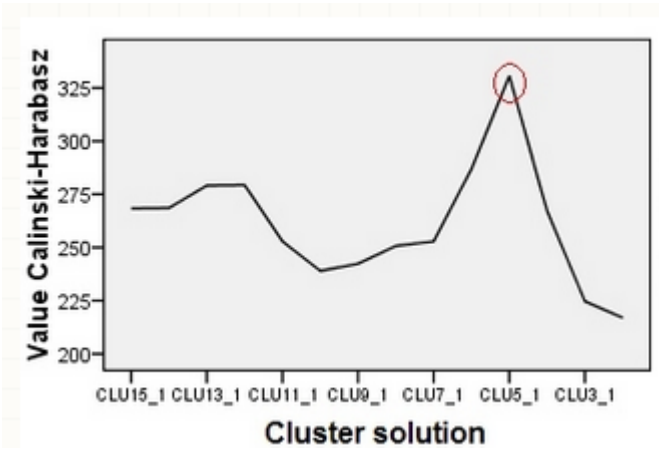
S_B is the overall **between-cluster** scatter
S_W is the overall **within-cluster** scatter

Well-defined clusterings will have a large S_B and a small S_W

Probabilistic clustering (k-means and E-M)

Choosing the number of clusters





Left: scatterplot of data generated as 5 normally distributed clusters, lying quite close to each other

Right: clustering solutions from $K = 15$ clusters through $K = 2$ clusters against CH(K)

Probabilistic clustering (k-means and E-M)

Gaussian mixture models

A **Mixture of Gaussians** (MoG) is a very flexible and elegant way for modelling an unknown density $p(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathbb{R}^d$:

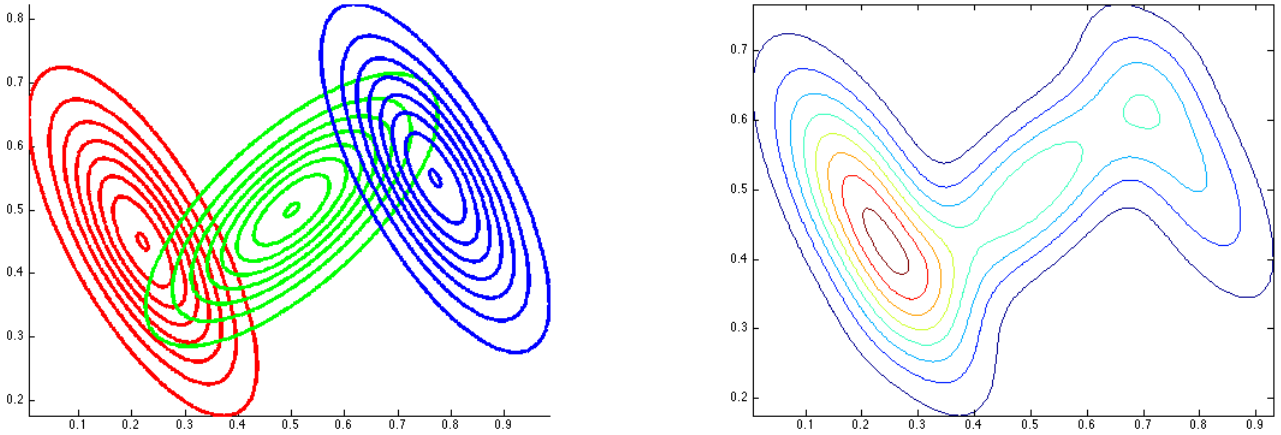
$$p(\boldsymbol{x}) = \sum_{k=1}^{K} p(\boldsymbol{x}, k) = \sum_{k=1}^{K} p(\boldsymbol{x}|k)P(k) = \sum_{k=1}^{K} \pi_k N(\boldsymbol{x}; \boldsymbol{\mu}_k, \Sigma_k)$$

- Every $N(\boldsymbol{x}; \boldsymbol{\mu}_k, \Sigma_k)$ is a multivariate Gaussian known as a **component** of the mixture

- The π_k are the mixture **coefficients**, such that $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^{K} \pi_k = 1$

Probabilistic clustering (k-means and E-M)

Gaussian mixture models



Left: Three normally distributed clusters $N(\boldsymbol{x}; \boldsymbol{\mu}_k, \Sigma_k), k = 1, 2, 3$

Right: Mixture distribution $p(\boldsymbol{x})$

How do we sample from such a distribution?

Probabilistic clustering (k-means and E-M)

Gaussian mixture models

Assumption: each data point is drawn from the MoG so it is assumed to have been generated by one of the Gaussians

... but we don't know which one!

Idea: there is a "theoretically complete" data set $(\boldsymbol{x}, \boldsymbol{z})$, where $\boldsymbol{z}$ indicates the true component of $\boldsymbol{x}$

Goal: use an algorithm that "completes" probabilistic data sets: this is what the **Expectation-Maximization** (E-M) family of algorithms do

---

Notice the analogies *clustering* = *MoG* and *cluster* = *component*

Probabilistic clustering (k-means and E-M)

Gaussian mixture models

Let us augment the $\boldsymbol{x}$ data with **latent** (unobserved) $\boldsymbol{z}$ data such that we have a **complete data** space $(\boldsymbol{x}, \boldsymbol{z})$:

1. We now have a joint distribution $p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})P(\boldsymbol{z})$

2. The $\boldsymbol{z} = (z_1, \ldots, z_K)^\mathsf{T}$ are such that only one $z_k = 1$ (the rest are 0)

3. The $\boldsymbol{z}$ vector represents the true membership of the corresponding $\boldsymbol{x}$ to every one of the K clusters

Probabilistic clustering (k-means and E-M)

Gaussian mixture models

The **marginal** distribution over $\boldsymbol{z}$ is:

$$P(z_k = 1) = \pi_k$$

therefore we may write

$$P(\boldsymbol{z}) = \prod_{k=1}^{K} (\pi_k)^{z_k}$$

Probabilistic clustering (k-means and E-M)

Gaussian mixture models

The **conditional** distribution of $\boldsymbol{x}$ given $\boldsymbol{z}$ is:

$$p(\boldsymbol{x}|z_k = 1) = N(\boldsymbol{x}; \boldsymbol{\mu}_k, \Sigma_k)$$

therefore we re-write the mixture as

$$p(\boldsymbol{x}) = \sum_{\boldsymbol{z}} p(\boldsymbol{x}|\boldsymbol{z})P(\boldsymbol{z}) = \sum_{k=1}^{K} p(\boldsymbol{x}|z_k = 1)P(z_k = 1) = \sum_{k=1}^{K} \pi_k N(\boldsymbol{x}; \boldsymbol{\mu}_k, \Sigma_k)$$

Probabilistic clustering (k-means and E-M)

Gaussian mixture models

The **conditional** distribution of $\boldsymbol{z}$ given $\boldsymbol{x}$ is:

$$\begin{aligned} P(z_k = 1|\boldsymbol{x}) &= \frac{p(\boldsymbol{x}|z_k = 1)P(z_k = 1)}{p(\boldsymbol{x})} \\ &= \frac{p(\boldsymbol{x}|z_k = 1)P(z_k = 1)}{\sum_{k=1}^{K} p(\boldsymbol{x}|z_k = 1)P(z_k = 1)} \\ &= \frac{\pi_k N(\boldsymbol{x}; \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(\boldsymbol{x}; \boldsymbol{\mu}_j, \Sigma_j)} \\ &=: \gamma_k(\boldsymbol{x}) \end{aligned}$$

which has a nice Bayesian interpretation

Probabilistic clustering (k-means and E-M)

Maximum likelihood

We have an i.i.d. sample of N *unlabelled* observations $D = \{\boldsymbol{x}_n\}_{n=1,\ldots,N}$, $\boldsymbol{x}_n \in \mathbb{R}^d$, which we want to **model** as a MoG of K components.

Let us maximize the log-likelihood for $\theta = \{\boldsymbol{\mu}_k, \Sigma_k, \pi_k\}$:

$$\begin{aligned} l(\theta) &:= \ln \mathcal{L}(\theta) = \ln \prod_{n=1}^{N} p(\boldsymbol{x}_n; \{\boldsymbol{\mu}_k, \Sigma_k, \pi_k\}) \\ &= \ln \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k N(\boldsymbol{x}_n; \boldsymbol{\mu}_k, \Sigma_k) \\ &= \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k N(\boldsymbol{x}_n; \boldsymbol{\mu}_k, \Sigma_k) \right\} \end{aligned}$$

This optimization problem is nontrivial, because the mixture log-likelihood surface can have many local maxima

Probabilistic clustering (k-means and E-M)

The E-M algorithm

- The E-M algorithm(*) is an iterative method for solving difficult likelihood problems in the presence of missing data
- In the mixture model context, the "missing data" are the latent labels z_k identifying the data component
- The E-M procedure guarantees convergence to a local maximum of the log-likelihood function (there is no guarantee of convergence to a global optimum)
- The procedure is often initialized from multiple randomly chosen initial conditions

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

Probabilistic clustering (k-means and E-M)

The E-M algorithm for Gaussian mixtures

$$\frac{\partial l(\theta)}{\partial \boldsymbol{\mu}_k} = 0$$

yields

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \gamma_k(\boldsymbol{x}_n)\boldsymbol{x}_n}{\sum_{n=1}^{N} \gamma_k(\boldsymbol{x}_n)} = \frac{\sum_{n=1}^{N} P(z_k = 1|\boldsymbol{x}_n)\boldsymbol{x}_n}{\sum_{n=1}^{N} P(z_k = 1|\boldsymbol{x}_n)}$$

---

the average of all $\boldsymbol{x}_n$, weighted by the posterior probability that each instance was generated by the k-th component

Probabilistic clustering (k-means and E-M)

The E-M algorithm for Gaussian mixtures

$$\frac{\partial l(\theta)}{\partial \Sigma_k} = 0$$

yields

$$\Sigma_k = \frac{\sum_{n=1}^{N} \gamma_k(\boldsymbol{x}_n)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\mathsf{T}}{\sum_{n=1}^{N} \gamma_k(\boldsymbol{x}_n)}$$

---

the sample covariance matrix of all $\boldsymbol{x}_n$, weighted by the posterior probability that each instance was generated by the k-th component

Probabilistic clustering (k-means and E-M)

The E-M algorithm for Gaussian mixtures

Maximizing now $l(\theta) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$

The conditions $\frac{\partial l(\theta)}{\partial \pi_k} = 0$ and $\sum_{k=1}^{K} \pi_k = 1$ yield:

$$\pi_k = \frac{1}{N} \sum_{n=1}^{N} \gamma_k(\boldsymbol{x}_n)$$

---

the average, for all $\boldsymbol{x}_n$, of the posterior probability that each was generated by the k-th component

Probabilistic clustering (k-means and E-M)

The E-M algorithm for Gaussian mixtures

- These update equations have the simple interpretation of being standard maximum likelihood estimates for mean, covariance and membership parameters, respectively, but where the data points are *weighted* by their membership probabilities

- The procedure is guaranteed to converge to a fixed point that need not be a global maximum and is a function of the initial conditions

- In practice, several different initial conditions can be tried (say, 10) and the maximum maximum likelihood among these selected

Probabilistic clustering (k-means and E-M)

The E-M algorithm for Gaussian mixtures

The procedure is often **initialized** by running k-means!

1. getting the $\boldsymbol{\mu}_k$ delivered by k-means

2. computing the sample Σ_k for each cluster

3. computing the π_k as the fraction of the $\boldsymbol{x}_n$ in each cluster

Convergence can be assessed by the changes in the $l(\theta)$

Probabilistic clustering (k-means and E-M)

The E-M algorithm for Gaussian mixtures

1. **Initialize** the $\{\boldsymbol{\mu}_k, \Sigma_k, \pi_k\}$ using k-means

2. **repeat**

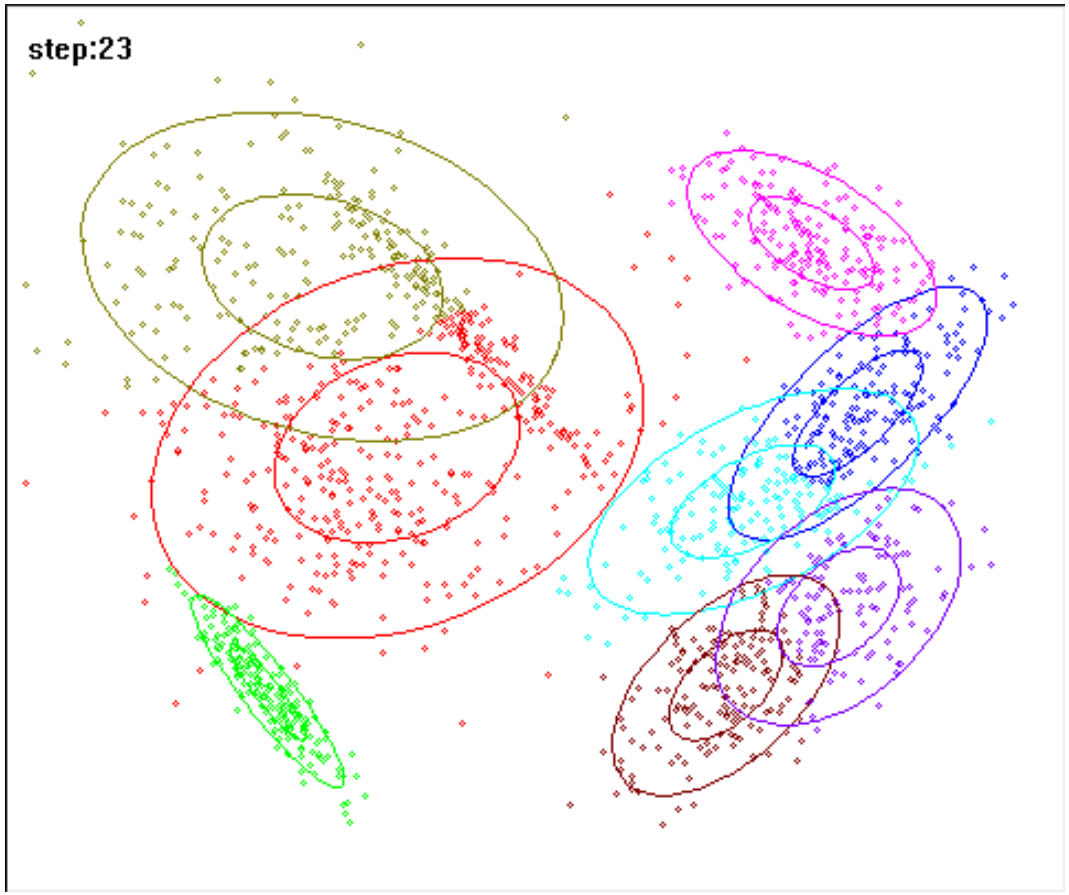
   E-step (re)compute the assignments $\gamma_k(\boldsymbol{x}_n)$

   M-step (re)compute the $\{\boldsymbol{\mu}_k, \Sigma_k, \pi_k\}$

3. **until** no further changes in $l(\theta)$ are detected

Probabilistic clustering (k-means and E-M)

The E-M algorithm for Gaussian mixtures



Nice E-M animation with $K = 8$

Probabilistic clustering (k-means and E-M)

The E-M algorithm for Gaussian mixtures

Notice that the E-M cluster assignments $\gamma_1(\boldsymbol{x}_n), \ldots, \gamma_K(\boldsymbol{x}_n)$ for a given $\boldsymbol{x}_n$ are "soft" (and sum to one)

It turns out that k-means is a degenerate case of E-M when $\Sigma_k = \sigma^2 I$

Indeed, $N(\boldsymbol{x}; \boldsymbol{\mu}_k, \sigma^2 I) = \dfrac{1}{\sigma(2\pi)^{d/2}} \exp\left(-\dfrac{\|\boldsymbol{x} - \boldsymbol{\mu}_k\|^2}{2\sigma^2}\right)$

It can be shown that, if we make $\sigma^2 \to 0$, then $\gamma_k(\boldsymbol{x}_n) \to r_{nk}$ and hence

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \gamma_k(\boldsymbol{x}_n)\boldsymbol{x}_n}{\sum_{n=1}^{N} \gamma_k(\boldsymbol{x}_n)} \to \frac{\sum_{n=1}^{N} r_{nk}\boldsymbol{x}_n}{\sum_{n=1}^{N} r_{nk}}$$

Probabilistic clustering (k-means and E-M)

Closing remarks: subjective considerations

- Why do we need clustering at all? What is the goal of clustering?
  - The same as any ML technique: finding regularities in data
  - One way of expressing regularity is to put a set of objects into groups that are similar to each other

- Benefits of a good clustering
  1. prediction
  2. lossy compression
  3. highlight interesting objects

Probabilistic clustering (k-means and E-M)

Closing remarks: conceptual considerations

- Where does the distance come from? What if we use a different distance between $\boldsymbol{x}$ and $\boldsymbol{\mu}$? How can we choose the 'best' distance?

- In mixture density modelling, the choice of distance corresponds to a choice of density; for example, standard Euclidean distance

$$d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2 = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$$

  corresponds to Gaussian clusters with $\Sigma_k = \sigma^2 I$ (yes, k-means)

- What distance corresponds to a MoG?

Probabilistic clustering (k-means and E-M)

Closing remarks: how to handle other spaces?

- In data analysis, there is a plethora of metrics for many data types; for example, when $\boldsymbol{x}, \boldsymbol{y}$ are presence/absence $(+/-)$ vectors:

$$d(\boldsymbol{x}, \boldsymbol{y}) = \frac{\#\{j / x_j \neq y_j\}}{\#\{j / x_j \neq - \wedge y_j \neq -\}} \qquad \text{Jaccard distance}$$

Example:

$$\boldsymbol{x} = + - + - - - - - - - - + - - -$$
$$\boldsymbol{y} = - - + - - - + - - - + - - -$$

then $d(\boldsymbol{x}, \boldsymbol{y}) = \frac{2}{4} = \frac{1}{2}$.

- We can also embed the data in an Euclidean space (using MCA, MDS, ...)

Machine Learning

Syllabus

1. Introduction to Machine Learning

2. Theoretical issues (I): regression

3. Linear regression and beyond

4. Theoretical issues (II): classification

5. Generative classifiers

6. Discriminative classifiers

7. Clustering
8. Learning with kernels (I): The SVM
9. Learning with kernels (II): Kernel functions
10. Learning with kernels (III): Other kernel methods
11. Artificial neural networks (I): the MLP
12. Artificial neural networks (II): the RBF
13. Ensemble methods: Random Forests
14. Advanced topics and frontiers