Cross-Entropy Clustering

Przemysław Spurek

Jagiellonian University

https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master

Table of Contents

Table of Contents

Motivation (GMM)

Clustering plays a basic role in many parts of data engineering, pattern recognition, and image analysis.

Motivation (GMM)

Clustering plays a basic role in many parts of data engineering, pattern recognition, and image analysis.

- Some of the most important clustering methods are based on GMM – Gaussian Mixture Model.

Motivation (GMM)

Clustering plays a basic role in many parts of data engineering, pattern recognition, and image analysis.

- Some of the most important clustering methods are based on GMM – Gaussian Mixture Model.
- Clustering based on density estimation techniques which use Expectation Maximization (EM) method,

Motivation (GMM)

Clustering plays a basic role in many parts of data engineering, pattern recognition, and image analysis.

- Some of the most important clustering methods are based on GMM – Gaussian Mixture Model.
- Clustering based on density estimation techniques which use Expectation Maximization (EM) method,

Cross-Entropy Clustering similar like EM is a general method for clustering.

Motivation (k-means)

Several of the most popular clustering methods are based on the k-means approach.

Motivation (k-means)

Several of the most popular clustering methods are based on the k-means approach.

- Although k-means is easily scalable, it has the tendency to divide the data into spherically shaped clusters of similar sizes. Consequently, it is not affine invariant and does not deal well with clusters of various sizes.

Motivation (k-means)

Several of the most popular clustering methods are based on the k-means approach.

- Although k-means is easily scalable, it has the tendency to divide the data into spherically shaped clusters of similar sizes. Consequently, it is not affine invariant and does not deal well with clusters of various sizes.
- Moreover, it does not change dynamically number of clusters, and therefore in order to efficiently apply k-means we needs use additional tools.

Motivation

The relation between the above two methods is well described V. Estivill-Castro and J. Yang in paper *Fast and robust general purpose clustering algorithms*:

"[...] The weaknesses of k-means results in poor quality clustering, and thus, more statistically sophisticated alternatives have been proposed.
[...] While these alternatives offer more statistical accuracy, robustness and less bias, they trade this for substantially more computational requirements and more detailed prior knowledge."

- `https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_01_mouse.R`

Motivation

The Cross–Entropy Clustering (CEC) approach joins the clustering advantages of k-means and EM.

Motivation

The Cross–Entropy Clustering (CEC) approach joins the clustering advantages of k-means and EM.

It occurs that CEC inherits the speed and scalability of k-means, while overcoming the ability of EM to use mixture models.

In particular, contrary to GMM, new models can easily be added without the need for complicated optimization.

Table of Contents

Cross-Entropy Clustering

Let it be recalled that in general EM aims to find

$$p_1, \dots, p_k \geq 0 : \sum_{i=1}^{k} p_i = 1, \tag{1}$$

and $f_1, \dots, f_k \in \mathcal{F}$, where $\mathcal{F}$ is a fixed (usually Gaussian) family of densities such that the convex combination

$$f := p_1 f_1 + \ldots + p_k f_k \tag{2}$$

optimally approximates the scattering of the data under consideration $X = \{x_1, \dots, x_n\}$.

Cross-Entropy Clustering

The optimization is taken with respect to an MLE based cost function

$$\mathrm{EM}(f, X) = -\frac{1}{|X|}\sum_{j=1}^{n} \ln\left(p_1 f_1(x_j) + \ldots + p_k f_k(x_j)\right), \qquad (3)$$

where $|X|$ denotes the cardinality of a set X.

Cross-Entropy Clustering

The optimization is taken with respect to an MLE based cost function

$$\mathrm{EM}(f, X) = -\frac{1}{|X|} \sum_{j=1}^{n} \ln \left(p_1 f_1(x_j) + \ldots + p_k f_k(x_j) \right), \tag{3}$$

where $|X|$ denotes the cardinality of a set X.
The optimization in EM consists of the Expectation and Maximization steps.

- While the Expectation step is relatively simple,
- the Maximization usually (except for the simplest case when the family F denotes all Gaussian densities) needs a complicated numerical optimization.

Cross-Entropy Clustering

The goal of CEC is similar, i.e. aims at minimizing the cost function (which is a small modification of that given in (3) by substituting the sum with a maximum):

$$\mathrm{CEC}(f, X) := -\frac{1}{|X|}\sum_{j=1}^{n} \ln\left(\max(p_1 f_1(x_j), \ldots, p_k f_k(x_j))\right), \qquad (4)$$

where all p_i for $i = 1, \ldots, k$ satisfy the condition (1):

$$\mathrm{EM}(f, X) = -\frac{1}{|X|}\sum_{j=1}^{n} \ln\left(p_1 f_1(x_j) + \ldots + p_k f_k(x_j)\right).$$

- `https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_02_1D.R`

Cross-Entropy Clustering

Now we have to answer two main questions:

- How to minimize the function?
- Why we cold the method Cross-Entropy Clustering?

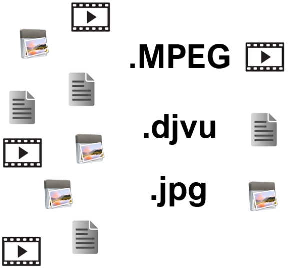
Normal distribution

The core idea of our method comes from information theory and is based on the observation, that in data compression it is often profitable to use various coding algorithms which compress different types of data.

Normal distribution

The core idea of our method comes from information theory and is based on the observation, that in data compression it is often profitable to use various coding algorithms which compress different types of data.

Normal distribution



Dual reasoning leads to the gathering in one cluster those data which is coded by the same algorithm.

Differential entropy

Let us assume that message $X = (x1, \ldots, x_N)$ and the compression methods $w_1, \ldots, w_K \in W$ are given. We denote the algorithm that encodes point x_l (defines the cluster it belongs to) with w_{kl}, where $k_l \in 1, ..., K$. Then the memory cost of coding the message X equals

$$\sum_{i=1}^{K} (\text{cost of identification of } k_l +$$

$$+\text{the amount of memory algorithm } w_{kl} \text{ uses to code } x_l).$$

Differential entropy

Consequently we can explain the cost function of CEC.

To do so, let it be recalled that by the *cross-entropy of data set X with respect to density f* is given by

$$H^{\times}(X\|f) = -\frac{1}{|X|}\sum_{\mathrm{x}\in X} \ln(f(\mathrm{x})).$$

If we want to code X by the code optimized for random variable Y with density f we obtain the cross-entropy, which describe approximately how many bits we need to code element from X.

Cross-Entropy Clustering

Summarizing, given the density families $\mathcal{F}_1, \ldots, \mathcal{F}_n$, the goal of the CEC algorithm is to divide the data-set X into k (possibly empty) clusters $X_1, \ldots, X_k$ such that the value of the function

$$\mathrm{CEC}(X_1, \mathcal{F}_1; \ldots; X_k, \mathcal{F}_k) = \sum_{i=1}^{k} p_i \cdot (-\ln(p_i) + H^{\times}(X_i \| \mathcal{F}_i)),$$

$$\text{where } p_i = \frac{|X_i|}{|X|} \tag{5}$$

is minimal.

We use notation

$$H^{\times}(X \| \mathcal{F}) := \inf_{f \in \mathcal{F}} H^{\times}(X \| f),$$

for density family $\mathcal{F}$.

Cross-Entropy Clustering

CEC allows an automatic reduction of “unnecessary” clusters, since, contrary to the case of classical k-means and EM, there is a cost of using each cluster.

- `https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_03_interactive.R`

Table of Contents

Input

number of clusters $k > 0$

initial conditions

obtain initial clustering $X_1, \dots, X_k$

obtain probabilities $p_i = \frac{|X_i|}{|X|}$, covariance matrix Σ_i and mean m_i for $i \in \{1, \dots, k\}$

repeat

obtain new clustering $X_1, \dots, X_k$ by matching elements to the cluster such that

$$-\ln(p_i) - \ln(N(m_i, \Sigma_i))$$

is minimal

obtain new probabilities p_i, covariance matrix Σ_i and mean m_i for $i \in \{1, \dots, k\}$

until No change

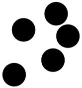
Table of Contents

1. $\mathcal{G}_\Sigma$ – Gaussian densities with covariance Σ. The clustering will have the tendency to divide the data into clusters resembling balls with respect to the Mahalanobis distance $\|\cdot\|_\Sigma$.

$$\Sigma_{\mathcal{G}_\Sigma}(X) = \Sigma$$

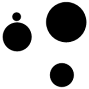
$$H^\times(\mathrm{y}\|\mathcal{G}_\Sigma) = \frac{N}{2}\ln(2\pi) + \frac{1}{2}\mathrm{tr}(\Sigma^{-1}\Sigma_X) + \frac{1}{2}\ln\det(\Sigma)$$

2. $\mathcal{G}_{r\mathrm{I}}$ – subfamily of $\mathcal{G}_{\Sigma}$, for $\Sigma = r\mathrm{I}$ and $r > 0$ is fixed, which consists of the spherical (radial Gaussian) with covariance matrix $r\mathrm{I}$ (the clustering will have tendency to divide the data into balls with fixed radius proportional to $\sqrt{r}$).



$$\Sigma_{\mathcal{G}_{r\mathrm{I}}}(X) = r\mathrm{I}$$
$$H^{\times}(X\|\mathcal{G}_{r\mathrm{I}}) = \frac{N}{2}\ln(2\pi) + \frac{N}{2}\ln(r) + \frac{1}{2r}\mathrm{tr}(\Sigma_X)$$

3. $\mathcal{G}_{(\cdot I)}$ – spherical (radial) Gaussian densities meaning those Gaussians for which the covariance is proportional to identity. The clustering will try to divide the data into balls of arbitrary sizes.



$$\Sigma_{\mathcal{G}_{(\cdot I)}}(X) = \frac{\mathrm{tr}(\Sigma_X)}{N}\mathrm{I}$$

$$H^{\times}(X \| \mathcal{G}_{(\cdot I)}) = \frac{N}{2}\ln(2\pi e/N) + \frac{N}{2}\ln(\mathrm{tr}\Sigma_X)$$

4. $\mathcal{G}_{\mathrm{diag}}$ – Gaussians with diagonal covariance. The clustering will try to divide the data into ellipsoids with radii parallel to coordinate axes.

$$\Sigma_{\mathcal{G}_{\mathrm{diag}}}(X) = \mathrm{diag}(\Sigma_X)$$
$$H^{\times}(X \| \mathcal{G}_{\mathrm{diag}}) = \frac{N}{2}\ln(2\pi e) + \frac{1}{2}\ln(\det(\mathrm{diag}(\Sigma_X)))$$

5. $\mathcal{G}_{\lambda_1,\ldots,\lambda_N}$ – Gaussian densities with the covariance matrix having eigenvalues $\lambda_1, \ldots, \lambda_N$ such that $\lambda_1 \leq \ldots \leq \lambda_N$. The clustering will try to divide the data into ellipsoids with fixed shape rotated by an arbitrary angle.

$$\Sigma_{\mathcal{G}}(X) = \Sigma_{\lambda_1,\ldots,\lambda_N}$$

$$H^{\times}(X \| \mathcal{G}_{\lambda_1,\cdots,\lambda_N}) = \frac{N}{2}\ln(2\pi) + \frac{1}{2}\sum_{i=1}^{N}\frac{\lambda_i^X}{\lambda_i} + \frac{1}{2}\ln\left(\prod_{i=1}^{N}\lambda_i\right)$$

6. $\mathcal{G}$ – all Gaussian densities. In this case the dataset is divided into ellipsoid-like clusters without any preferences concerning the size or the shape of the ellipsoid.

$$\Sigma_{\mathcal{G}}(X) = \Sigma_X$$
$$H^{\times}(X\|\mathcal{G}) = \frac{N}{2}\ln(2\pi e) + \frac{1}{2}\ln\det(\Sigma_X)$$

Table of Contents

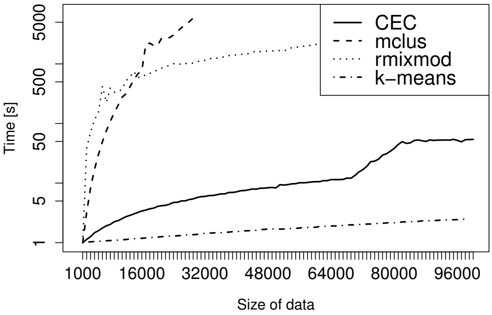
CEC: Cross-Entropy Clustering

Cross-Entropy Clustering (CEC) divides the data into Gaussian type clusters.

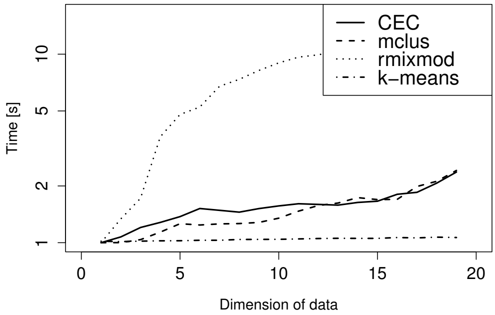
Version:	0.9.4
Imports:	graphics, methods, stats, utils
Published:	2016-04-24
Author:	Konrad Kamieniecki [aut, cre], Przemyslaw Spurek [ctb]
Maintainer:	Konrad Kamieniecki <konrad.kamieniecki at uj.edu.pl>
License:	GPL-3
URL:	https://github.com/azureblue/cec
NeedsCompilation:	yes
Materials:	README NEWS
In views:	Cluster
CRAN checks:	CEC results

- https://github.com/azureblue/cec
- https://cran.r-project.org/web/packages/CEC/index.html
- https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_04_all_models.R
- https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_06_R_d.R

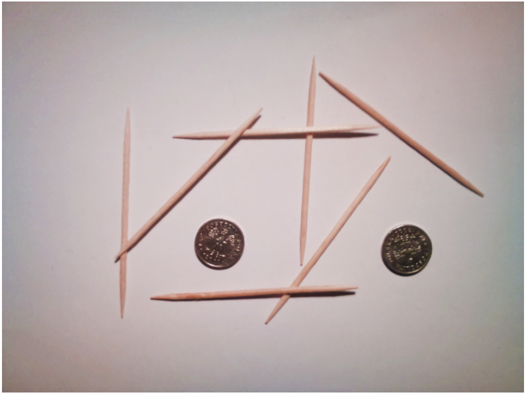
CEC method gives better results than **mclust** and **Rmixmod** for large data sets. In the case of high dimensional data CEC gives comparable results to the **mclust** and better than the **Rmixmod**.



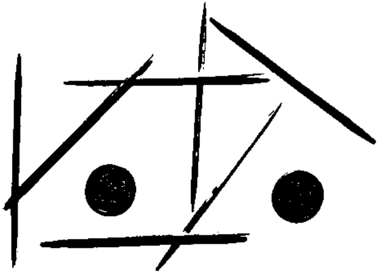
CEC method gives better results than **mclust** and **Rmixmod** for large data sets. In the case of high dimensional data CEC gives comparable results to the **mclust** and better than the **Rmixmod**.



One of the most powerful properties of the CEC algorithm is the possibility of mixing models.



One of the most powerful properties of the CEC algorithm is the possibility of mixing models.



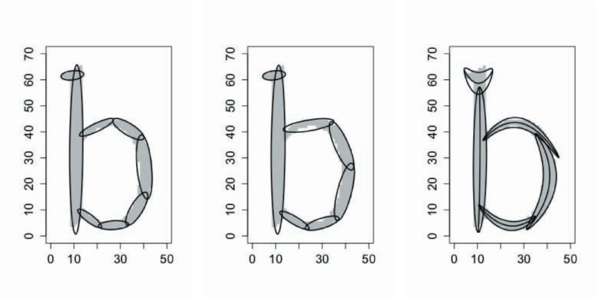
- https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_05_mixed.R

Table of Contents

The growing need for more flexible tools to analyze datasets that exhibit nonnormal features, including asymmetry, multimodality, and heavy tails, has led to intense development of non-normal model-based methods.

The growing need for more flexible tools to analyze datasets that exhibit nonnormal features, including asymmetry, multimodality, and heavy tails, has led to intense development of non-normal model-based methods.

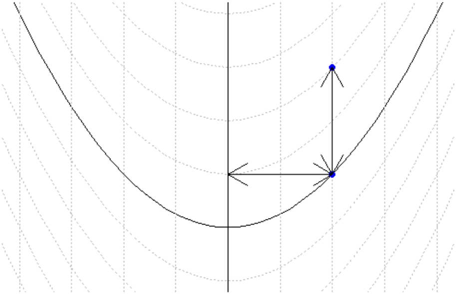
Gaussian Mixture Models (GMM) have many applications in density estimation and data clustering. However, the models do not adapt well to curved and strongly nonlinear data, since many Gaussian components are typically needed to appropriately fit the data that lie around the nonlinear manifold.

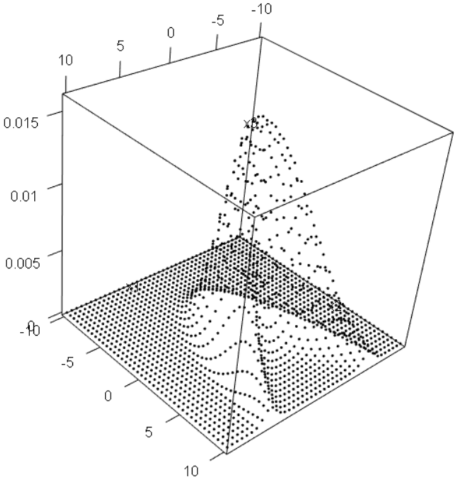


Fitting a b-type set by using (a) GMM, (b) CEC, (c) afCEC.

In the case of a “curvilinear” coordinate system, we adapt the Gaussian density to the arbitrary given function $f \in C(\mathbb{R}, \mathbb{R})$. For each point, we use the Euclidean distance along the second coordinate to curve f

$$N(\mathrm{m}, \Sigma, f)([x_1, x_2]^T) = N(m_1, \sigma_1^2)(x_1) \cdot N(m_2, \sigma_2^2)(x_2 - f(x_1)). \tag{6}$$





-10
-5
0
5
10
0.015
0.01
0.005
0
-10
-5
0
5
10
x1
x2
x3

- https://github.com/GeigenPrinzipal/afCEC
- https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_07_afCEC_fire.R
- https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_08_afCEC_dog.R
- https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_09_afCEC_airplane.R
- https://github.com/przem85/Cross_Entropy_Clustering_presentation/tree/master/example_10_afCEC_ship.R

Thank you for your attention.