CLUSTERING

VERA LEZIONE 12
24/11/19

Given a dataset D with n rows (INSTANCES) and d columns (FEATURES), we want to partition D into g sub data matrices $D_1, D_2, \dots, D_g$ such that:

i) Instances belonging to the same sub-matrix are "similar";

ii) Instances belonging to different sub-matrices are "not similar".

OSS: In clustering we don't use the class feature.

It may only be used for evaluating the result obtained by the clustering algorithm.

If $g := \#$ of clusters is known, then we have PARTITIONAL CLUSTERING ALGORITHMS

If g is not known we have to find the optimal value of g.

There are two possible approaches to solve CLUSTERING:

1) PARTITIONING ALGORITHM

We can start with 1 cluster conteining ll the instences of the detaset end then we partition it in smaller clusters.

2) AGGLOMERATIVE ALGORITHM,

We start with n clusters, one for each instence, end merge them as needed.

COOL CAT ~ (08:00/1)

Uses the concept of entropy.

Based on the CLUSTERING ENTROPY, defined as,

$$CE := \frac{1}{n} \sum_{k=1}^{g} \left(\overset{\#D_k}{n_k}\right) \cdot H(D_k)$$

where,

$$H(D_k) := \sum_i P(x_i^k) \cdot \log\left(\frac{1}{P(x_i^k)}\right)$$

$$x_i := i\text{-th row of } D$$

The basic idea is to minimize the entropy of the various clusters. Since the entropy is a measure of dis-similarity, by minimizing it we are trying to maximize the similarity in the various clusters.

The COOLCAT algorithm works by Knowing k in two phases:

- INITIALIZATION PHASE

Selects k instances that are dis-similar with respect to the d-dimensional distance.

- SECOND PHASE

For all $n-k$ remaining instances do the following:

Select a remaining instance i.

Add i to D_1 and compute CE_1

Add i to D_2 and compute CE_2

$\vdots$

Add i to D_k and compute CE_k

Finally, add i to the cluster j that minimizes CE_j.

MAXIMUM LIKELIHOOD METHOD ~ (19:00/1)

Statistical method used to estimate the parameters of a probability distribution.

Consider X discrete r.v. with p.m.f. $P_X(x|\theta)$, where θ is the parameter we want to estimate.

In M.L. methods we estimate the value of ϑ with the value that maximizes the LIKELIHOOD of ϑ, which is defined as:

$$L(\theta) := \prod_{i=1}^{n} P_X(x_i|\vartheta) \quad \text{(DISCRETE CASE)}$$

$$L(\theta) := \prod_{i=1}^{n} f_X(x_i|\theta) \quad \text{(CONTINUOUS CASE)}$$

where $\{x_1, \dots, x_n\}$ form an i.i.d. sample with pmf $P_X(x|\theta)$.

Intuitively, the likelihood of ϑ given a sample $\{x_1, \dots, x_n\}$ is the probability that ϑ is the parameter of the data given we have observed $x_1, \dots, x_n$.

The M.L. method tries to find

$$\hat{\theta} := \underset{\vartheta}{\operatorname{argmax}} \; L(\theta)$$

That is, the value of ϑ that maximizes the probability of observing $x_1, \dots, x_n$.

EXAMPLE OF M.L.

If we assume $X \sim N(\mu, \sigma^2)$, that is

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

then we are interested in estimating

$$\hat{\theta} = (\hat{\mu}, \hat{\sigma}^2)$$

It can be proved that

$$\hat{u} = \frac{1}{n} \sum_{i} x_i \quad \left(\text{SAMPLE MEAN} \right)$$

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i} (x_i - \hat{u}) \quad \left(\text{SAMPLE VARIANCE} \right)$$

MIXTURE

In a previous python example had generated a dataset by mixing observation generated by two different normal distribution.

Whenever a sample is obtained by mixing different pdfs, we talk about mixtures, since the pdf of the sample is a mix of various pdfs.

If we mix different gaussians, then we have a Gaussian mixture.

If we mix g different multivariated Gaussian, each of which has

$\underline{\mu}_K$, mean vector of $\underline{x}_K$

$\underline{\Sigma}_K$, covariance matrix of $\underline{x}_K$

then we have g pdf having a multivariate gaussian density:

$$f_K(\underline{x}) = g(\underline{x} \mid \underline{\mu}_K, \underline{\Sigma}_K)$$

$$= \frac{1}{\sqrt{2\pi \det(\underline{\Sigma}_K)}} \cdot e^{-\left(\frac{(\underline{x}-\underline{\mu}_K)^T \cdot \underline{\Sigma}_K^{-1} (\underline{x}-\underline{\mu}_K)}{2}\right)}$$

In this setting, our dataset is no longer generated by a single pdf. Given an instance, we don't know which pdf was used to generate that instance.

The problem of finding the pdf used to generate an instance can be formalized as a CLUSTERING or CLASSIFICATION problem.

We then introduce a latent discrete r.v.

↳ NOT OBSERVABLE

This r.v. is useful to model a dataset generated with a Gaussian mixture.

In particular we define it as follows

$$z := \text{index of pdf used to generate the instance}$$

With z we can thus model the belonging of each instance to a particular pdf.

If $z = 1$, then the instance was generated from $g_1(\underline{x})$.

Using z we can write the pdf of any instance of the dataset as follows

$$f(\underline{x}) = \sum_{k=1}^{g} P_z(k) \cdot f_k(\underline{x})$$

$$= \sum_{k=1}^{g} P_z(k) \cdot f(\underline{x} \mid \underline{\mu}_k, \underline{\Sigma}_k)$$

Notice that the set of parameters for this pdf is

$$\underline{\theta} = [\underline{\mu}_1, \underline{\Sigma}_1, P_z(1), \dots, \underline{\mu}_g, \underline{\Sigma}_g, P_z(g)]$$

which means that we need to estimate many scalar values?

<u>Q</u> : How many co-ordinates does $\underline{\mu}_1$ have? d? Therefore don't we need to estimate

$$d \cdot g + g + d^2 \cdot g \text{ scalar values.}$$

Let us now define the likelihood function

$$L(\underline{\theta}) = P(D \mid \underline{\theta}) =$$ probability of observing the whole dataset given the set of parameters.

In our case we have that

$$L(\theta) = P(D \mid \underline{\theta})$$

$$= \prod_{i=1}^{n} f(x_i)$$

↳ i-th row of dataset

WHY USING M.L.? ~ (13:00/4)

If our dataset is generated from a Gaussian Mixture, then the best clustering is obtained by estimating the value of the latent variable z.

The M.L. method can be used as a statistical method for evaluating the "goodness" of a set of clusters.

In particular, if after a step the likelihood is larger, then the set of clusters is better than the previous set of clusters.

EVALUATING A CLUSTERING ALGORITHM

~ (03:00/5)

We can use the class label to evaluate the performance of a clustering process.

In particular, after we constructed our clusters, we can analyze the class features of all the instances that were grouped inside the same cluster.

The class feature is considered an EXTERNAL MEASURE.

We can measure the performance of a cluster algorithm also by using INTERNAL MEASURES. Indeed, given

K := index of cluster

J := index of the instance
in the cluster

We can define, for each cluster, the mean and the variance

$$\mu_K := \frac{1}{n_K} \cdot \sum_{j}' x_j^K$$

$$\sigma_K^2 := \frac{1}{n_K} \cdot \sum_{j}' \left(x_j^K - \mu_K \right)^2$$

The idea is to compute the SPREAD of the cluster, that is

$$\partial_K := \sqrt{\frac{1}{n_K} \cdot \sum_{j}' \underset{\uparrow}{\delta} \left(x_j^K , \mu_K \right)}$$

DISTANCE - FUNCTION
(EUCLIDEAN DISTANCE)

We can then use the spread to measure the goodness of a set of clusters. One of the most common INTERNAL MEASURE is the DAVES BOULDIM INDEX (DBI), defined as

$$DBI := \frac{1}{g} \sum_{K=1}^{g}{}' \max_{j, i \neq K} \frac{\partial_i + \partial_K}{\delta \left(\underline{\mu}_j , \underline{\mu}_i \right)}$$

Small DBI ⟷ Good clustering algorithm

A good clustering algorithm is one that

i) minimizes the spread in every cluster.

ii) the distance between the centers of different clusters should be large.

As we can see, the basic block of clustering algorithm is a metric that computes the goodness of a set of clusters. We have seen the following performance metrics:

1) Entropy - based.
2) M.I. - based.
3) DBI - based.

There are more than these three performance metric.

For a given performance metric there are several iterative algorithms that can be used to build the clusters.