Aprendizagem Automática

Assignment 2 - Clustering

Andrea Mascaretti N52222 Daniel Pimenta N45404

28th November 2017

Abstract

TODO

1 Introduction

2 Clustering Algorithms

2.1 K-Means Algorithm

In our exposition of the K-Means Algorithm we rely heavily on [HTF01] and [JW07]. Suppose that we have a set of n observations, indexed by $I = \{1, ..., n\}$, of some quantitative variables

$$C = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$$

where

$$\boldsymbol{x}_i \in \mathbb{R}^p, \ \forall i \in I$$

We also have a set of labels (at this stage we will consider their number as given and such that $card(\mathcal{L}) = k, \ k \in [1, n] \cap \mathbb{N}$)

$$\mathcal{L} = \{l_1, \ldots, l_k\}, \ K := \{1, \ldots, k\}$$

Our goal is to construct a (*measurable*) function to associate each and every element of C to one label in $\mathcal{L}$. It is easy to see that it makes no difference to work with $\mathcal{L}$ or K as we can always find a bijection between the two sets.

We have reasons to believe that the elements belonging to one particular group are somewhat more similar to each other than they are to members of other groups and that one element can belong to one group and one group only. We also hypothesise that there are no empty groups. Translating this into mathematical terms, our aim is to find a collection

$$\{C_i\}_{i \in K}$$

such that

$$C_i \cap C_j = \emptyset, \ i \neq j$$

and

$$\bigcup_{i=1}^{k} C_i = C$$

or, in other words, we want to partition C into k clusters.

We also want to define a function

$$\delta : \ \boldsymbol{x}_i \mapsto l_i$$

such that

$$\forall i \in K, \ \delta^{-1}(l_i) \neq \emptyset.$$

One immediate idea is to consider the following:

$$\delta(\boldsymbol{x}) = \sum_{i=1}^{k} \left(\mathcal{I}_{C_i}(\boldsymbol{x}) \cdot i\right)$$

where

$$\mathcal{I}_A(\underline{x}) = \begin{cases} 1 & \boldsymbol{x} \in A \\ 0 & \boldsymbol{x} \notin A \end{cases}$$

so that we have a function that takes as an input a value from C and gives as an output the index of the label we are considering (and we know they are finite).

The main issue is to find a method to partition the set C into the clusters. First of all, we need to define in a more precise manner the meaning of similarity: we will consider the squared *Euclidean distance* for this purpose

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{k=1}^{p} (x_{ik} - x_{jk})^2 .$$

We now need to define some *loss* function and try to minimise it. The most natural idea follows from the assumption that members of one given cluster will be closer to each other than to member of other clusters: this is equivalent to stating that the correct partitioning will minimise the variability *within* the clusters. We need to translate this key concept into an objective function.

We define

$$I_k := \{i \in I \; s.t. \; \boldsymbol{x}_i \in C_k\}$$

and in this case we will have

$$W(C_k) = \frac{1}{card(C_k)} \sum_{i,j \in I_k} \sum_{l=1}^{p} (x_{il} - x_{jl})^2$$

and the problem becomes as follows [1]

$$\min_{C_1,\ldots,C_k} \left\{ \sum_{k=1}^{K} W(C_k) \right\}$$

Now, we want to find a way to solve this problem and here we find the first issue as there are almost K^n ways to partition n observations into K clusters. We shall therefore look for a local optimum as follows.

2.2 Algorithm for K-means Clustering

1. Randomly assign a number, from 1 to K, to each observation: they will serve as initial cluster assignments. Set $\mu := 0$
2. Iterate what follows until termination criterion holds:
   (a) For each cluster, compute the *centroid*
   $$\boldsymbol{c}_i^{\mu} = \frac{1}{card(C_i^{\mu})} \sum_{j \in I_i^{\mu}} \boldsymbol{x}_i,$$
   computed as the mean vector of the observations in the k-th cluster at μ-th iteration
   (b) Assign each observation to the cluster whose centroid is closest (with respect to the Euclidean distance) as follows
   $$\forall j \in I, \; i^* = \min_{i \in K} \left\{ d(\underline{x}_j, \underline{c}_i^{\mu}) \right\} = \min \left\{ d(\underline{x}_j, \underline{c}_1^{\mu}), \ldots, d(\underline{x}_j, \underline{c}_k^{\mu}) \right\} \implies \underline{x}_j \in C_{i^*}^{\mu}$$
   (c) Set $\mu := \mu + 1$
3. Termination occurs when cluster assignments stop changing

[1] It is important to notice that this function surely decreases for increasing k and in the extreme case of $k = n$ we have that each cluster contains only one point and we have no variability whatsoever. It is important to understand that we are looking for classes that do have some internal variability.

2.3 Some comments regarding K-Means

The algorithm we have just defined is descent, that is it improves the solution at each iteration so that when the termination condition is true, we are certain to have reached a *local optimum*. It is therefore a good practice that of iterating the algorithm various times and pick the best solution.

There is also one last important point to state. Any time we perform the K-means methods, we will find k clusters on our data. Now, the question is: are we really separating actual subgroups or are we just *clustering the noise*? Moreover, suppose that we are dealing with a case in which the vast majority of data belongs to a small number of subgroups, but for a small subset that is quite different from the rest. As our method *forces* our data into a fixed number of clusters, the presence of an outlying class might bring to distorted results. So it is good practice not only to try various values of k to see what happens, but also to try and apply to algorithm to subsets of our initial set to check that results remain somewhat stable.

COMPARISON WITH OTHER CLUSTERING METHODS

References

[HTF01] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[JW07] Richard A. Johnson and Dean W. Wichern. *Applied multivariate statistical analysis*. Pearson Prentice Hall, Upper Saddle River, NJ, sixth edition, 2007.