

# UNSUPERVISED STATIC DISCRETIZATION METHODS IN DATA MINING

**Daniela Joița**

Titu Maiorescu University, Bucharest, Romania

[daniela.joita@utm.ro](mailto:daniela.joita@utm.ro)

**Abstract.** Discretization of real-valued data is often used as a pre-processing step in many data mining algorithms. In this paper we review some important unsupervised discretization methods among which there are the discretization methods based on clustering. We propose a discretization method based on the k-means clustering algorithm which avoids the  $O(n \log n)$  time requirement for sorting.

**Keywords:** discretization, clustering, k-means

## 1. INTRODUCTION

Many data mining techniques often require that the attributes of the data sets are discrete. If this is not the case, one has either to choose a different algorithm or to find a way to discretize the continuous data attributes prior to applying the desired algorithm. Given that most of the experimental data are continuous, not discrete, the discretization of the continuous attributes is an important issue. At the same time, some machine learning algorithms that can handle both continuous and discrete attributes perform better with the discrete-valued attributes.

Discretization techniques are often used by the classification algorithms but their applications are not restricted to them. Discretization is also used by genetic algorithms, instance-based learning.

This paper is organized as follows. In the next section we present the definition of discretization of a continuous attribute. In Section 3 we present the classifications of discretization methods as they appear in the speciality literature and some important unsupervised discretization methods: equal-width interval, equal-frequency and discretization methods based on clustering. In Section 4 we describe a proposed discretization method based on the k-means clustering algorithm. Finally, Section 5 summarizes our contributions and discusses future work.

## 2. DISCRETIZATION

An attribute is discrete if it has a relatively small (finite) number of possible values while a continuous attribute is considered to have a very large number of possible values (infinite) [2]. In other words, a discrete data attribute can be seen as a function whose range is a finite set while a continuous data attribute as a function whose range is an infinite totally ordered set, usually an interval. For example, the temperature, the humidity are continuous attributes of a database, while the color of eyes, the day of the week or the quality of wine are discrete attributes.

The goal of discretization is to reduce the number of possible values a continuous attribute takes by partitioning them into a number of intervals [2]. Let give a definition of discretization [1].

Let  $A$  be an attribute of a finite data set  $D$ . Let  $n$  be the number of examples in  $D$ . We denote by  $adom(A)$  the set of all values of the attribute  $A$  in the data set  $D$ , called the *active domain of  $A$*  and by  $a = (a_1, a_2, \dots, a_n)$  the vector of all values of the attribute  $A$  for all  $n$  examples. To discretize the numeric attribute  $A$  means to find a partition of  $adom(A)$ . This implies to determine the cut points  $t_0, t_1, \dots, t_k$  with  $t_0 < t_1 < \dots < t_k$  such that the set  $\{P_1, P_2, \dots, P_k\}$  forms a partition of  $adom(A)$ , where  $P_i$  is defined by  $P_i = \{a \in adom(A) : t_{i-1} \leq a < t_i\}$  for  $i = \overline{0, k-1}$  and  $P_k = \{a \in adom(A) : t_{k-1} \leq a \leq t_k\}$  and  $t_0 = \min adom(A)$  and  $t_k = \max adom(A)$ .

Then the attribute  $A$  is replaced by the discretized attribute  $A^{disc}$  whose values are defined as follows:

$$A^{disc} = (a_1^{disc}, a_2^{disc}, \dots, a_n^{disc}), \quad a_j^{disc} = i \text{ iff } a_j \in P_i \text{ for } j = \overline{1, n}.$$

Therefore each value of the attribute  $A$  which falls in  $P_i$  is replaced by  $i$ .

### 3. DISCRETIZATION METHODS

There is a large variety of discretization methods. Dougherty et al. (1995) [3] present a systematic survey of all the discretization method developed by that time. They also make a first classification of discretization methods based on three directions: global vs. local, supervised vs. unsupervised, static vs. dynamic. Discretization methods are local or global. Local methods, like the decision tree learners, produce partitions that are applied to localized regions of the instance space, during the machine learning process. Global methods partition each attribute into regions independent of the other attributes prior to the learning process and produce a global mesh over the entire continuous instance space [3]. Discretization methods can be unsupervised or supervised. While the unsupervised methods are independent of the target attribute, the supervised ones make use intensively of the target attribute [2]. Discretization methods can be static or dynamic. The static attribute discretization is performed on one attribute at a time, not considering the other attributes. The static discretization is repeated for the other continuous attributes as many times as it is needed. On the contrary, the dynamic discretization method discretizes all attributes at the same time [2].

Usually, the discretization process consists of two steps [2]. First, the number of discrete intervals needs to be chosen. Even though there are discretization methods which determine this number in the discretization process, this step is done usually by the user either by some heuristic techniques or by running the discretization technique for different number of intervals and deciding what is the best choice by using a criterion. Second, the cut points must be determined, which is often done by a discretization algorithm itself.

In [3] five discretization methods were compared: two unsupervised global methods (equal width and equal frequency interval), two supervised global methods (1RD (Holte 1993) and Fayyad & Irani's (1993) entropy minimisation), and a supervised local method (the classification algorithm C4.5). The methods were tested on 16 data sets from the UC Irvine Machine Learning Database Repository. They found that supervised techniques are more accurate when are applied to algorithms like classification, because the class information is taken into account. On the other hand, the unsupervised techniques could be considerably faster. Surprisingly, no method produced the highest accuracy for all data sets. Fayyad & Irani's method achieved the best overall results.

Next we will concentrate only on unsupervised static discretization methods.

#### 3.1 Unsupervised discretization methods

Among the unsupervised discretization methods there are the simple ones (equal-width and equal-frequency interval binning) and the more sophisticated ones, based on the clustering analysis, such as k-means discretization.

##### Equal-width interval discretization

In this case, the domain  $dom(A)$  is divided in  $k$  intervals of equal width determined by  $h = \frac{a_{\max} - a_{\min}}{k}$  where  $a_{\max} = \max\{a_1, a_2, \dots, a_n\}$  and  $a_{\min} = \min\{a_1, a_2, \dots, a_n\}$ . The  $k+1$  cut points are  $a_{\min}, a_{\min} + h, \dots, a_{\min} + kh = a_{\max}$ . The limitations of this method are given by the uneven distribution of the data points: some intervals may contain much more data points than other.

##### Equal-frequency interval discretization

This algorithm tries to overcome the limitations of the equal-width interval discretization by dividing the domain in intervals with the same distribution of data points. It determines the minimum and maximum values of the attribute, as the equal-width interval discretization, sorts all values in increasing order, and divides the interval  $[a_{\min}, a_{\max}]$  into  $k$  intervals, such that every interval contains the same number  $\left(\frac{n}{k}\right)$  of sorted values, possible duplicated. It has been found that equal-frequency interval discretization in conjunction with the Naïve Bayes learning scheme can give excellent results [10]. This method is called *proportional k-interval discretization*.

##### Discretization methods based on clustering

Other discretization methods are based on clustering. The purpose of clustering is to search for similar examples and group them into clusters such that the distance between examples within cluster is as small as possible and the distance between clusters is as large as possible.

The problem of choosing the right number of clusters is very important for all the clustering methods. In practice, usually one runs the clustering algorithm for several different number of clusters and finds the "best" number based on some measure of "goodness" of clustering.

The *k-means clustering method* remains one of the most popular clustering method. This algorithm has been identified by the IEEE International Conference on Data Mining (ICDM) in December 2006 as a top 10 algorithm, being considered among the most influential data mining algorithms in the research community[11]. The algorithm has been discovered by several researchers across different disciplines, among which, most notably mentioned in the above survey paper, Lloyd (1957, 1982) , Forgy (1965) [5], Friedman and Rubin (1967), and MacQueen (1967) [9].

The k-means clustering discretization method can be seen either as a supervised discretization method[2], when the clustering is performed for all attribute values for each value of the target attribute or as an unsupervised discretization method[3] when one discretizes the attribute without taking in account the values of the target attribute (classes).

The k-means clustering algorithm operates on a set of data points and assumes that the number of clusters to be determined ( $k$ ) is given. Initially, the algorithm assigns randomly  $k$  data points to be the so called *centers* (or *centroids*) of the clusters. Then each data point of the given set is associated to the closest center resulting the initial distribution of the clusters. After this initial step, the next two steps are performed until the convergence is obtained:

1. Recompute the centers of the clusters as the average of all values in each cluster.
2. Each data point is assigned to the closest center. The clusters are formed again.

The algorithm stops when there is no data point that needs to be reassigned or the number of data points reassignments is less than a given small number.

The convergence of the iterative scheme is guaranteed in a finite number of iterations but the convergence is only to a local optimum, depending very much on the choice of the initial centers of the clusters.

Other types of clustering methods can also be used as the baselines for the designing discretization methods, for examples *hierarchical clustering* methods. As opposed to the k-means clustering method which is an iterative method, the hierarchical methods can be either divisive or agglomerative. Divisive methods start with a single cluster that contains all the data points. This cluster is then divided successively into as many clusters as needed. Agglomerative methods start by creating a cluster for each data point. These clusters are then merged, two clusters at a time, by a sequence of steps until the desired number of clusters is obtained. Both approaches involve design problems: which cluster to divide/which clusters to merge and what is the right number of clusters. After the clusterization is done, the discretization cut points are defined as the minimum and maximum of the active domain of the attribute and the midpoints between the boundary points of the clusters (minimum and maximum values in each cluster).

#### 4. PROPOSED METHOD

Based on the k-means clustering algorithm, we propose a discretization method which avoids the  $O(n \log n)$  time requirement for sorting the data points. We propose a technique of choosing the initial centers of the clusters designed specifically for the clustering of a one-dimensional vector of real valued data, to be used in the process of discretization of a single attribute. We assume that the number of clusters is given.

The idea of the algorithm is to chose initial centers such that they are in increasing order. In this way, the recomputed centers are also in increasing order and therefore to determine the closest cluster for each value of the attribute  $A$  the algorithm does less comparisons than in the general case. The closest cluster either remains the one in which the value belongs to or it is one of the two neighbouring clusters. In this way the number of comparisons done for reallocation of cluster is no longer  $k$  but 3. Also there is no need to order all the values in  $adom(A)$  like in the case of equal- frequency interval discretization.

The cut points are defined as the minimum and maximum of the active domain of the attribute and the midpoints between the centers of the clusters.

**Input:** Vector of real valued data  $a = (a_1, a_2, \dots, a_n)$  and the number of clusters to be determined  $k$ .

**Goal:** Our goal is to find a partition of the data in  $k$  distinct clusters.

**Output:** The set of cut points  $t_0, t_1, \dots, t_k$  with  $t_0 < t_1 < \dots < t_k$  that defines the discretization of the  $adom(A)$ .

We use the arrays:

- $cluster$  with  $n$  elements such that  $cluster[a_i]$  represents the cluster associated with the value  $a_i, 1 \leq i \leq n$
- $C$  with  $k$  elements where  $C[i]$ = center of cluster  $i, 1 \leq i \leq k$ .

We use the boolean variable *change* to monitor the reallocation of clusters, *change* = true if at least one value is being moved to a different cluster.

**Algorithm:**

Compute  $a_{\max} = \max\{a_1, a_2, \dots, a_n\}$  and  $a_{\min} = \min\{a_1, a_2, \dots, a_n\}$ .

// Initialization of centers of the clusters:

Choose the centers as the first  $k$  distinct values of the attribute  $A$ .

```

Arrange them in increasing order i. e. such that  $C[1] < C[2] < \dots < C[k]$ .
Define boundary points  $b_0 = a_{\min}$ ,  $b_j = \frac{C[j]+C[j+1]}{2}$  for  $j = \overline{1, k-1}$ ,  $b_k = a_{\max}$ .
for  $i = 1$  to  $n$  do
    // Find the closest cluster to  $a_i$ .
    if  $a_i = b_k$  then  $cluster[a_i] = k$ 
        else Find  $j$ ,  $1 \leq j \leq k$  such that  $b_{j-1} \leq a_i < b_j$ .
             $cluster[a_i] = j$ .
        endif
    endif
endfor
// Iteration steps
change = true
while change
    change = false
    Recompute the centers of the clusters as the average of the values in each cluster.
    for  $i = 1$  to  $n$  do
        Let  $j = cluster[a_i]$ .
        // Find the closest cluster to  $a_i$  from the possible clusters  $\{j-1, j, j+1\}$ .
        if  $a_i < C[j]$  and  $a_i - C[j-1] < C[j] - a_i$  then  $cluster[a_i] = j-1$ ,  $change = true$ 
            endif
        if  $a_i > C[j]$  and  $C[j+1] - a_i < a_i - C[j]$  then  $cluster[a_i] = j+1$ ,  $change = true$ 
            endif
        endif
    endfor
endwhile
// Determination of the cut points
 $t_0 = a_{\min}$ 
for  $i = 1$  to  $k-1$  do
     $t_i = \frac{C[i]+C[i+1]}{2}$ 
endfor
 $t_k = a_{\max}$ 

```

## 5. CONCLUSION

We presented an unsupervised, global and static discretization technique based on the k-means clustering algorithm, which does not require sorting the data, a time consuming operation, common to the discretization methods. Future work may include testing the technique against other discretization methods of the same type and of different type. The method will be tested also against a random discretizer that generates randomly a subset of all midpoints between the values of  $adom(A)$  and takes this set as the set of cut points. A common measure of "goodness" of the discretization will have to be defined .

## BIBLIOGRAPHY

1. Butterworth R., *Contributions to Metric Methods in Data Mining*, Doctoral Dissertation, University of Massachusetts Boston, 2006
2. Cios, K., Pedrycz, W., Swiniarski, R., Kurgan, L., *Data Mining A Knowledge Discovery Approach*, Springer, 2007
3. Dougherty, J., Kohavi, R., Sahami, M., *Supervised and Unsupervised Discrimination of Continuous Features*. In: Proceedings of the 12<sup>th</sup> International Conference, Morgan Kaufman, (1995) p. 194-202
4. Fayyad, U. M., & Irani, K. B. *On the Handling of Continuous-valued Attributes in Decision Tree Generation*, Machine Learning 8, (1992). p. 87-102.
5. Forgy E.W., *Cluster analysis of multivariate data: efficiency vs. interpretability of classifications*, Biometrics 21 (1965), p.768-769.

6. Hartigan, J., Wong, M., *A k-means Clustering Algorithm*, Applied Statistics 28(1979) p.100–108.
7. Ho, K., Scott P., *Zeta: A Global Method for Discretization of Continuous Variables*, In: Proceedings of the 3<sup>rd</sup> International Conference on Knowledge Discovery and Data Mining (1997), p. 191-194
8. Kurgan, L., Cios, K., *CAIM discretization algorithm*, IEEE Transactions on Knowledge and Data Engineering, (2004) 16, no.2, p.145-153
9. MacQueen J, *Some methods for classification and analysis of multivariate observations*. Proceedings of the 5<sup>th</sup> Berkeley Symposium on Mathematics, Statistics and Probability 3 (1967), p. 281–297.
10. Witten, I., Eibe, F., *Data Mining. Practical Machine Learning Tools and Techniques*, Second edition, Morgan Kaufman, 2005
11. Wu, X. et al, *Top 10 Algorithms in Data Mining*, Knowledge Information Systems (2008) 14, p.1–37