Generalized *k*-means algorithm on nominal dataset

S. H. Al-Harbi & A. M. Al-Shahri
Information Technology Center, Riyadh, Saudi Arabia

Abstract

Clustering has typically been a problem related to continuous fields. However, in data mining, often the data values are nominal and cannot be assigned meaningful continuous substitutes. The largest advantage of the *k*-means algorithm in data mining applications is its efficiency in clustering large data sets. The *k*-means algorithm usually uses the simple Euclidean metric which is only suitable for hyperspherical clusters, and its use is limited to numeric data. This paper extends our work on the $\mathcal{D}_{\mathcal{CV}}$ metric which was introduced to deal with nominal data, and then demonstrates how the popular *k*-means clustering algorithm can be profitably modified to deal with the $\mathcal{D}_{\mathcal{CV}}$ metric. Having adapted the *k*-means algorithm, the $\mathcal{D}_{\mathcal{CV}}$ metric will be implemented and the results examined. With this development, it is now possible to improve the results of cluster analyses on nominal data sets.
Keywords: clustering, data mining, Mahalanobis metric, $\mathcal{D}_{\mathcal{CV}}$ metric, Hamming metric, k-means.

1 Introduction

A way of extracting information from a large data set is to cluster it. Clustering involves assigning objects into groups such that the objects in a group are similar to each other, but different from the objects in the other groups. Similarity is fundamental to the definition of a cluster and being able to measure the similarity of two objects in the same feature space is essential to most clustering algorithms. In a metric space, the dissimilarity between two objects is modelled with a distance function that satisfies the triangle inequality. It gives a numerical value to the notion of closeness between two objects in a high-dimensional space. More details of metric spaces can be found, for example, in [3]. Applications of clustering exist in diverse areas, e.g.

WIT Transactions on Information and Communication Technologies, Vol 40, © 2008 WIT Press
www.witpress.com, ISSN 1743-3517 (on-line)
doi:10.2495/DATA080051

Genetics: Finding similar DNA or protein sequences [8].
Disease: Finding a virus similar to a given one from a large virus dataset or finding groups of viruses with certain common characteristics [6].
Image recognition: Finding images similar to a given one from a large image library [7]
Document retrieval: Finding documents related to a given document in a digital library [4].
World Wide Web: Clustering or finding sets of related pages [5].

Many of the fields in data mining consist of categorical data which describe the symbolic values of objects. The most important types of categorical data can be classified as follows.

- Ordinal data: induces an ordering of objects. As well as distinguishing between $x = y$ and $x \neq y$, the ranking of the objects can be used to obtain an inequality, $x > y$ or $x < y$.
- Nominal data: an object has one of two or more values but no ordering can be given to these values.

Since ordinal data can be ordered, it can be transformed into integer values. These values are not necessarily sequential. Therefore, there is a need for metrics that can adequately deal with categorical data, especially when this contains nominal fields. Therefore, a new metric, $\mathcal{D}_{\mathcal{CV}}$, which uses *Cramer's V* statistic [12], is proposed to satisfy these requirement [2]. The $\mathcal{D}_{\mathcal{CV}}$ metric is based on the Hamming metric which deals with nominal data, but is also derived from the Mahalanobis metric. Unlike the Mahalanobis metric however, the $\mathcal{D}_{\mathcal{CV}}$ metric uses the relationship matrix of nominal fields.

The rest of this paper is organized as follows. Section 2 presents the $\mathcal{D}_{\mathcal{CV}}$ metric. In Section 3, we introduce the concept of generalized k-means algorithm. The combination of the generalized k-means algorithm and the $\mathcal{D}_{\mathcal{CV}}$ metric is demonstrated in Section 4. Section 5 describes experiment with the new algorithm and discusses the results. Section 6 summarizes our research.

2 $\mathbf{D}_{CV}$ metric

If the fields are categorical, special metrics are required, since distance between their values cannot be defined in an obvious manner. The Hamming distance is the most popular measurement that is used to compute the distance between nominal data. If p and q are nominal values, the Hamming distance is defined as follows:

$$\delta_{\mathcal{H}}(p, q) = \begin{cases} 0 & \text{if } p = q, \\ 1 & \text{otherwise.} \end{cases} \tag{1}$$

If p, q are n-tuples of categorical values, then we define the Hamming distance between them to be:

$$\delta_{\mathcal{H}}(p_1, q_1) + \cdots + \delta_{\mathcal{H}}(p_n, q_n). \tag{2}$$

where n is the number of categorical values. However, the Hamming metric has some limitations due to its lack of ability to handle any relationships between fields.

Let x and y be two objects described by nominal data, where $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, x_n)$. Then, we can introduce a new dissimilarity metric between x and y, thus

$$D_{\mathcal{CV}}(x, y) = \sqrt{\delta_{\mathcal{H}}(x, y)\, \mathcal{O}^{-1}\, \delta_{\mathcal{H}}(x, y)^T}, \tag{3}$$

where $\mathcal{O}$ is the relationship between fields and $\delta_{\mathcal{H}}(x, y)$ is defined to be the vector of the Hamming metric of corresponding nominal values. The relationship between two fields measures their correlation. All of the relationship $o(s, t)$ can be collected together into a relationship matrix $\mathcal{O} = [o_{st}]$, which is defined as follows.

$D = D_1 \times \ldots \times D_n$ is the domain of the database. A general record $r \in D$ is of the form $r = (r_1, \ldots, r_n)$ where $r_i \in D_i$ determines the value of attribute A_i. Let $D_s = \{u_1, \ldots, u_I\}$, $D_t = \{v_1, \ldots, v_J\}$. Then a contingency table for D_s and D_t is

$$\text{contingency-table}\,(D_s, D_t) = [N_{ij}], \tag{4}$$

where $N_{ij} = |\{r : r \in D, r_i = u_i \text{ and } r_j = v_j\}|$. Define $K_i = \sum_{j=1}^{J} N_{ij}$, $L_j = \sum_{i=1}^{I} N_{ij}$, $M = \sum_{i=1}^{I} K_i = \sum_{j=1}^{J} L_j$, then the chi-square value for D_s, D_t is computed by

$$\chi^2_{st} = \sum_{i,j} \frac{(M\, N_{ij} - K_i L_j)^2}{M\, K_i\, L_j}. \tag{5}$$

We then define the *Cramer's V* statistic [12] by

$$o_{st} = \sqrt{\frac{\chi^2_{st}}{N \min(I - 1, J - 1)}}, \tag{6}$$

Equation (6) measures how D_s and D_t are related, giving a value between 0 and 1. A value of zero indicates no correlation (i.e. independent fields), and a value of one indicates a strong correlation. For more details see [1, 2]

3 The generalized *k*-means algorithm

The well-known k-means algorithm [11] is the most popular algorithm for partitioning data. It uses an iterative, hill-climbing technique. Starting with an initial k partition, objects are moved from one partition to another in an effort to improve the results of clustering. In the case where the points are real valued, the centroid is the mean of the points in the cluster, and hence the algorithm is known as the k-means algorithm.

The largest advantage of the k-means algorithm in data mining applications is its efficiency in clustering large data sets. The k-means algorithm usually uses the simple Euclidean metric which is only suitable for hyperspherical clusters, and its

use is limited to numeric data. However, the most distinct characteristic of data mining is that it deals with categorical data. Consequently, Huang [9] presents an algorithm, called k-modes, to extend the k-means paradigm to categorical objects. k-modes uses the Hamming metric. However, the Hamming metric does not take into account any correlations between fields. In order to extend the k-means algorithm, and to make it applicable for dealing with different metric spaces, it needs to be generalized in the context of metric space.

The k-means algorithm starts with an initial partition, and then moves objects from one cluster to another in an effort to improve the value of the clustering. Thus, we are indeed using a simple greedy algorithm that searches for an optimal or near-optimal clustering.

Different metaheuristic search methods (such as simulated annealing, steepest ascent or tabu search [13]) could also be used to search for an optimal or near-optimal clustering.

In addition, if an objective function uses a centre point, a greedy algorithm can operate more efficiently. Thus, algorithms based on moving centres have been developed. Such "Moving-Centres" algorithms proceed by: (1) computing the centre point for every particular subset as a representative point, and (2) exploiting a metric, δ, to measure the distance between objects and that centre point. k-means is one of the most popular of these algorithms. The direct k-means algorithm uses the mean as a centre point, however, the idea of k-means and the notion of mean can be extended to general metric spaces.

In order to improve the efficiency of the greedy algorithm, the following functions are requires:

(1) A function *initial partition* to provide the initial k clusters $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k\}$.
(2) A function *centre* : $2^{\mathcal{C}} \to \mathcal{C}$ which given a cluster computes its "centre".
(3) A function *select*: $\text{Ç}(\mathcal{C}) \to 2^{\mathcal{C}}$ which given a cluster determines the elements of $\mathcal{C}$, which should be considered for reassigning.
(4) A termination condition *finish* : $\text{Ç}(\mathcal{C}) \to \{T, F\}$ which decides if and when the clustering is satisfactory, and then terminates the algorithm.

Definition 1. *Given a cluster* $\text{Ç} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k\}$ *and* $x \in \mathcal{C}$ *we define*

- $cluster(x) = i$ *iff* $x \in \mathcal{C}_i$
- $nearest(x) = j$ *iff for all* $j \neq l$, $\delta(x, centre(\mathcal{C}_j)) \leq \delta(x, centre(\mathcal{C}_l))$ *and if* $\delta(x, centre(\mathcal{C}_j)) = \delta(x, centre(\mathcal{C}_l))$ *then* $j < l$.

Algorithm 1, below, shows the abstract code for the generalization of the k-means algorithm.

3.1 The centre point

The notion of a centre is the key element in using the generalized k-means algorithm (i.e. Algorithm 2). In addition, it is crucial for the time efficient convergence of the algorithm. This convergence can be seen when the objective function uses the centre point as a representative point for each cluster, then each object is reallocated only if it is nearer to the centre point of a gaining cluster than to the centre

Ç ← *initial partition*;
for each $\mathcal{C}_i \subseteq$ Ç do $c_i \leftarrow centre(\mathcal{C}_i)$;
while not *finish*(Ç) do
  S:= *select*(Ç);
  for each $x \in S$ do
   • adjust Ç such that x is removed from $\mathcal{C}_{cluster(x)}$
    to $\mathcal{C}_{nearest(x)}$;
   • recompute $c_{cluster(x)}$ and $c_{nearest(x)}$;
  end-for
end-while

Algorithm 1. Generalization of the k-means algorithm.

point of a losing cluster. In this case the distance from the centre point decreases more for the losing cluster than it increases for the gaining cluster, giving an overall decrease in the objective function. Therefore, it is important that the centre point is efficiently identified. Centre points may be categorized into two types: (1) centroid, and (2) medoid.

Definition 2. *Given any set of points, $\mathcal{C}$, in a metric space, $\mathcal{M}$, a point $\hat{c} \in \mathcal{M}$ is called a centroid of $\mathcal{C}$ if*

$$\sum_{x \in \mathcal{C}} \delta(\hat{c}, x) \text{ is minimised.} \tag{7}$$

$\hat{c}$ will be denoted by *centroid*($\mathcal{C}$) [11].

Definition 3. *Given any set of points, $\mathcal{C}$, in a metric space, $\mathcal{M}$, a point $\hat{m} \in \mathcal{C}$ is called a medoid of $\mathcal{C}$ if*

$$\sum_{x \in \mathcal{C}} \delta(\hat{m}, x) \text{ is minimised.} \tag{8}$$

$\hat{m}$ will be denoted by the *medoid*($\mathcal{C}$) [10]. Note that a medoid must be an element of $\mathcal{C}$ whilst a centroid can be any element of $\mathcal{M}$. Neither is necessarily a unique point.

4 Implementing the $\mathcal{D}_{\mathcal{CV}}$ metric

As the $\mathcal{D}_{\mathcal{CV}}$ metric deals with categorical data, the use of the medoid, $\hat{m}$, is appropriate. When using such a metric, the generalized k-means algorithm needs to retain the medoid by building a summary table showing the frequency of each of the values. This table is important for two reasons; firstly, it reduces the amount of data to be stored and analysed, and secondly, it aids efficiency when computing medoids.

The summary table is used by each cluster during the first iteration of the generalized k-means algorithm, but it is used only once. Subsequent determinations of the medoid will use a local summary table for each cluster. Each local summary table contains the frequency values of the objects in that cluster. Then in each iteration of the algorithm, if a point $(x_1, \ldots, x_n)$ which belongs to cluster $\mathcal{C}_i$ is moved to cluster $\mathcal{C}_j$, then the medoids of both clusters ($\mathcal{C}_i$ and $\mathcal{C}_j$) are adjusted accordingly. As a result of this, the frequencies will be changed and the new centroid of each cluster will be updated according to the new frequencies.

This method defines a way of computing the medoid of $D_{\mathcal{CV}}$ from a given categorical data set. The advantage of this method is that it allows the generalized k-means algorithm to cluster categorical data without losing efficiency. The generalized k-means algorithm, together with $\mathcal{D}_{\mathcal{CV}}$, may be represented in the following operations:

1. **Preprocessing Stage**.
   a. Compute the relation matrix $\mathcal{O}$, and its inverse.
   b. Compute the weight for each field.
2. **Clustering Stage.**
   a. Select k random initial clusters, $\mathcal{C}_i$ $(1 \leq i \leq k)$, and compute the medoid, $\hat{m}_i$, of each cluster.
   b. Compute the summary tables for each cluster.
   c. Measure the distance between an object and the medoid point of each cluster and reassign an object to its nearest medoid.
   d. Adjust both summary tables of the clusters from which the object has been removed, and to which that object has been reassigned.
   e. Adjust both medoids of the cluster from which the object has been removed, and to which that object has been reassigned.
   f. Repeat steps c, d and e until a termination condition is held.

The algorithm may terminate either when there is no movement of objects or the value of an objective function is not improving. This algorithm uses the average connectedness measurement, which is the sum of the distances between the medoid of cluster, $\mathcal{C}$, and other points within $\mathcal{C}$.

$$f(\text{Ç}) = \frac{\sum_{i=1}^{k} \sum_{j=1}^{n} \mathcal{D}_{\mathcal{CV}}(x_j, \hat{m}_i)}{k}, \tag{9}$$

where n is the number of points.

The implementation of the generalized k-means algorithm together with the $\mathcal{D}_{\mathcal{CV}}$ metric may be summarized as follows:

To confirm this implementation, an experiment will be performed on a real data set and the results analysed, in the following section.

5 Experiment and results

To test the performance of the $\mathcal{D}_{\mathcal{CV}}$ metric, the Mushroom data set was taken from the UCI repository. The data set was originally drawn from the Audubon Society

$\mathcal{O}^{-1} \leftarrow$ the inverse of relationship matrix (Ç);
Ç $\leftarrow$ initial partitions, and compute the summary tables;
for each $\mathcal{C}_i \subseteq$ Ç do $\hat{m}_i \leftarrow medoid(\mathcal{C}_i)$;
Compute initial value of clusters $f(\text{Ç})_{.new}$;
  Repeat
    $f(\text{Ç})_{.old} = f(\text{Ç})_{.new}$;
    for each x do
      • adjust Ç such that x is removed from $\mathcal{C}_{cluster(x)}$ to $\mathcal{C}_{nearest(x)}$;
      • readjusted summary tables of $\mathcal{C}_{cluster(x)}$ and $\mathcal{C}_{nearest(x)}$;
      • readjusted $\hat{m}_{cluster(x)}$ and $\hat{m}_{nearest(x)}$;
    end-for
  until $f(\text{Ç})_{.new}$ is not less than $f(\text{Ç})_{.old}$

Algorithm 2. The implementation of the $\mathcal{D}_{\mathcal{CV}}$ metric.

Field Guide to North American Mushrooms [14]. The purpose of this experiment is to test the $\mathcal{D}_{\mathcal{CV}}$ metric in conjunction with the generalized k-means algorithm on this relatively large data set.

This data set includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the agaricus and lepiota family. There are 8124 records in the original data set, each record involves 22 fields, all with nominal data. Only one field, 'stalk-root', contains missing data. This field contains 2,480 missing values (denoted by '?'), and in this experiment these missing values were removed. The 'veil-type' field has only one value (i.e. 'p'), in which case it would not be helpful in the discrimination between the characteristics of the Mushroom data. Therefore, it was also removed from the experiment.

Firstly, the relation matrix, $\mathcal{O}$, and the its values which represent the correlation values between the fields were computed, for more detail see [1]. It may be observed that the 'veil-color' field has a correlation value equal to 1, together with the 'stalk-color-above-r' and 'stalk-color-below-r' fields. This indicates that the 'veil-color' field has a very strong correlation with these two fields. Secondly, the inverse matrix, $\mathcal{O}^{-1}$ is computed, and the weight for each field is also computed.

Algorithm 2 was then implemented and run fourteen times using different input values for the number of clusters (ranging from 2 to 15). The results are presented graphically as Figure 1.

Figure 1 shows that there is a dramatic decrease in the quality measurement between the second and third clusters, which then decreases more gradually to Cluster 8. However, when the number of clusters is greater than 8, the rate of change in the optimization criteria is both negligible and constant. Thus, 8 is probably the most appropriate number of clusters for this Mushroom data set.

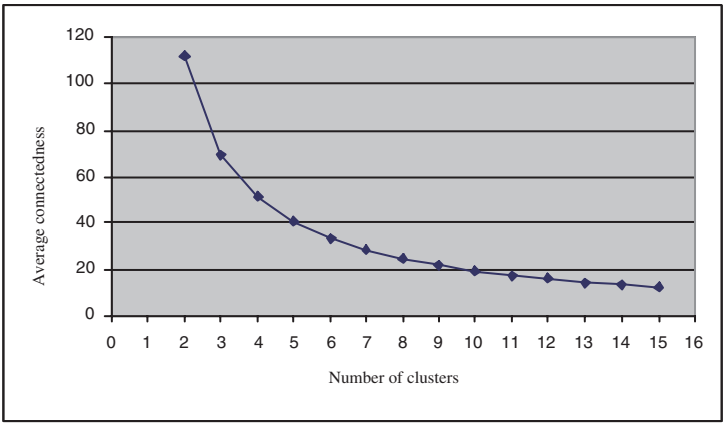

Figure 1: Results of the generalized k-means algorithm for clustering the Mushroom data set.

6 Summary

Clustering has typically been a problem related only to continuous fields. However, in data mining, data values are often categorical and cannot be assigned meaningful continuous substitutes. This paper has been a study of this problem and the $\mathcal{D}_{\mathcal{CV}}$ metric has been tested on a real data set, and the results of this experiment shows that the $\mathcal{D}_{\mathcal{CV}}$ metric is a useful distance function for measuring similarity in nominal data. However, in order to fully capitalize on this development, some changes in the algorithm used have also been proposed.

The biggest advantage of the k-means algorithm in data mining applications is its efficiency in clustering large data sets. However, its use is limited to numerical values. The generalized k-means algorithm presented in this paper has removed this limitation whilst preserving its efficiency.

These extensions allow us to use the k-means paradigm with different metric spaces. In this paper, we used this algorithm to cluster nominal data without the need for data conversion.

This combination of the generalized k-means algorithm and the $\mathcal{D}_{\mathcal{CV}}$ metric has presented a new and exciting approach to the problems inherent in the effective analysis of data. Categorical data, in particular, deserves more attention in the future. The results of these investigations are promising and prospects of more successful analyses are good.

References

[1] S. Al-Harbi. Clustering in Metric Spaces. PhD. Thesis, University of East Anglia, Under Progress.

[2] S. Al-Harbi, G. P. McKeown, and V. J. Rayward-Smith. A New Metric

for Categorical Data. In H. Bozdogan, editor, *Statistical Data Mining and Knowledge Discovery*, pages 339–351. CRC Press, 2003.
[3] E. T. Copson. *Metric Spaces*. Cambridge University Press, 1968.
[4] F. Daniel. An Analysis of Recent Work on Clustering Algorithms. Technical report, University of Washington, April 1999.
[5] O. Etzioni and O. Zamir. Web Document Clustering: A Feasibility Demonstration. In *Proceedings of the 21st Annual International ACM SIGIR Conference*, pages 46–54, 1998.
[6] A. Freitas. A Survey of Evolutionary Algorithms for Data Mining and Knowledge Discovery. In *A. Ghosh and S. Tsutsui (Eds.), Advances in Evolutionary Computation*. Springer-Verlag, 2001.
[7] K. S. FU. *Digital Pattern Recognition*. Springer-Verlag, 1976.
[8] E. Hartuv, A. Schmitt, J. Lange, S. Meier-Ewert, H. Lehrachs, and R. Shamir. An Algorithm for Clustering cDNAs for Gene Expression Analysis. In *RECOMB*, pages 188–197, 1999.
[9] Z. Huang. Clustering Large Data Sets with Mixed Numeric and Categorical Values. *Proceedings of The First Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1997.
[10] L. Kaufman and P. Rousseeuw. *Finding Groups IN DATA: An Introduction to Cluster Analysis*. John Wiley and Sons Inc, 1990.
[11] J. MacQueen. Some methods for classification and analysis of multivariate observations. *Proceeding of the 5th Berkeley Symposium*, 1:281–297, 1967.
[12] P.S. Nagpaul. *Guide to Advanced Data Analysis using IDAMS Software*. National Institute of Science Technology and Development Studies, New Delhi (India), 1999.
[13] V. J. Rayward-Smith, I. H. Osman, C. R. Reeves, and G. D. Smith. *Modern Heuristic Search Methods*. John Wiley and Sons Ltd., 1996.
[14] J. Schlimmer. The Audubon Society Field Guide to North American Mushrooms. In *ftp://ftp.ics.uci.edu/pub/machine-learning-databases/mushroom/*. UCI repository of machine learning databases, 1981.