Winter 12-5-2004

A New Clustering Algorithm for Categorical Attributes

Chunbin Tang

Weidong Zhao

A New Clustering Algorithm for Categorical Attributes

Chunbin Tang[1], Weidong Zhao[2]

[1] Management School, Fudan University, Shanghai 200433, China
[2] Software School, Fudan University, Shanghai 200433, China
bingotang@hotmail.com, wdzhao@fudan.edu.cn

ABSTRACT

Clustering over categorical attributes is an important yet tough task. In this paper, we present a new algorithm K-means Ⅱ to extend the famous K-means algorithm which is efficient only on numerical clustering, by using new cluster center definitions and new similarity measures. Thus, our algorithm can be used in categorical clustering while preserving the efficiency. Experiments on both real-life datasets and synthetic datasets show that the K-means Ⅱ algorithm can produce high quality results and deserve good scalability at the same time.

Keywords: clustering, categorical attributes, similarity, data mining

1. INTRODUCTION

Clustering is a widely used technique in which data points are partitioned into groups, in such a way that points in the same group, or cluster, are similar, and are dissimilar otherwise [1]. Much of the previous work focuses on numerical data which can be exploited to naturally defined distance functions between points. There are many algorithms proved to work quite well under numerical conditions, such as K-means [2], and so on.

However, much of the data in practice are categorical, where attribute values can not be naturally ordered as numerical values. Clustering for categorical attributes is thus an important task: it is applicable in different domains, e.g. an e-business recommender system. Let's consider a market basket database in a recommender system containing one transaction per customer, each transaction containing the set of items purchased by the specific customer. The transactions can be viewed as tuples with their attributes not numerical but categorical. As is shown in [3], we can set the attribute value as True if and only if an item is purchased; otherwise, it is False. We can cluster over the dataset and get the characterizations of each group, which can be used in targeted marketing and advertising. The characterizations are also effective in predicting buying patterns of new customers based on their profiles. Another example of categorical attributes may be “color” which can get values from the domain {black, yellow, white}. Something should be noted, that not all values of categorical attributes are of interest in practice during clustering. For example, we do not pay much attention to attribute value “False” while clustering on the market basket database.

Categorical clustering is yet a tough work, because the former familiar clustering algorithms for numerical attributes cannot be used directly. Fortunately, several algorithms have been proposed in this domain.

ROCK [3] is an adaptation of an agglomerative hierarchical clustering algorithm. They define "links" as the sum of number of common neighbors between two tuples, and then optimize the "links" based criterion function. We can get good results by using this algorithm. However, with the increase of the dataset size, scalability will degrade.

K-modes [4, 5] is an algorithm extending the K-means algorithm to work for categorical attributes. They define a new dissimilarity measure for categorical objects, “modes”, instead of means, and then use a frequency-based method to update modes in the clustering process. It is scalable as the number of clusters and the dataset size increases. However, the algorithm is unstable because of non-uniqueness of the modes.

STIRR [6] is an iterative algorithm based on non-linear dynamical systems. Some research shows that the known dynamical systems cannot guarantee convergence [7].

There are a few other clustering algorithms trying to solve the categorical clustering problem from different perspectives, such as CACTUS [8], Squeezer [9], COOLCAT [10], and so on. In this paper, we present K-means Ⅱ, a new clustering algorithm for categorical attributes. Similar to K-modes, K-means Ⅱ also extends K-means to categorical space. It modifies the definitions of cluster centers and similarity measures. Like K-means, it can achieve both high quality results and scalability.

The rest of this paper is organized as follows. In Section 2, several definitions related to categorical clustering are given. Section 3 shows our new algorithm in details. Some basic analysis is given in this section as well. In Section 4, experimental results on both real-life datasets and synthetic datasets are demonstrated. Finally, the conclusion is in Section 5.

2. DEFINITIONS

In this section, several definitions related to categorical clustering are given. Our definitions are quite like those in [5].

Definition 1: Let $A_1, A_2, \ldots, A_n$ be n attributes describing a space Ω, and $D_1, D_2, \ldots, D_n$ the domains of the attributes respectively. D_i is defined as categorical if it is finite and unordered, e.g., for any $a, b \in D_i$, either $a=b$, or $a \neq b$. A_i is then called a categorical attribute. Ω is a categorical space if all $A_1, A_2, \ldots, A_n$ are categorical.

Definition 2: A_i is called Boolean attribute if A_i is categorical and has two different values. For simplicity, we often represent D_i as $\{0, 1\}$. Ω is called a Boolean space if all $A_1, A_2, \ldots, A_n$ are Boolean. The market basket dataset above is a case in point.

Definition 3: Let the dataset D be a set of tuples where each tuple $t : t \in D_1 \times D_2 \times \ldots \times D_n$. If Ω is a categorical space, each data object X in the dataset D can also be represented as a conjunction of attribute-value pairs: $(A_1 = x_1) \cap (A_2 = x_2) \cap \ldots \cap (A_n = x_n)$, where $x_i \in D_i$, for $i=1,2, \ldots, n$. For the sake of simplicity, we present X as a tuple: $(x_1, x_2, \ldots, x_n) \in D_1 \times D_2 \times \ldots \times D_n$.

3. K-MEANS Ⅱ ALGORITHM

In this section, we introduce a new clustering algorithm over categorical space -- K-means Ⅱ, which is named after the famous K-means clustering algorithm. Since our K-means Ⅱ is similar with K-means, two main problems are met in our algorithm as well, which are the calculations of the cluster centers and the similarity measures between tuples and cluster centers. Therefore, we will solve the two problems before putting out our new algorithm.

3.1 Cluster Centers

We now define the center of a cluster as follows. Let $C= \{X_1, X_2, \ldots, X_m\}$ be a cluster of data objects, with $X_i=(x_{i,1}, x_{i,2}, \ldots, x_{i,n})$, for $i=1, 2, \ldots, m$. Let $t_j(a)$ be the number of the tuples with their j-th attributes' value as "a" in the cluster. Define $P_j(a)$ as the percentage of a, thus $P_j(a)= t_j(a)/m$. Furthermore, let P_j be the conjunction of $P_j(a)$, for all $a \in D_j$. Finally, we can define the cluster center Q as follows:

$$Q = (P_1, P_2, \ldots, P_n) \tag{1}$$

3.2 Similarity Measurement

Before we define the similarity between the tuple X_i and the cluster center Q, we first define the similarity between the j-th attribute of X_i and the j-th attribute of Q as:

$$\delta_{i,j} = P_j(x_{i,j}) \tag{2}$$

The following definition shows the similarity δ_i between X_i and Q:

$$\delta_i = \sum_{j=1}^{n} \delta_{i,j} \Big/ n \tag{3}$$

δ_i works quite well when all values in Dj $(j=1, 2, \ldots, n)$ are of great importance in clustering the dataset. However, in practice, we sometimes do not pay much attention to some attribute values. As is shown in Section I, the attribute value "False" is not of interest to us in the market basket. Therefore, we should modify the above definitions.

Let V_j be the set of important values of A_j, therefore, $V_j \subseteq D_j$. Then we define the similarity $\delta'_{i,j}$ between the j-th attribute of X_i and the j-th attribute of Q and its weight $\theta_{i,j}$.

$$\delta'_{i,j} = \begin{cases} P_j(x_{i,j}), x_{i,j} \in V_j \\ 0, x_{i,j} \notin V_j \end{cases} \tag{4}$$

$$\theta_{i,j} = \begin{cases} 1, x_{i,j} \in V_j \\ \sum_{a \in V_j} P_j(a), x_{i,j} \notin V_j \end{cases} \tag{5}$$

Thus, we can get the similarity δ'_i between X_i and Q:

$$\delta'_i = \sum_{j=1}^{n} \delta'_{i,j} \Big/ \sum_{j=1}^{n} \theta_{i,j} \tag{6}$$

When $V_j=D_j$ (for $j=1, 2, \ldots, n$), or in other words, when all values in Dj $(j=1, 2, \ldots, n)$ are of great importance in clustering the dataset, the two similarity δ'_i and δ_i equals. Thus, we can use δ'_i to replace δ_i. Finally, we get the function $Sim(X_i, Q)$ to compute the similarity between tuple X_i and the cluster center Q as follows:

$$Sim(X_i, Q) = \delta'_i \tag{7}$$

The above formula $Sim(X_i, Q)$ seems to be quite complex and odd. However, it does derive from the famous simple match coefficient and Jaccard's coefficient* developed to show the similarity for binary data. The function $Sim(X_i, Q)$ extends the above two

* The two are coefficients developed for binary data. Let a=conjoint presence (1, 1), b=mismatch (1, 0), c=mismatch (0, 1), and d=conjoint absence (0, 0). Then Simple Matching Coefficient= (a+d)/(a+b+c+d). Absence and presence as well as matches and mismatches have equal weights in this coefficient. Jaccard's Cofficient= a/(a+b+c). Conjoint absence (0, 0) is ignored in the coefficient.

functions, and can be used not only for binary data.

3.3 Our Algorithm

Our K-means Ⅱ algorithm is similar to the famous K-means [2] algorithm. It can also be described in four steps.

Step 1: Select k cluster centers randomly. (We may choose the first k tuples as the beginning cluster centers)

Step 2: For each tuple X_i, calculate the similarities between X_i and Q_l, which is $Sim(X_i, Q_l)$, for $l=1, 2, ..., k$. Assign X_i to the cluster C_l (from the formal cluster $C_{l'}$) so that the similarity between X_i and Q_l is the largest.

Step 3: Calculate the new k cluster centers, one for each cluster. Update both Q_l and $Q_{l'}$.

Step 4: Go to Step 2 until no tuple has changed clusters after a full cycle test of the whole dataset. Otherwise, end.

3.4 Selection of *k*

The selection of k plays an important role in the final cluster result. To facilitate the selection, we define a notion Δ to show the average similarity of the whole dataset.

$$\Delta = \sum_{X_i \in D} Sim(X_i, Q) \Big/ num \qquad (8)$$

Q is the center of the cluster which X_i is in, and num is the total number of tuples in the dataset. It is not surprising that Δ increases as k increases. If Δ is too small, it indicates that k is not chosen properly, and we should increase k. So it seems that we will get accurate results if we set k at a high level. However, if k is too large, we will not get very good cluster results, either. We can use Δ as an index to select an acceptable k by experiments.

3.5 Time Complexity

The time complexity of K-means Ⅱ algorithm depends on the size of dataset (num), the number of attributes (n), the number of iterations (i), and the number of clusters (k). Thus we can get that our algorithm has the worst-case time complexity $O(num*n*i*k)$. It shows that the time complexity is linear with the size of dataset, the number of attributes and the number of clusters. Therefore, we can make the conclusion that our algorithm deserves good scalability.

4. EXPERIMENTAL RESULTS

In this section, the results about the performance of K-means II are demonstrated. We examine the quality of the clustering results on real-life datasets, and the efficiency on synthetic datasets. Our algorithm is implemented in Java. All the experiments are conducted on a Pentium III-600 machine with 256M of RAM running Windows XP Professional.

4.1 Real-Life Datasets

We experiment our algorithm on two real-life datasets. One is the Congressional Voting dataset, and the other is the Mushroom dataset, both of which can be obtained from the UCI Machine Learning Repository [11]. The two datasets are also used in the algorithm ROCK which can produce good clustering results [3]. Thus, we can compare our K-means Ⅱ with ROCK.

Congressional Voting Dataset: It is the United States Congressional Voting Records in 1984. Each record corresponds to one Congress man's votes on 16 issues. Nearly all attributes are Boolean (Yes and No) values, and a few contain missing values. A classification label of Republican or Democrat is provided with each data record. The dataset contains tuples for 168 Republicans and 267 Democrats.

Mushroom Dataset: Each tuple in the dataset contains information that describes the physical characteristics of a single mushroom, with a poisonous or edible label. All attributes are categorical. It has 8124 different mushroom records, of which 4208 are edible. A few contain missing values, as is in Congressional Voting Dataset.

Table 1: results for congressional voting dataset

ROCK			K-means Ⅱ *(k=4)*		
No.	Rep.	Dem.	No.	Rep.	Dem.
1	144	22	1	23	3
2	5	201	2	140	30
K-means Ⅱ *(k=2)*			3	1	34
No.	Rep.	Dem.	4	4	200
1	7	220			
2	161	47			

As Table 1 illustrates, both ROCK and K-means Ⅱ can identify clusters with the majority of Republicans or Democrats. Due to the elimination of tuples, the sum of the sizes of clusters is not equal to 435 in ROCK. However, K-means Ⅱ does not eliminate any tuples. As is suggested in sub-section 3.4, we can increase the average similarity Δ by increasing k. Tables 1 also shows the results of K-means Ⅱ when k = 4. The result is quite accurate and encouraging. Therefore, we can conclude that K-means Ⅱ can produce good clustering results as ROCK does.

Table 2 describes the clustering results for Mushroom dataset by using ROCK and K-means Ⅱ. Nearly all clusters found by both algorithms are pure edible or

poisonous ones. Exceptions occur in cluster 5 and 10 (K-means Ⅱ), and in cluster 15 (ROCK). It shows that both algorithms perform well.

Table 2: results for mushroom dataset

ROCK					
No.	Edible	Poisonous	No.	Edible	Poisonous
1	96	0	12	48	0
2	0	256	13	0	288
3	704	0	14	192	0
4	96	0	15	*32*	*72*
5	768	0	16	0	1728
6	0	192	17	288	0
7	1728	0	18	0	8
8	0	32	19	192	0
9	0	1296	20	16	0
10	0	8	21	0	36
11	48	0			
K-means Ⅱ *(k=27,Δ=0.81)*					
No.	Edible	Poisonous	No.	Edible	Poisonous
1	192	0	15	224	0
2	96	0	16	0	864
3	288	0	17	0	40
4	372	0	18	192	0
5	*16*	*72*	19	0	864
6	276	0	20	324	0
7	0	675	21	32	0
8	192	0	22	324	0
9	0	288	23	432	0
10	*96*	*8*	24	128	0
11	768	0	25	0	256
12	0	621	26	160	0
13	0	192	27	0	36
14	96	0			

Experiments on both datasets show that our algorithm can produce accurate clustering results as ROCK does. We can conclude that our algorithm can produce high quality clusters.

4.2 Synthetic Datasets

In order to test the scalability of our algorithm, we experiment with synthetic datasets which are generated by using a data generator. We set the number of attribute values for each attribute to 5. All possible values are produced with (approximately) equal probability. We test three scalabilities of the algorithm using these datasets. The first one is the scalability of our algorithm against the dataset size, the second is the scalability against the number of clusters, and the third is the scalability against the number of attributes. In Figure 1, the dataset size is increased from 10000 to 100000, and the numbers of attributes and clusters are fixed to 10 and 30 respectively. Figure 2 shows the results when increasing number of clusters from 10 to 50, and the cluster size and the number of attributes are set to 10000 and 10. In Figure 3, we increase the number of attributes from 10 to 40, and we set cluster size at 10000 and the number of clusters at 30 fixedly.

These results are very encouraging because they show clearly a linear increase in time as the cluster size, the number of clusters, and the number of attributes increase. They accord with the analysis in sub-section 3.5. Therefore, we can make the conclusion that our algorithm has good scalability with respect to the above three dimensions.

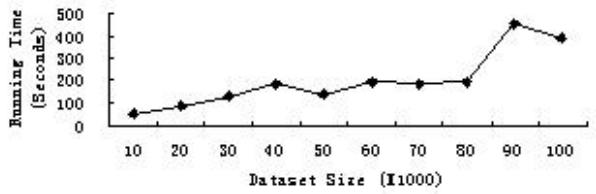

Figure 1: scalability to dataset size (attributes: 10, clusters: 30)

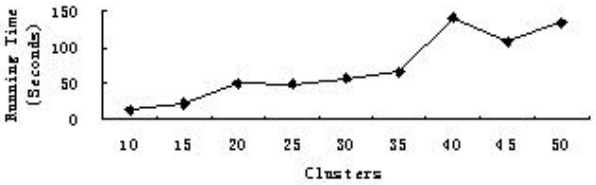

Figure 2: scalability to clusters (dataset size: 10000, attributes: 10)

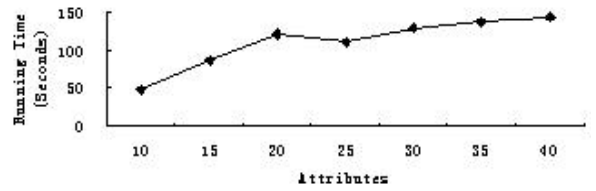

Figure 3: scalability to attributes (dataset size: 10000, clusters: 30)

5. CONCLUSION

Clustering of categorical attributes is an important yet difficult task. In this paper, we propose an efficient algorithm called K-means Ⅱ. Both analysis and experiments show that our algorithm can produce high quality results and at the same time deserve good scalability.

The K-means Ⅱ algorithm has made extensions to the K-means algorithm by using new cluster center definitions and new similarity measures. Thus, the K-means Ⅱ can be used in categorical clustering while preserving its efficiency. The new definition of cluster

center can also be used as characteristic descriptions which are useful in interpreting clustering results. Another advantage is that the algorithm can handle attribute values which are not of interest. Thus, the algorithm can handle missing values very easily, because missing values can be viewed as new attribute values which are not of interest while measuring the similarity.

For the future work plan, we will revise K-means Ⅱ to make it more efficient on very large dataset size, for examples, datasets with billions of tuples. The extension to K-means Ⅱ to make it work on mixed numerical and categorical attributes is another direction for us.

REFERENCES

[1] Anderberg, M., *Cluster Analysis for Applications*, Academic Press, 1973.
[2] MacQueen, J., Some Methods for Classification and Analysis of Multivariate Observations, In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp281-297, 1967.
[3] Guhay, S., Rastogi, R., Shim, K., ROCK: A Robust Clustering Algorithm for Categorical Attributes, In *Proceeding 1999 Internation Conference Data Engineering*, pp512-521, 1999.
[4] Huang, Z., Clustering Large Data Sets with Mixed Numeric and Categorical Values, In *Proceedings of The First Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1997.
[5] Huang, Z., A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining, *In Proceeding SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, pp146-151, 1997.
[6] Gibson, D., Kleiberg , J., Raghavan, P., Clustering categorical data : an approach based on dynamic systems, In *Proceeding of Very Large Database*, 1998.
[7] Zhang, Y., Fu, A., Cai, C., et al., Clustering Categorical Data, In *Proceeding of ICDE*, 2000.
[8] Ganti, V., Gehrke, J., Ramakrishnan, R., CACTUS-clustering categorical data using summaries, In *Proceeding of Knowledge Discovery and Data Mining*, pp73-83, 1999.
[9] He, Z., Xu, X., Deng, S., Squeezer: An Efficient Algorithm for Clustering Categorical Data, *Journal of Computer Science and Technology*, Vol. 17, No. 5, pp611-624, 2000.
[10] Barbará, D., Couto, J., Li, Y., COOLCAT: An entropy-based algorithm for categorical clustering, *Eleventh International Conference on Information and Knowledge Management (CIKM'02)*, 2002.
[11] *UCI Machine Learning Repository*. http://www.ics.uci.edu/~mlearn/MLRepository.html