SCALING K-MODE ALGORITHM FOR CLUSTERING LARGE CATEGORICAL DATASET AND ITS PERFORMANCE ANALYSIS

Ritesh Joshi*
Anil Patidar**

Abstract

Scalable data mining algorithms have become crucial to efficiently support KDD processes on large datasets. The k-mode is one of the partitioning algorithms used for the purpose of clustering. We show that basic k-mode algorithm is very much time consuming for large dataset instead we present the advanced algorithm which performs much better than known algorithm. In addition to presenting detailed experimental results for advanced k-mode algorithm, we also conduct an experimental study with real life datasets to demonstrate the effectiveness of our technique. We address the task of scaling up k-modes based algorithm through the utilization of memoization technique. Experimental result based on several datasets, including synthetic and real data, show that the proposed algorithm may reduce the number of distance calculations by a factor of more than a thousand times when compared to existing algorithms while producing clusters of comparable quality.

Keywords- Clustering, Categorical Datasets, K-mode, Memoization

1 Introduction Data mining is a process of extracting or mining knowledge from large dataset. Many people treat data mining as a synonym for another popularly used term knowledge discovery in databases (KDD). In the knowledge discovery process, we recognize the pattern and assign that pattern to a particular class. Thus it is important how a data mining system search patterns and assign to classes. For the same activity, on the basis of their working, different techniques are available like characterization and prediction, cluster analysis and outlier analysis, association mining etc.

1.1 Clustering: In cluster analysis we categorize our data into different groups on the basis of their similarity or according to some defined criteria such that objects within a cluster have high similarity in comparison to one another, but they are very dissimilar to objects in other clusters. Cluster analysis is a general activity, and a method of classification which we have been using in our day-to-day life frequently from the child age. Widely used in number of applications like data analysis, pattern recognition, image processing and market research. By clustering, we have to do

---

* Asst. Prof, MCA Department, Medicaps Institute of Technology & Management, Indore
** Associate Prof, MCA Department, Medicaps Institute of Technology & Management, Indore

find out density of data and relationship among data attributes. Cluster analysis is used in biology to derive plant and animal taxonomies, gene s categorization and similar functionality. Cluster analysis tools are based on K-means, K-modes and several other methods built and integrated in many statistical analysis software packages or system such as S-Plus, SPSS and SAS.

1.2 Types of Clustering Methods: Partitioning methods: These methods constructs K-partition form a data set of n records (as) where K < n) Deng at al proposed a partition based methods for cluster analysis[4].

Hierarchical methods: A hierarchical method creates a hierarchical decomposition of given data set. This method is further classified based on how decomposition is performed via (a) Top down (b) Bottom up. Nevin have proposed a hierarchical latent class models as a framework where the local dependence problem can be addressed in a principled manner.

Density-based methods: The general concept of this methods to continuously grows the size of given cluster as long as the density in the neighborhood exceeds some limit point.

Grid-based methods: These methods quantize the object space into a finite number of cells that form a grid structure. Main advantage of this method is fast processing time.

One of the popular algorithms for cluster analysis is K-mean based on partitioning method, which creates the cluster of records on the basis of the mean value and difference of record with the mean value. Main drawback of K-mean algorithm is works for only numerical data. Many of the previous algorithms have focused on numerical data because inherent properties can be exploited to naturally define the distance function between two points. But distance functions for categorical data sets are not naturally defined.

Data mining deals with very large datasets and data set contain many categorical attributes. So algorithm designed for clustering must be scalable and works for the non numeric data as well. To overcome with the problem of categorical data in the K-mean algorithm, we have used K-mode [2] algorithm which works on categorical data and is scalable. Now we have extended Basic K-mode algorithm to advanced K-mode algorithm using memorized environment.

1.3 Need of Memorization for Clustering: For speed up the data mining process two possible ways are available: data oriented approach, memorization approach. In first speed up can be achieved by removing noise from the data available where as in second we can achieve speed up by designing efficient and scalable memorized algorithms for efficient data mining.

As mentioned earlier in this project we have tried to scale the basic k-mode algorithm using the technique of memorization. In basic version when a new point is added to the cluster the change in cluster mode is calculated which is very cumbersome. Similarly-

extent of each point of cluster is calculated to be a ch point in the cluster which abruptly increases the complexity. Thus final the order of cluster then to complexity count of order O (n*n).

Proposed Enhancement: If we memoize the similarity extent of each point with respect to the cluster we can certainly reduce the effort, as only newly added point has to check to all the data point and subsequent changes can be reflected in the respective data point.

Similarity extent of each point stored with the data point with respect to the cluster to which it belongs. So whenever a new point is added its similarity extent is calculated to all the data point in the cluster and changed due to this point added to the respective points of cluster, reducing complexity to O (n).

1.4 Justification: K-mode algorithm is efficient algorithm for small data sets. On the basis of this mode selection method in maximize inter-cluster distance and minimize intra-cluster distance, which is the quality measure of clusters. This algorithm is applicable on categorical datasets, that we have selected K-mode method for clustering.

This paper is arranged as follows: Section 2 describes the related work in the domain. Section 3 describes background and preliminary definition and base formulation. Section 4 describes basic K-mode algorithm and its proposed advanced (scalable) version. Section 5 presents the experimental environments and test cases used. Section 6 illustrates the clustering results obtained. Section 7 presents the performance analysis of the proposed algorithm. Finally section 8 presents the conclusions that we can draw.

2 Related Work: Memoization is a fundamental and powerful technique for result-use. It has been used extensively in many areas such as dynamic programming, incremental computation, and many others. Memoization is a well known optimization technique used to eliminate redundant calls for pure functions. If a call to a function f with argument x yields result r, a subsequent call to f with x can be immediately reduced to r without the need to re-evaluate f's body. Eliminating redundant computation is an important optimization supported by many language implementations. One important instance of this optimization class is memoization (Liu and Teitelbaum 1995; Pugh and Teitelbaum 1989; Acar et al. 2003), we know nearly dynamic technique that can be utilized to avoid performing a function application by recording the arguments and results of previous calls. If a call is supplied an argument that has been previously cached, the execution of the function body can be elided, with the corresponding result immediately returned instead.

Umut A. Acar et al. in their paper [1] have proposed a framework for applying memoization selectively. Their framework provides programmer control over equality, space usage, and identification of precise dependences so that memoization can be applied according to the needs of an application. They discussed that although memoization can dramatically improve performance and can require some changes to the code, but no language or library support for memoization. Instead, many successful uses of memoization rely on application specific support code. The underlying reason for this

is one of control the cemem oization is all about performance; the user must be able to control the performance of memoization. Many subtleties of memoization, including the cost of equality checking and the cache replacement policy for memo tables, can make the difference between exponential and linear running time.

Ganti et al. proposed the CACTUS [1] the methodology for clustering by using cluster summaries. To find out the cluster they perform the process in three phases: *summarization, clustering and validation*. In the summarization phase, it computes the summary information from the dataset. In the clustering phase, it uses the summary information to discover a set of candidate cluster. In the validation phase, it determines the actual set of clusters from the set of candidate clusters. Overall CACTUS formalized the definition of a cluster when the data consists of categorical attributes, and then introduced a fast summarization based algorithm for discovering such cluster in categorical data.

Clustering of categorical data is an extensively researched area not only by data mining and database researchers, but also by other discipline people in the area of clustering categorical data. Han et al [11] addresses the problem of clustering transaction in a market basket database by representing frequent item sets as hyperedge in a weighted hypergraph. The weight of the graph is computed as the average of the confidence for all possible association rules that can be generated from the item set. Then a hypergraph partitioning algorithm is employed to partition the items, minimizing the weight of the cut hyperedge. The algorithm does not produce a clustering of the transactions and it is not obvious how to obtain one from the item clusters. Another related paper by Gibson et al. [14] also treats categorical approach to solve it based on non-linear dynamical system.

3. Definition Of Terms And Base Formulation

3.1 Categorical Data As referred in the paper categorical data term means data objects which have only categorical attribute. Here we consider single valued attribute as a categorical attribute and do not consider multi valued attributes as categorical attribute.

3.2 Categorical Domains and Attributes: Let $X_1, X_2, X_3 \ldots X_d$ be the d categorical attributes defining a space $Q(\theta)$ and Dom (X_1), Dom (X_2)… … Dom (X_d) the domain of attribute. Dom (X_1) is defined as all possible values of the attribute X_1 and is unordered.

3.3 Categorical Objects Let R be set of categorical objects {$R_1, R_2, R_3 \ldots \ldots \ldots R_N$}. We can define

$R_i = R_j$ (Equivalent of Categorical objects, ***overlap method***)

If $R_k = R_{jk}$ for all $k = 1$ to d,

In this project we have clustered records based on data driven similarity measures, overlap measure is one of them.

3.4 Similarity Measure: Let *X* and *Y* be two categorical objects on nattributes. The similarity measure between *X* and *Y* can be defined as total similarity of the respective attribute categories of the two objects. The greater the value of the total similarity, represents more closeness of two objects.

Note that we have converted measures that were originally proposed as distance to similarity measures in order to make the measures comparable in this study. The measures discussed henceforth will be in the context of similarity, with distance measures being converted using the formula:

$$sim = \frac{1}{1 + dist}$$

Almost all similarity measures assign a similarity value between two data instances X and Y belonging to the data set D as follows:

$$S(X, Y) = \sum_{k=1}^{d} w_k S_k(X_k, Y_k)$$

Where Sk(X k,Y k) is the per attribute similarity between two values for the categorical attribute A k. Note that X k,Y k ∈ A k. The quantity w k denotes the weight assigned to the attribute A k.

4 K-Mode Algorithm

4.1 Basic K-mode Algorithm : Sequential K-mode algorithm works by assigning *K* initial data objects from the dataset as modes for *K*-clusters. In next step algorithm selects one by one categorical object from the dataset and assign it to the proper cluster i.e. cluster whose mode is having maximum similarity with current record and re-compute mode of assigned cluster. When all objects are allocated to respective clusters algorithm rechecks the similarity of all categorical objects with current modes assigned to all clusters. If a categorical object is found whose similarity is greater to some other cluster mode then re-allocation of the unit is performed.

Data

Count to store similarity with cluster mode

IP

Number of clusters (*k*)

Dataset of categorical objects (DS)

Process

Step-1 :

Select k initial categorical objects from the dataset and assign them as initial mode of K-cluster.

Step-2 :

Select one by one categorical data point object from the dataset and assign it to the proper cluster i.e. assign it to the cluster whose mode is most similar with current data point. Formally

For every R_i in DS

{

For ($j = 1; j <= k; j++$)

$Count = Sim(Y_j, R_i)$

Where Y_j is the current mode of cluster and k is number of clusters.

Index = Index number of cluster to whom R_i is most similar

Add R_i object to index cluster and recalculate the centroid mode

}

Step-3 :

When all objects are allocated to clusters Recheck the similarity of all categorical objects with current modes assigned to clusters. If categorical object is found which is nearest to some other cluster mode then de-allocate from the current cluster and allocate categorical object to new cluster & recalculate the mode

Step-4 :

Repeat step-3 until no object has changed clusters after a full cycle test of whole dataset

4.2 Advanced K-mode Algorithm for Categorical Data Objects: The advanced K-mode algorithm uses the technique of memorization. We store the similarity extent of each data point with respect to the cluster.

Data

Count to store similarity with cluster mode

I/P

No of cluster (k)

Dataset of categorical object (DS)

Process

Step-1 :

Select k initial objects from the dataset and assign them as initial mode.

Step-2 :

Select one by one categorical data point object from the dataset and assign it to the proper cluster i.e. assign it to the cluster whose mode is most similar with current data point. Formally

For every R_i in DS

{

For ($j = 1; j <= k; j++$)

{

$Count = Sim(Y_j, R_i)$

Where Y_j is the current mode of cluster

Index = Index number of cluster to whom R_i is most similar

}
Add R_i object into index cluster
Calculate similarity extent of data point with all the co-cluster points
Memorize the Similarity value of data point
Recalculate the mode using previously memorized similarity values
}

Step 3:

Recalculate similarity of all records with all clusters (Shuffling Phase)
For cluster C_k
{
For all records R_i in C_k
For all clusters where $0<j<k$
Count ← *Sim* (R_i, M_j)
Index ← Index of which similarity is maximum.
If (j ≠ position)
{
Remove(R_i, j)
Add(R_i, *Index*)
}
}

Step 4:

Repeat step 3 until no object has changed clusters after a full cycle test of data set

Advanced K-mode algorithm has the following properties:

1. Basic *K-mode* algorithm has the complexity of $O(kmn^2t)$ where *k* is number of cluster, *m* is number of attributes, *n* is number of Categorical data set and *t* is number of iteration. In advanced *K-mode* algorithm we have reduced the complexity by n-factor using memorization. So new algorithm is having complexity of $O(kmnt)$
2. As data is not being modified in any form, so the actual meaning of data never lost.
3. Because of memorized operations, throughput of algorithm also increases.

To get insight into *K-mode* algorithm and its advanced version, we can assume that a set of categorical data objects consisting of *1000* (*n*) records with *22* (*m*) categorical attributes, and suppose the number of clusters to be created is *4*. Then the time complexity of sequential algorithm will be $1000 \times 1000 \times 4 \times 22 \times t \approx 88000000t$ units of time, where, *t* represents number of iterations to be taken for above data set by a single processor as per initial selection of the modes. We can extend the basic *K-mode* algorithm to advanced *K-mode* algorithm by using memorization technique.

As from the advanced *K-mode* algorithm all *1000* (*n*) categorical data objects having *22* (*m*) attributes are clustered into *4* clusters with initial values of *4*-modes (in our case). Now we have applied memorization technique which reduced the complexity by n times. So this version of K-mode algorithm has time complexity of $1000 \times 4 \times 22 \times t \approx 88000t$ units of time, with the assumption that the number of categorical data objects and number of attributes remain same.

5 Experimental Environment

5.1 Used Data Set We have used Mushroom data set to evaluate the computational efficiency of advanced *K-mode* algorithm. We have evaluated computational efficiency of both basic and advanced *K-mode* algorithms as discussed in this paper.

The Mushroom data set is one of the standard test data sets used in the machine learning community. It has often been used to test conceptual clustering algorithms. We have chosen this data set to test our algorithm because of its publicity and because all attributes can be treated as categorical without categorization.

The Mushroom data set has 8124 observations, each being described by 22 attributes. Each observation is identified by one of the 2 classes: Edible and Poisonous. Edible has 4208 observations and poisonous has 3916 observations.

5.2 Test cases- we create K-mode algorithm by considering following variations:

1. By keeping the number of clusters fixed and varying the record.
2. By keeping the records fixed and varying the number of cluster.
3. By varying the number of attributes in categorical space.

5.3 Testing Environment : Clustering experiments is performed on the *P4, 512* MB RAM and Windows XP Professional operating system. The Application for performance evaluation is developed in Java.

6 Clustering Result

6.1 Change in Number of Records We have fixed the number of cluster to 10. Changing the number of records from 5000 to 10000 (that is size of data set is changed) and evaluate the time required to partition the data set by both algorithms and draw the graph between number of records and time (Number of attributes in each tuple is 22).

Table 1 shows the comparison of number of records and time required for creating the cluster by Basic and Advanced algorithm :

Table 1 Relation between time and the number of Records

No of records	Time taken to execute Basic implementation (in milliseconds)	Time taken to execute Advanced implementation (in milliseconds)
500	2574	1748
600	3245	2043
700	5975	2402
800	7332	2761
900	10171	2995
1000	11638	3572

Figure 1: Represents the comparison graph corresponding to table 1.

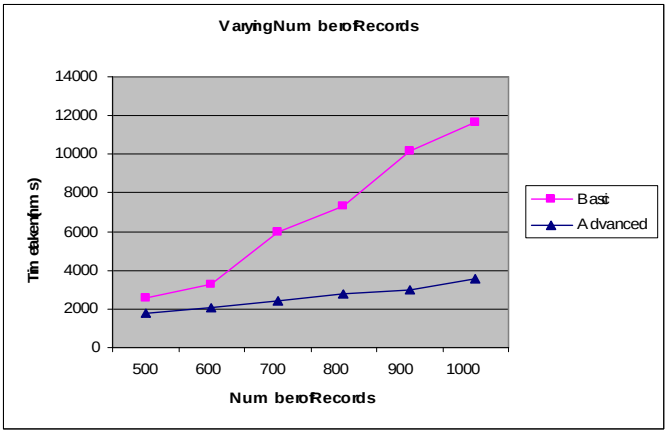

Figure 1: Relation between Records and Time

6.2 Change in Number of Cluster: We have fixed the number of record as 1000 and change the number of cluster from 6 to 10 and evaluate the time required to partition the records into clusters by both the algorithms and draw the graph between number of clusters and time (Number of attributes in each tuple is 2)

Table 2 shows the comparison of number of clusters and time required for each cluster by Basic and Advanced algorithm :

Table 2: Relation between Time and the Number of clusters

N o of C lusters	Tim e taken to execute B asic im plem entation (in m illiseconds)	Tim e taken to execute A dvanced im plem entation (in m illiseconds)
6	12293	3230
7	15522	3245
8	9079	4009
9	10250	3307
10	11762	3214

Figure 2 presents the graph corresponding to table 2:

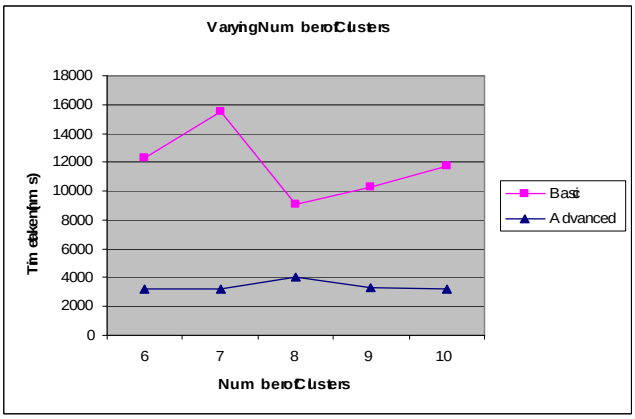

Fig 2: Relation between Time and the Number of clusters

6.3 Changing the Number of Attributes: We have fixed the number of records to 1000 and the number of clusters to 10, varied the number of attributes from 4 to 22 and evaluated the time required by both the algorithms and drawn the graph between attributes and time.

Table 3 shows the comparison for number of attributes and Time required for creating the clusters by Basic and Advanced algorithm:

Table 3: Relation between Attributes and Time

No of Attributes	Time taken to execute Basic implementation (in milliseconds)	Time taken to execute Advanced implementation (in milliseconds)
4	4337	2044
7	4181	2402
10	7160	2917
13	5506	2418
16	6817	2996
19	7129	2699
22	12511	3666

Figure-3 Represents the comparative graph corresponding table-3.

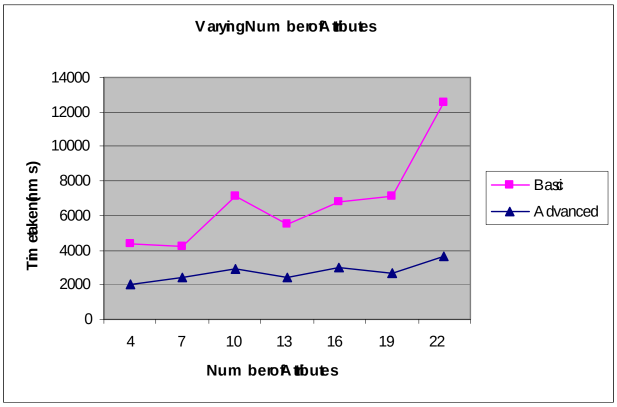

Fig-3 Relation between Attributes and Time

We have evaluated the three cases shown in Table-1 and corresponding figure-1 which considered the fixed number of clusters, attributes and changed the number of records and calculated the time for creating the clusters. In that way we evaluated the time by changing the clusters and keeping the number of records and attributes fixed shown in Table-2 and Figure-2 and calculated the time by changing attributes and keeping the clusters and records fixed in Table-3 and Figure-3. We observed that advanced algorithm is very efficient over the basic algorithm.

7. Conclusion: The Advanced version of K-mode algorithm presented in this paper has complexity of $O(tmn)$ that we achieve by the memoization technique. Such that, the algorithm is capable to perform the partitioning very large heterogeneous dataset of categorical objects into a number of smaller and more manageable objects. Implementation shows that the clustering problem can be optimized by adopting memoization method.

The implementation of the proposed advanced k-mode algorithm with variations in the environmental variables by keeping the number of records and attributes fixed and vary the number of clusters and/or by keeping the number of attributes and clusters fixed and vary the number of records and/or by keeping the number of records and clusters fixed and vary the number of attributes justifies the performance of the proposed algorithm.

8. References:

1. Umut A. Acar, Guy E. Blelloch, Robert Harper, "Selective Memoization", POPL'03, January 15-17, 2003, New Orleans, Louisiana, USA.
2. Lukasz Ziarek, KC Sivaramakrishnan, Suresh Jagannathan, "Partial Memoization of Concurrency and Communication", ICFP'09, August 31–September 2, 2009, Edinburgh, Scotland, UK.
3. Raymond T. Ng and Jiawei Han, "CLARANS: A Method for Clustering Objects for Spatial Data Mining", IEEE Transactions On Knowledge And Data Engineering, Vol. 14, No. 5, September/October 2002.
4. Shyam Boriah, Varun Chandola and Vipin Kumar, "Similarity Measures for Categorical Data: A Comparative Evaluation", Proceedings of 2008 SIAM Data Mining Conference, April 2008, Atlanta, GA.
5. V. Ganti, J. Gehrke, R. Ramakrishnan, "CACTUS-Clustering Categorical Data Using Summaries", Knowledge Discovery in Data Mining, 1999.
6. A. Ahmad and L. Dey, "A method to compute distance between two categorical values of same attribute in unsupervised learning for categorical data set", Pattern Recognition Lett., 28(1) 110–118, 2007.
7. Maria Camila N. Barioni, Humberto L. Razente, Agma J. M. Traina, Caetano Traina Jr., "An efficient approach to scale up k-mode based algorithms in large databases", XX Simpósio Brasileiro de Banco de Dados, 2003.
8. M. Ester, H. P. Kriegel, J. Sander, X. Xu, "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases", Proc. 1996 Int. Conf. Knowledge Discovery and Data Mining (KDD'96), pp. 226-231, Portland, OR, Aug. 1996.
9. Sudipto Guha, Rajeev Rastogi, Kyuseok Shim, "CURE: A clustering algorithm for large databases", Proceedings of the ACM Sigmod international conference on management of data, 1998.
10. David Gibson, Jon Kleinberg, Prabhakar Raghavan, "Clustering Categorical Data: An Approach Based on Dynamical Systems", Proc. 1998 Int. Conf. On Very Large Databases, pp. 311-323, New York, August 1998.

11. Y.Zhang,A.daW.aiC.heeFu,C.hunH.ingcai,PengA.nn Heng,"Clustering CategoricalD.ata"inProc.2000IEEEInt.Conf.D.a ataEngineering,SanD.eigo, U.S.A,M.arch2000.

12. Z.H.uang,"Extensionsto theK-M.eanA.lgorithm ford ustering LargeD.atasets w.ithCategoricalV.alues"D.ataM.iningandK.now ledgeD iscovery2,pp283-304, 1998.

13. C.Li,G.B.isw as,"U.nsupervisedLearningw.ithM.ixed N.um ericandN.om inal D.ata"IEEETransactionsonK.now ledgeandD.ataEngin eering,V.ol14N.o4, 2002.

14. T.C.hiu,D.Fang,C.hen,Y.W.ang,"A.R.oubstand ScalableClustering A.lgorithm forM.ixedTypeA.ttributesinLargeD.ataba seEnvironm ent"Proc. 2001Int.Conf.O.nK.now ledgeD.iscoveryandD.ataM.i ning,pp263-268,2001.

15. K.aufm an,L.andR.ousseeuw,P.(2005)Findinggr oupsindata:A.n introductiontoclusteranalysisJohnW.ileyandSo ns.