Clustering Data Without Distance Functions

G.D. Ramkumar
Information Technology Lab
Hitachi America

Arun Swami
Neta Corporation

Abstract

Data mining is being applied with profit in many applications. Clustering or segmentation of data is an important data mining application. One of the problems with traditional clustering methods is that they require the analyst to define distance functions that are not always available. In this paper, we describe a new method for clustering without distance functions.

1 Introduction

Mining for information from databases has several important applications [11]. Three of the most common methods to mine data are association rules [1, 2], classification [7], and clustering [8, 3]. Association rules derive patterns from grouped data attributes that co-occur with high frequency. Classification methods produce hierarchical decision models for input data that is sub-divided into classes. Finally, clustering methods group together the records of a data set into disjoint sets that are similar in some respect. Clustering also attempts to place dissimilar records in different partitions.

Clustering arises naturally in several applications. For example, in the context of super market data, clustering of sale items to perform effective shelf-space organization is a common application. Another application is in clustering medical patient care data so that services performed for similar treatments are grouped together.

There is a lot of work on clustering particularly in the field of statistics. Traditional clustering algorithms [6, 4, 9, 10] use distance functions to measure similarity. The distance between any two elsements to be clustered is used in the decision on whether to put them in the same cluster or in disjoint clusters. For many applications, there may not be natural distance measures. For example, if the elements to be clustered have many categorical attributes, defining useful distance measures is problematic.

In this paper, we explore methods of clustering without distance functions. The methods are based on two basic principles. The first principle is that elements that share many attribute values are similar. One method will favor clustering such elements. The second principle is to group elements that share a common value for a distinguished attribute or label. We present a method to cluster records using a combination of these two methods: the exact combination depends on an input parameter specified by the user.

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

2 Parametrized Clustering

We use the term *co-occurrence maximization* to describe the clustering method that tries to group together records that have frequently co-occurring items. We use the term *label minimization* to describe the clustering method that tries to cluster records of the same label. We propose that these two methods can co-exist in a hierarchical partitioning of data. At a node of the partition, one of co-occurrence maximization and label minimization is performed; choice of the type of partition depends on a) quality measures of co-occurrence maximization and label minimization, and b) a user specified numeric parameter that allows continuous variation between co-occurrence maximization and label minimization.

2.1 Co-occurrence Maximization

Let the input consists of a set of data records R. Each record $r \epsilon R$ defines a set of *(attribute, value)* pairs. We define the set of items I from R as the set of *(attribute, value)* pairs occurring in R. Items and combinations of items are interesting if they occur frequently enough in the data. The threshold frequency is given by a parameter *support* s. Pairs of items present in greater than a fraction s of the records are referred to in this paper as *frequent* item sets, $F \subseteq I^2$.

Recent work of Vipin Kumar, et al [3] studied clustering using frequent item set computation. Consider the set of items to be vertices of a graph. Frequent item sets can be viewed as edges of this graph (more generally, the authors consider frequent item sets of cardinality higher than 2, which induce hyper edges). The vertices I of the graph are partitioned into I_A and I_B so that the weight of edges across the partition is minimized. We refer to such an item partition as p_{AB}. Item partition p_{AB} can be used to induce partitions on the records of R, as described in [3]. Let us denote the induced partition as q_{AB} which divides the records of R into subsets R_A and R_B.

We implement co-occurrence maximization as follows:

1. Compute frequent item sets F of cardinality 2 from the data records, using any association mining algorithm, e.g., the Apriori algorithm [1].

2. Construct a graph G with vertices I as items and item sets J as the edges. The weight on a edge corresponds to the support for the item set.

3. Partition I into subsets I_A and I_B so that a) the weight of edges across the partition is minimized, and b) the number of vertices in the larger partition is close to $n/2$. We achieve such a partitioning using the randomized MinCut algorithm [5]; we terminate partitioning when condition b) is reached.

4. Partition the records of R into R_A and R_B. A record r of R goes into partition R_A (respectively, partition R_B) if its overlap with I_A is greater than its overlap with R_B. In calculating the overlap, the effect of single item frequency is taken into account by weighting each item's contribution with the inverse of its support.

2.1.1 Quality function Q_c

We measure the quality of a partitioning p_{AB} as a numeric between 0 and 1 as follows. Let $r \subseteq R$ be a record in the data set. If r falls entirely in I_A (or respectively, I_B) then the partition quality is maximum, namely 1. On the other hand, if r overlaps I_A and I_B equally, then the quality is minimum, namely 0. We are now ready to formally define the quality of a partitioning. Let the weighted cardinality of an item set $w(I)$ be defined as $\sum_{i \epsilon I} 1/s(i)$ where $s(i)$ is the support of the item in the data.

Definition: Quality $Q_c(r)$ of record r over partition p_{AB} is defined as

$$Q_c(r) = \frac{\textbf{abs}(s(r \cap I_A) - s(r \cap I_B))}{|I_A| + |I_B|}$$

Definition: Quality $Q_c(R)$ of a data set R over partition p_{AB} is

$$Q_c(R) = \frac{\sum_{r \epsilon R} Q_c(r, p_{AB})}{|R|}$$

We observe that the quality functions defined above have the desired properties mentioned earlier, in the way that they measure the partitions of items and records.

2.2 Label Minimization

The goal of label minimization based clustering of records is to increase the uniformity of class labels in each cluster; traditionally, this is measured using information gain at the clusters, as derived from an entropy function. We use a classification method similar to C4.5 [7] to partition the data records R based on their class labels. Let C denote the set of class labels of R. Partitioning of records is performed so that entropy of the data is reduced; in other words, the number of labels in each partition is reduced. The ultimate goal of partitioning is to make each cluster label uniform. Partitioning is performed using candidate tests derived from *(attribute, value)* pairs.

We look at how candidate tests are determined. Attributes of data records are of two types: categorical and numerical. For categorical attributes, the candidate tests are equality tests to each value of the attribute. For numerical attributes, candidate tests are range comparisons to check if the value of the attribute is less than or equal to the test value. Conceptually, for each test t, we divide the data records into R_A and R_B. We measure the gain in information attained by the division of R into R_A and R_B.

Let us define the information content of a set of records R along the lines of [7]. For every class label $c \epsilon C$, let $n(c, R)$ denote the frequency of occurrence of c in R. The information content $i(R)$ is defined as:

$$i(R) = \frac{\sum_{c \epsilon C} n(c, R) \times \ -\log_2(\frac{n(c,R)}{|R})}{|R|}$$

The information content of R_A and R_B combined is:

$$i(R_A, R_B) = \frac{i(R_A)|R_A| + i(R_B)|R_B|}{|R_A| + |R_B|}$$

The information gained by partitioning the data records into R_A and R_B using a test t is $i(R_A, R_B) - i(R)$. We pick the test that maximizes information gain in partitioning the set of data records. We are now ready to define the quality of label minimization as the relative information gain:

$$Q_l(R, t) = i(R_A, R_B) - i(R)$$

As in the case of co-occurrence, the quality measure Q_l is a number in the range $[0, 1]$.

2.3 Parametrization

We have now described two methods of partitioning the data: co-occurrence maximization based partitioning and label minimization based partitioning. Each method has a quality measure associated with it. We need a way to choose between the two partitioning methods at a recursive partitioning step. Such a choice can be facilitated using a user defined parameter w as follows. If $w * Q_c(R) > (1 - w) * Q_l(R)$, then pick co-occurrence maximization. Otherwise, pick label minimization.

Applying this strategy recursively results in a break down of the data into clusters. The parameter w can be used to control the clustering. As w increases from 0 to 1, emphasis goes continuously from co-occurrence maximization to label minimization. At an intermediate value of, say 0.5, emphasis is equally shared between co-occurrence maximization and label minimization. That is, the partitioning that has the best quality among the two is picked.

3 Experiments

We implemented our method for parametric clustering and tested it on some well known data sets. For brevity, we give only two examples in detail. We used data sets from the Machine learning data repository at UC Irvine[1]. We present sample results from applying our method to the following data sets:

- Zoo data set of animals.
- Adult data set of persons with attributes. We ran tests on both the test and the full data sets.

3.1 Results on Zoo Data Set

For the zoo data set, the schema attributes were:

1. Animal name: string.
2. Hair, feathers, eggs, milk, airborne, aquatic, predator, toothed, backbone, breathes, venomous, and fins: all boolean.
3. Legs: numeric (set of values: 0,2,4,5,6,8).
4. Tail, domestic, catsize: all boolean.
5. Type: numeric (integer values in range [1,7]).

The class label is the last attribute, namely type. The type attribute distinguishes mammals, fish, insects, birds, and so on.

When we ran parametric clustering with a parameter $w = 0.35$ and support threshold 0.06 for frequent itemsets, we got an interesting partition tree as shown in Figure 1. The test at the root node of the tree was a label minimization test on the milk attribute. All mammals were classified to the left. Subsequent test was a co-occurrence maximization test which divided the animals into two clusters A and B. Following are samples of animals from clusters A and B:

Partition A: Bear, boar, buffalo, calf, cheetah, deer, elephant, leopard, and others.

Partition B: Dolphin, fruitbat, girl, gorilla, platypus, seal, sealion, and others.

We noticed distinguishing characteristics of the partitions that are not otherwise apparent in the data attributes. Animals in partition A do not fly and partition B does not include any fast moving animals on land. Even though the class labels of the two partitions are identical, co-occurrence maximization was able to partition the data into qualitatively distinguishable groups.

3.2 Results on Adult Data Set

We ran tests on the adult data set as well and saw a few interesting clusters. For the adult data set, the schema attributes were:

1. age: continuous.
2. workclass, education: categorical.
3. education-num: continuous.
4. marital-status, occupation, relationship, race, sex: categorical.

[1]The URL is `http://www.ics.uci.edu/ mlearn/MLRepository.html`

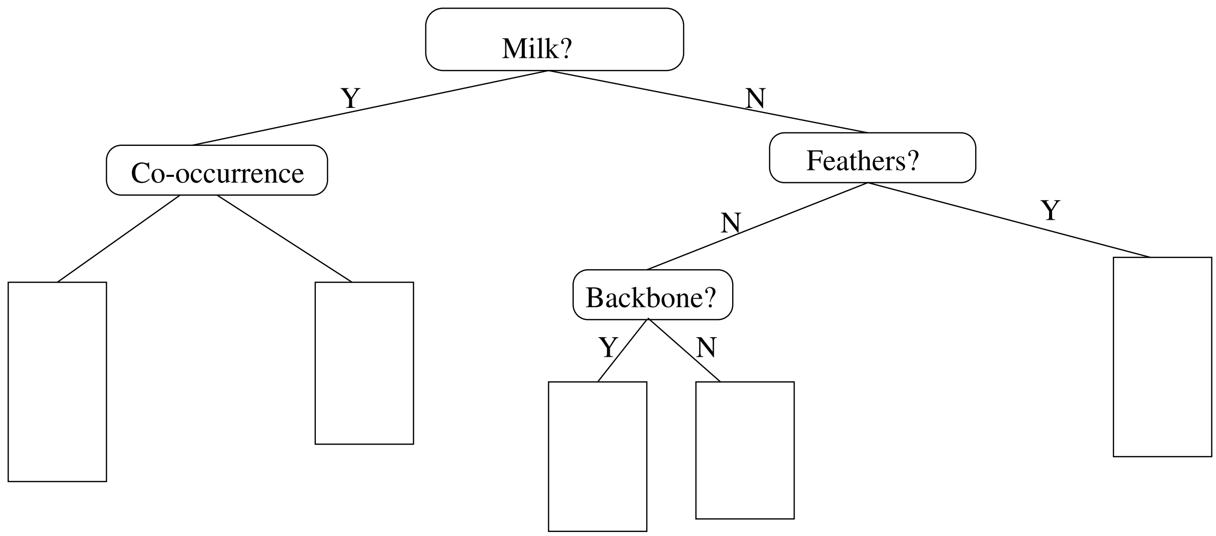

Figure 1: Partition tree obtained by performing parametric clustering on the zoo database, with parameter w set to 0.35 and a support threshold of 0.06.

5. capital-gain, capital-loss, hours-per-week: continuous.

6. native-country: categorical.

7. Class attribute is $> 50K$, $<= 50K$: categorical.

When we ran parametric clustering with a parameter $w = 0.3$ and support threshold 0.05 for frequent itemsets, we got an interesting partition tree. The test at the root node of the tree was a label minimization test on the marital status attribute. A few further partitions of the tree were co-occurrence maximizations. Subsequently, we noticed a label minimization partition testing the continuous attribute education level. The following test was a co-occurrence maximization test which divided the records into two clusters A and B. In Figure 3.2, we show some records sampled from partitions A and B. For clarity we show only a few attributes for each data record (we show missing attribute values by '?').

Note the distinguishing characteristics of partitions A versus partition B. Most records of partition A are white male. Partition B contains mostly private sector employees; also most records of B are female or not white. In this example as well, co-occurrence maximization was able to partition the data into qualitatively distinguishable groups, thereby enhancing the clustering capabilities of label minimization.

4 Conclusion

We presented a method for parametrized clustering of data using a strategy based on the principles of co-occurrence maximization and label minimization. The principles are naturally implemented using well known techniques from association rules and classification. Our method combines the benefits from the two approaches by seeking to optimize a quality function that is a weighted average of the qualities of the co-occurrence maximization and label minimization methods. Based on a user-defined parameter, parametrized clustering can be tuned to favor one of the methods more. The quality function we use depends linearly on the parameter and the quality functions of co-occurrence maximization and label minimization. Other quality functions could be used and are the subject of further research.

Partition A:

Employment	Education	Work class	Relationship	Race	Sex	Class
Self-emp-not-inc	HS-grad	Other-service	Husband	White	Male	$> 50K$
?	HS-grad	Other-service	Husband	White	Male	$<= 50K$
Self-emp-not-inc	10th	Farming-fishing	Husband	White	Male	$<= 50K$
Private	10th	Sales	Husband	White	Male	$<= 50K$

Partition B:

Employment	Education	Work class	Relationship	Race	Sex	Class
Private	10th	Other-service	Wife	White	Female	$<= 50K$
Private	11th	Exec-managerial	Husband	White	Male	$<= 50K$
Private	5th-6th	Machine-op-inspct	Wife	White	Female	$<= 50K$
?	10th	?	Husband	Asian-Pac-Islander	Male	$> 50K$

Figure 2: Partition tree obtained by performing parametric clustering on the adult database, with parameter w set to 0.3 and a support threshold of 0.05.

We tested our method on data sets from the machine learning library at UC Irvine. Further work remains on devising efficient algorithms and testing performance of parametric clustering.

References

[1] R. Agrawal, T. Imilienski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, pages 207–216, May 1993.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB Conference, Santiago, Chile*, 1994.

[3] E. Han, G. Karypis, V. Kumar, and B. Mobasher. Clustering based on association rule hypergraphs. In *Proc. Workshop on Research Issues on Data Mining and Knowledge Discovery*, 1997.

[4] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons, 1990.

[5] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, Cambridge, 1995.

[6] R. T. Ng and Jiawei Han. Efficient and effective clustering methods for spatial data mining. *Proc. of the Int'l Conf. on Very Large Data Bases (VLDB)*, 1994.

[7] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[8] T. Zhang, R. Ramakrishnan, and M. Linvy. Birch: an efficient data clustering method for large databases. In *Proc. of the 1996 ACM SIGMOD Int'l Conf. on Management of Data*, Montreal, Quebec, 1996.

[9] D. Fisher. Improving inference through conceptual clustering. In *1987 AAAI Conference*, pages 461–465, Seattle, Washington, 1987.

[10] D. Fisher. Optimization and simplification of hierarchical clusterings. In *Proc. of the First Int'l Conference on Knowledge Discovery and Data Mining*, pages 118–123, Montreal, Quebec, 1995.

[11] U.M. Fayyad, G. Piatetski-Shapiro, and P. Smith. From data mining to knowledge discovery: An overview. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 1–34. AAAI/MIT Press, 1996.