k-representatives Algorithm: a Clustering Algorithm with Learning Distance Measure for Categorical Values

Jae Heon Park[1)] and Sang Chan Park[2)]

KAIST (Korea Advanced Institute of Science and Technology), Department of Industrial Engineering ([1)]dewy@major.kaist.ac.kr, [2)]sangpark@kaist.ac.kr)

Abstract

We propose *k*-representatives algorithm, a clustering algorithm for data with categorical valuee. *k*-representatives algorithm takes an iterative refinement approach. It uses our modified value difference metric that measures the distance between all values of each feature statistically and clusters objects in dataset based on this metric. The algorithm iterates these two processes. We validate our algorithm with two real world datasets from UCI collection.

Keywords: Clustering Algorithm, Categorical Attribute, Value Difference Metric

1. Introduction

Segmentation is one of the major data mining operations. Its goal is to partition objects in database into segments of similar records, that is, records that share a number of prototypes and so are considered to be homogeneous. Segmentation is useful in a number of tasks and supports such as customer profiling or target marketing, cross selling and customer retention [1].

Clustering is the most frequently used technique to implement the segmentation operation. Though a lot of data sets dealt in data mining have categorical attributes, most existing clustering algorithms are limited to numeric attributes. The traditional approach is to convert category attributes into binary attributes and to treat the binary attributes as numeric in the clustering algorithms developed for numeric attributes. Ralambondrainy used applied approach to the *k*-means algorithm to cluster categorical data [2]. This approach needs to handle a large number of binary attributes when data sets have a great many categorical values. Huang presented *k*-modes algorithm which extends the *k*-means algorithm to categorical domains [3]. In the *k*-modes algorithm, the dissimilarity measure between two objects is defined by the total mismatches of the corresponding attribute categories of them. This measure is often referred to as simple matching.

Both of the dissimilarity measures used in [2] and [3] consider only mismatches of attribute categories of objects. They ignore the various degrees of similarity between categorical values. In this paper, we use a modified version of value difference metric, first introduced by Stanfill [4]. We developed *k*-representatives algorithm, an iterative clustering algorithm using this measure. It iteratively updates both of clusters and the distance measure.

In the next section, we introduce the value difference metric and its modified version used in this paper. In the third section, our clustering algorithm is described. The algorithm is validated by experiments with real world datasets in the next section. Discussion and conclusion are presented in the final section.

2. Value Difference Metric

In domains with categorical features, the "overlap" metric is usually used, counting the number of features that differ [5]. Cost and Salzberg observe that the overlap metric gives relatively poor performance in their learning tasks in categorical feature domains [6]. In 1986, Stanfill and Waltz proposed a new powerful metric for measuring the difference between two instances in domains with categorical features and they called it value difference metric (VDM) [4]. VDM takes into account similarity of feature values. This is the value difference metric.

$$\Delta(X,Y)=\sum_{i=1}^{N}\delta(x_i,y_i) \tag{1}$$

$$\delta(x_i,y_i)=d(x_i,y_i)w(x_i,y_i) \tag{2}$$

$$d(x_i,y_i)=\sum_{l=1}^{n}\left|\frac{D(f_i=x_i\cap g=c_l)}{D(f_i=x_i)}-\frac{D(f_i=y_i\cap g=c_l)}{D(f_i=y_i)}\right|^{k} \tag{3}$$

$$w(x_i,y_i)=\sqrt{\sum_{l=1}^{n}\left(\frac{D(f_i=x_i\cap g=c_l)}{D(f_i=x_i)}\right)^{2}} \tag{4}$$

where X and Y are two instances. x_i and y_i are values of the i^{th} feature for X and Y. N is the number of features and n is the number of classes. f_i and g indicate the i^{th} predicate feature and the class feature, respectively. c_l is one of possible classes. *D(condition)* is the number of instances in a given training dataset which satisfy the condition.

$d(x_i\ ,y_i)$ is a term for measuring the difference overall similarity between feature values x_i and y_i.. The term $\frac{D(f_i=x_i\cap g=c_l)}{D(f_i=x_i)}$ is the likelihood that an instance with x_i of i^{th} feature value will be classified as class c_l. d(x_i, y_i) has a small value if two values give similar likelihoods for all possible classes and this means that two values are similar. Though Stanfill and Waltz used the value of k=2 in their equation, Cost and Salzberg observed that experiments indicated that equally good performance is achieved when k=1. We also used the value of k=1 for simplicity.

$w(x_i, y_i)$ measures the strength with which the i^{th} feature constrain the values of the class. This measure represents the importance of each feature in classification. In our paper, we remove this term in order to give same weights to features because the classification information is not given in clustering tasks.

Our value difference metric in this paper is

$$\Delta(X,Y)=\sum_{i=1}^{N}\delta'(x_i,y_i) \tag{5}$$

$$\delta'(x_i,y_i)=d(x_i,y_i)=\sum_{l=1}^{n}\left|\frac{D(f_i=x_i\cap g=c_l)}{D(f_i=x_i)}-\frac{D(f_i=y_i\cap g=c_l)}{D(f_i=y_i)}\right| \tag{6}$$

3. *k*-representatives algorithm

Our algorithm takes iterative refinement approaches, which include EM and *k*-means and known as the most effective among various appraoches to solve the clustering problems. Fig. 1 shows the algorithm.

```
Initialize clusters and their representatives
Repeat
        Decide the memberships of instances to clusters
                For each feature, derive value difference matrix
                For each instance, measure distances between instances and clusters and classify
                it to the closest cluster
        Re-estimate the representatives of clusters
    Until no object has changed clusters.
```

Fig. 1 *k*-representatives algorithm

As the other iterative clustering algorithms, *k*-representatives algorithm also repeats two processes, deciding the memberships of instances to clusters and re-estimating the centroids. In our algorithm, the representatives replace the centroids of clusters because centroids exist only in numerical domains. A representative of a cluster shows the occurring ratios of all possible values of features in the members in the cluster. In order to compute the distances between instances and representatives, a process for deriving value difference matrix is inserted.

3.1 Initialization

Before beginning repeating processes, the algorithm initializes the representatives of clusters. First, it distributes the instances in the training dataset into k clusters randomly. After this initial random clustering, the representatives can be derived by the method described in 3.2.

3.2 Estimating of the representatives of clusters

The representative of a cluster is the distribution of feature values. It has the same role of the center of a cluster in numerical domains. In *k*-means algorithm, the mean of members of a cluster is used as the center. In categorical feature domains, we can't calculate the mean of clusters. Thus, we replace the mean with the representative of a cluster. The representative of l_{th} cluster, R_l is defined by (7) and (8).

$$R_l = \left(r_l(i, j)\right) \tag{7}$$

$$r_l(i, j) = \frac{N_l(v_{ij})}{N_l} \tag{8}$$

N_l is the number of members of the l_{th} cluster. v_{ij} indicates the j_{th} value of i_{th} feature and $N_l(v_{ij})$ is the number of instances with v_{ij} among the members of the cluster.

3.3 Derivation of value difference matrix

For each feature, the value difference matrix is derived statistically based on the instance in the training dataset according to (9). In fact, (9) is a simpler form of (6).

$$\delta(v_1, v_2) = \sum_{l=1}^{n} \left| \frac{N_{1l}}{N_1} - \frac{N_{2l}}{N_2} \right| \tag{9}$$

v_1 and v_2 are the possible values of a feature and n is the number of the possible classes. N_1 is the number of times v_1 occurred. N_{1l} is the number of times v_1 was classified into the class *i*.

For example, let's assume that one of features of instances in a given training dataset has three possible values v_1, v_2 and v_3 and all the instances are classified into three classes c_1, c_2 and c_3. Seven instances are included in the given training dataset. If the seven pairs of the value of the feature and the class of them are (v_1, c_1), (v_1, c_1), (v_2, c_2), (v_2, c_1), (v_2, c_1), (v_3, c_2) and (v_3, c_3), then the distances between the values can be calculated using (9).

$$\delta(v_1, v_2) = \sum_{i=1}^{3} \left| \frac{N_{1i}}{N_1} - \frac{N_{2i}}{N_2} \right| = \left| \frac{2}{2} - \frac{2}{3} \right| + \left| \frac{0}{2} - \frac{1}{3} \right| + \left| \frac{0}{2} - \frac{0}{3} \right| = \frac{2}{3} = 0.67$$

$$\delta(v_1, v_3) = \sum_{i=1}^{3} \left| \frac{N_{1i}}{N_1} - \frac{N_{3i}}{N_3} \right| = \left| \frac{2}{2} - \frac{0}{2} \right| + \left| \frac{0}{2} - \frac{1}{2} \right| + \left| \frac{0}{2} - \frac{1}{2} \right| = 2$$

$$\delta(v_2, v_3) = \sum_{i=1}^{3} \left| \frac{N_{2i}}{N_2} - \frac{N_{3i}}{N_3} \right| = \left| \frac{2}{3} - \frac{0}{2} \right| + \left| \frac{1}{3} - \frac{1}{2} \right| + \left| \frac{0}{3} - \frac{1}{2} \right| = \frac{4}{3} = 1.33$$

Because of the similarity property, $\delta(v_2, v_1)$= $\delta(v_2, v_1)$=0.67, $\delta(v_3, v_1)$= $\delta(v_1, v_3)$=2 and $\delta(v_3, v_2)$= $\delta(v_2, v_3)$=1.33. Because a value has distance zero to itself, $\delta(v_1, v_1)$= $\delta(v_2, v_2)$= $\delta(v_3, v_3)$=0. Table 1 is the value difference matrix built from these results.

Table 1 An Example of Value Difference Matrix

	v_1	v_2	v_3
v_1	0.00	0.67	2
v_2	0.67	0.00	1.33
v_3	2	1.33	0.00

3.4 Measuring distances and re-clustering the instances

Using the value difference matrix and the representatives of clusters, the algorithm measures the distances between instances and clusters and re-clusters instances into their closest clusters. Because an representative is not an instance but the distributions of its cluster, we can't use (5), which is the value difference metric between two instances. In our algorithm, the expected value of the value difference, $E[\Delta(X,R)]$, is used. X and R indicate an instance in the training dataset and a representative of a cluster, respectively.

$$E[\Delta(X,R)] = E[\sum_{i=1}^{N}\delta(x_i,r_i)] = \sum_{i=1}^{N}E[\delta(x_i,r_i)] = \sum_{i=1}^{N}\sum_{j=1}^{V_j}E[\delta(x_i,r_i)\,|\,r_i = v_j]p(r_i = v_j) \qquad (10)$$

r_i represents the probability distribution of the i_{th} feature value of R. $E[\delta(x_i,r_i)|r_i=v_j]$ is $E[\delta(x_i,v_j)] = \delta(x_i,v_j)$. $p(r_i=v_j)$ is the probability that r_i is v_j and can be replaced by $r(i,j)$, the occurring ratio of the j_{th} value of i_{th} feature in the cluster which is defined by (8). By replacing $p(r_i=v_j)$ with $r(i,j)$ in (10) , we can get the final form of our value difference metric.

$$E[\Delta(X,R)] = \sum_{i=1}^{N}\sum_{j=1}^{V_j}\delta(x_i,v_j)r(i,j) \qquad (11)$$

4. Experimental Results

We chose two real world data, Nursery dataset and Mushroom dataset, from UCI collection to validate the performance of k-representatives algorithm.

Mushroom: This dataset includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family. Each species is identified as definitely edible or poisonous. The dataset has 8124 instances that are described by 22 categorical features. We clustered the instances using the features with 20 initial random clusters. For each cluster, we divided its members into two groups, edible and poisonous, according to their classes assigned a priori and counted the number of members included in each group. Table 2 shows the results.

Table 2 Clustering results of Mushroom dataset

Cluster No.	No. of poisonous	No. of edible	Cluster No.	No. of poisonous	No. of edible
1	8	0	11	0	192
2	456	96	12	359	0
3	36	0	13	32	64
4	0	192	14	1	1760
5	1152	0	15	0	1056
6	0	704	16	0	0
7	0	144	17	576	0
8	0	0	18	0	0
9	0	0	19	1296	0
10	0	0	20	0	0

During running of the algorithm, 5 clusters among initial 20 clusters was diminished and lost all of their members and thus only 15 nonempty clusters are meaningful. Among the nonempty clusters, 10 clusters have completely homogenous members and 3 clusters (2nd, 5th and 16th) are almost homogenous.

Nursery: Nursery Dataset was derived from a hierarchical decision model originally developed to rank applications for nursery schools. It includes 12960 instances with eight categorical features and classified into five clusters. We clustered them with 30 initial clusters. For each cluster, we counted the number of instances per class. Table 3 shows the results.

Table 3 Clustering results of Nursery dataset

Cluster No.	Not Recom.	Recommended	Very Recom.	Priority	Spec. prior.
1	480	0	0	0	0
2	0	0	0	234	126
3	0	0	0	276	684
4	480	0	0	0	0
5	0	0	0	0	0
6	0	1	56	318	105
7	0	0	28	420	512
8	0	0	0	78	162
9	0	0	0	0	0
10	0	0	0	328	152
11	720	0	0	0	0
12	480	0	0	0	0
13	0	0	0	238	482
14	0	0	0	82	38
15	0	0	0	0	0
16	0	0	38	455	467
17	0	0	18	306	396
18	0	0	0	0	0
19	0	0	0	0	0
20	240	0	0	0	0
21	0	0	10	114	116
22	0	1	56	318	105
23	480	0	0	0	0
24	0	0	42	322	116
25	0	0	42	322	116
26	480	0	0	0	0
27	0	0	38	455	467
28	480	0	0	0	0
29	240	0	0	0	0
30	240	0	0	0	0

5 clusters of initial 30 lost all of their members during running of the algorithm. The instances of class “Not Recommended” were separated from those of other classes, “Recommended”, “Very Recommended”, “Priority” and “Spec. Priority”. The instances of “Priority” and “Spec. Priority” are distributed to the same clusters. This means that instances of the two classes have very similar feature values.

5. Discussion and Conclusion

In this paper, we developed a clustering algorithm for data in categorical domains. The distance measure introduced in our algorithm considers the various degrees of similarity between feature values. The algorithm takes an iterative refinement approach. The structure of the distance measure and clusters are updated along the iteration of the algorithm. We validated the algorithm using two real world dataset from the UCI collection. We investigated the distribution of real classes for each cluster after clustering by our algorithm. The results show that instances of different classes are separated by clusters very successfully. Moreover, the number of clusters is adjusted through removing unnecessary clusters during running of the algorithm.

Our algorithm derives the distance between instances in categorical domains numerically and the structure of our algorithm takes the most popular approach. It can be incorporated with other distance measure in numerical domains such as Euclidean distance with an appropriate scaling method between our distance measure and numerical distance measures, which is our future work.

References

[1] P. Cabena et al.; Discovering Data Mining from Concept to Implementation, Prentice Hall Inc., pp.66-68, 1998.

[2] H. Ralambondrainy; A Conceptual Version of the *k*-means Algorithm, Pattern Recognition Letters, Vol. 16, pp. 1147-1157, 1995.

[3] Z. Huang; Extensions to the *k*-Means Algorithm for Clustering Large Data Sets with Categorical Values, Data Mining and Knowledge Discovery, pp. 283-304, Vol. 2, 1998.

[4] C. Stanfill and D. Waltz; Toward Memory-Based Reasoning, Communications of the ACM, Vol. 29, No. 12, pp. 1213-1228, 1986.

[5] T. K. Ming; Discretization of Continuous-Valued Attributes and Instance-Based Learning, Technical Report, No. 491, Basser Department of Computer Science University of Sydney, NSW 2006, Australia, 1994.

[6] S. Cost and S. Salzberg; A Weighted Nearest Neighbor Algorithm for Learning with Symbolic Features, Machine Learning, Vol. 10, pp. 57-78, 1993.