A Genetic k-Modes Algorithm for Clustering Categorical Data

Guojun Gan, Zijiang Yang, and Jianhong Wu

Department of Mathematics and Statistics, York University,
Toronto, Ontario, Canada M3J 1P3
{gjgan, zyang, wujh}@mathstat.yorku.ca

Abstract. Many optimization based clustering algorithms suffer from the possibility of stopping at locally optimal partitions of data sets. In this paper, we present a genetic k-Modes algorithm(GKMODE) that finds a globally optimal partition of a given categorical data set into a specified number of clusters. We introduce a k-Modes operator in place of the normal crossover operator. Our analysis shows that the clustering results produced by GKMODE are very high in accuracy and it performs much better than existing algorithms for clustering categorical data.

1 Introduction

As a primary tool of data mining, cluster analysis divides data into meaningful homogeneous groups. Many clustering algorithms have been proposed and studied[1, 2, 3, 4], and optimization (minimizing an object function) has been among popular approaches. Unfortunately, some optimization based clustering algorithms, such as the k-Means algorithm[5] and the k-Modes algorithm[6], may stop at a local minimum of the optimization problem.

To be more precise, let us consider a given database $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$ with n objects each described by d categorical variables. Chaturvedi et al.[6] formulated the k-Modes algorithm to be a bilinear clustering model as:

$$X_{n\times d} = W_{n\times k} Z_{k\times d} + \text{error}, \tag{1}$$

where X is the data matrix (x_{ij}) (x_{ij} is the j-component value of $\boldsymbol{x}_i$), W is a binary membership matrix of n objects in k mutually exclusive, non-overlapping clusters(the (i, j) entry of W is 1 if $\boldsymbol{x}_i$ is in the jth cluster, 0 if otherwise), and Z is a matrix of modes(the (j, l) entry of Z is the mode of the jth cluster in the lth dimension). Note that a mode is the most likely value while a center is the mean. For example, the mode of $(1, 1, 1, 0)$ is 1, while the center of $(1, 1, 1, 0)$ is 0.75. The data matrix X in Equation (1) is known, whereas both W and Z are unknown and they are estimated iteratively to minimize an L_p-norm based loss function $L_p = \sum_{i=1}^{n}\sum_{j=1}^{d} |x_{ij} - \hat{x}_{ij}|^p$, where x_{ij} and $\hat{x}_{ij}$ are the (i, j)th entry of X and $\hat{X} = WZ$. Note that in the limiting case as $p \to 0$, the L_p-norm based loss

X. Li, S. Wang, and Z.Y. Dong (Eds.): ADMA 2005, LNAI 3584, pp. 195–202, 2005.

function becomes the simple matching distance[7]. The k-Modes algorithm starts with an initial Z, and then iterates between estimating W given the estimates of Z and estimating Z given the estimates of W. This process is repeated until two successive values of the L_0 loss function are equal.

Some difficulties are encountered while using the k-Modes algorithm. One difficulty is that the algorithm can only guarantee a locally optimal solution[6]. To find a globally optimal solution for the k-Modes algorithm, genetic algorithm (GA)[8], originally introduced by Holland[9], has been used. In GA's, the parameters of the search space are encoded in the forms of *strings* called *chromosomes*. A GA maintains a *population*(set) of N coded strings for some fixed *population size* N and evolves over *generations*. During each generation, three genetic operators, i.e. *natural selection*, *crossover* and *mutation*, are applied to the current population to produce a new population. Each string in the population is associated with a fitness value depending on the value of the objective function. Based on the principle of survival of the fittest, a few strings in the current population are selected and each is assigned a number of copies, and then a new generation of strings are yielded by applying crossover and mutation to the selected strings.

GAs have been successfully applied to clustering[10, 11]. In particular, Krishna and Murty proposed a genetic k-Means algorithm(GKA)[12]. This GKA, incorporating GA into the k-Means algorithm, is very effective in recovering the inherent cluster structures and searches faster than some other evolutionary algorithms used for clustering. Unfortunately, GKA works only for numerical data sets. In the present paper, we develop a genetic clustering algorithm (called GKMODE) by integrating a k-modes algorithm[6] introduced by Chaturvedi et al and the genetic algorithm. We must emphasize here that GKMODE is inspired by the GKA, but focuses on clustering categorical data.

2 The Genetic k-Means Algorithm

The GKA[12] is a hybrid clustering algorithm that integrates the k-Means algorithm and GA's. GKA is similar to the conventional GA's except that it uses the k-Means operator(KMO), one step k-Means, instead of the crossover operator. Hence GKA retains the best features of GA's and is efficient for clustering.

Denote by $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$ a set of n objects with d numerical attributes. (note that we used D as a categorical data set in Section 1). Let $C_1, C_2, ..., C_k$ be k mutually exclusive, non-overlapping clusters of D and let $w_{ij} = 1$ if $\boldsymbol{x}_i \in C_j$, 0 if otherwise, for $i = 1, 2, ..., n$ and $j = 1, 2, ..., k$. Then the matrix $W = (w_{ij})$ has the following properties:

$$w_{ij} \in \{0, 1\} \text{ and } \sum_{j=1}^{k} w_{ij} = 1. \tag{2}$$

Let the within-cluster variation of C_j be $S^{(j)}(W) = \sum_{i=1}^{n} w_{ij} \sum_{l=1}^{d} (x_{il} - z_{jl})$, and the total within-cluster variation, also called squared Euclidean(SE) measure, be

$S(W) = \sum_{j=1}^{k} S^{(j)}(W) = \sum_{j=1}^{k} \sum_{i=1}^{n} w_{ij} \sum_{l=1}^{d} (x_{il} - z_{jl})$, where x_{il} is the l-th component of object $\boldsymbol{x}_i$, and z_{jl} is the l-th component of $\boldsymbol{z}_j$, the center of C_j defined as $z_{jl} = \left(\sum_{i=1}^{n} w_{ij} x_{il}\right) / \left(\sum_{i=1}^{n} w_{ij}\right)$ for $l = 1, 2, ..., d$. The objective is to find $W^* = (w_{ij}^*)$ such that $S(W^*) = \min_{W} S(W)$.

In GKA, the search space consists of all the matrices that satisfy (2). A matrix W is encoded as a string s_W of length n such that $s_W(i) = j$ if object $\boldsymbol{x}_i$ belongs to the jth cluster. The initial population $\mathcal{P}(0)$ is selected randomly. To avoid illegal strings, i.e. partitions with empty clusters, $\lfloor \frac{n}{k} \rfloor$ randomly chosen data points are assigned to each cluster and the rest of the points are assigned to randomly chosen clusters.

The selection operator randomly selects a chromosome from the previous population according to the distribution given by $P(s_i) = F(s_i) / \sum_{i=1}^{N} F(s_i)$, where N is the population size, $F(s_i)$ represents fitness value of the string s_i in the population and is defined by

$$F(s_W) = \begin{cases} f(s_W) - (\bar{f} - c\sigma), \text{ if } f(s_W) - (\bar{f} - c\sigma) \geq 0; \\ 0, \qquad\qquad\qquad\quad \text{otherwise,} \end{cases}$$

where $f(s_W) = -S(W)$, $\bar{f}$ and σ denote the mean and standard deviation of $f(s_W)$ in the current population, respectively, c is a constant between 1 and 3.

The mutation operator changes an allele value depending on the distance between the cluster center and the corresponding data point. To apply the mutation operator to the allele $s_W(i)$ corresponding to object $\boldsymbol{x}_i$, for example, the $s_W(i)$ is replaced with a value chosen randomly from the distribution: $p_j = P(s_W(i) = j) = (c_m d_{max} - d_j) / \left(k c_m d_{max} - \sum_{l=1}^{k} d_l\right)$, where d_j is the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{z}_j$, $c_m > 1$ and $d_{max} = \max_{1 \leq j \leq k} d_j$. To avoid empty clusters, an allele is mutated only when $d_{s_W(i)} > 0$.

KMO is just one step of the k-Means algorithm: (**a**) calculate Z for the given matrix W; (**b**) form $\hat{W}$ by reassigning each data point to the cluster with the nearest center. KMO may result in illegal strings, which can be avoided by some techniques, such as placing in each empty cluster an object from the cluster with maximum within-cluster variation. Lu et al. (2004) proposed a fast genetic k-Means algorithm(FGKA)[13] in which illegal strings are permitted. Using the finite Markov chain theory, GKA is proved to converge to the global optimum.

3 GKMODE

GKMODE is similar to GKA except that k-Modes Operator is used instead of KMO and, most important, illegal strings are permitted. As in GKA, GKMODE has five basic elements: *coding*, *initialization*, *selection*, *mutation* and *k-Modes Operator*. The search space is the space of all binary membership matrices W

that satisfy (2). Coding in GKMODE is exactly the same as in GKA. The initial population $\mathcal{P}(0)$ is randomly generated as in FGKA[13]. We now describe the genetic operators used in GKMODE in detail.

3.1 The Selection Operator

To describe the selection operator, let us start with the definition of fitness value of a string. The fitness value of a string s_W depends on the value of the loss function $L_0(W)$, the limiting case of $L_p(W)$ as $p \to 0$. Since the objective is to minimize the loss function $L_0(W)$, a string with relatively small loss must have relatively high fitness value. In addition, illegal strings are less desirable and should be assigned low fitness values. As in [13], we defined the fitness value $F(s_W)$ of a string s_W as follows,

$$F(s_W) = \begin{cases} cL_{max} - L_0(s_W), & \text{if } s_W \text{ is legal;} \\ e(s_W)F_{min}, & \text{otherwise,} \end{cases} \tag{3}$$

where c is a constant in the interval $(0, 3)$, L_{max} is the maximum loss of strings in the current population, F_{min} is the smallest fitness value of the legal strings in current population if it exists, otherwise it is defined as 1, and $e(s_W)$ is the legality ratio defined as the ratio of the number of non-empty clusters in s_W over k(so that $e(s_W) = 1$ if s_W is legal).

The selection operator randomly selects a string from the current population according to the distribution given by $P(s_i) = F(s_i)/\sum_{j=1}^{N} F(s_i)$, where N is the population size. The population of the next generation is determined by N independent random experiments, i.e. apply the selection operator N times.

3.2 The Mutation Operator

In GKMODE, mutation changes a string value based on the distances of the cluster mode from the corresponding data point. It performs the function of moving the algorithm out of a local minimum. The closer a data point to a cluster mode, the higher the chance of changing the data point to that cluster.

Precisely, let s_W be a solution string and let $\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_k$ be the cluster modes corresponding to s_W. During mutation, the mutation operator replaces $s_W(i)$ with a cluster number randomly selected from $\{1, 2, ..., k\}$ according to the distribution: $p_j = [c_m d_{max}(\boldsymbol{x}_i) - d(\boldsymbol{x}_i, \boldsymbol{z}_j)]/\sum_{l=1}^{k} [c_m d_{max}(\boldsymbol{x}_i) - d(\boldsymbol{x}_i, \boldsymbol{z}_l)]$, where $c_m > 1$ is a constant, $d(\boldsymbol{x}_i, \boldsymbol{z}_j)$ is the simple matching distance between $\boldsymbol{x}_i$ and $\boldsymbol{z}_j$, and $d_{max}(\boldsymbol{x}_i) = \max_{1 \le j \le k} d(\boldsymbol{x}_i, \boldsymbol{z}_j)$. As in FGKA[13], $d(\boldsymbol{x}_i, \boldsymbol{z}_j)$ is defined as 0 if the jth cluster is empty. In general, mutation occurs with some mutation probability P_m specified by users. By applying the mutation operator, an illegal string may be converted to a legal one and a data point is moving towards a closer cluster with a higher probability.

3.3 The k-Modes Operator

In GKA, KMO is used in place of the crossover operator in order to speed up the convergence process. In GKMODE, the k-Modes operator, one step of the k-Modes algorithm[6], is introduced for the same reason. Let s_W be a solution string, k-Modes operator on s_W which yields $s_{\hat{W}}$ consisting of the following two steps: **(a) Estimate Z:** Given estimates of W, the mode matrix Z is determined as follows. The (j,l) entry z_{jl} of Z should be the mode of $(x_l : \boldsymbol{x} \in C_j)$, where x_l is the l-component of $\boldsymbol{x}$ and $C_j = \{\boldsymbol{x}_i : s_W(i) = j,\ 1 \le i \le n\}$. The mode matrix Z formed above optimizes the L_0 loss function[6]. **(b) Estimate W:** Given estimates of Z, the binary membership matrix W is determined as follows. The loss function $L_0(W)$ can be written as $L_0(W) = \sum_{i=1}^{n} f_i$, where $f_i (1 \le i \le n)$ is defined as $f_i = \sum_{j=1}^{d} \delta(x_{ij}, z_{s_W(i)j})$ ($\delta(x,y) = 0$ if $x = y$, 1 otherwise.). Note that f_i is a function only of $s_W(i)$. Thus to minimize L_0, one can separately minimize f_i with respect to parameter $s_W(i)$ for $i = 1, 2, ..., n$. Since $s_W(i)$ has only k possible values, i.e. $\{1, 2, ..., k\}$, we can try all these k values and select the value that minimizes f_i, i.e. $s_W(i) = \arg \min_{1 \le l \le k} \sum_{j=1}^{d} \delta(x_{ij}, z_{lj})$. To account for illegal string, we define $\delta(x_{ij}, z_{lj}) = +\infty$ if the lth cluster is empty[13]. This new definition here is introduced in order to avoid reassigning all data points to empty clusters. Thus illegal strings remain illegal after the application of k-Modes operator.

4 Experimental Results

GKMODE and the k-Modes algorithm are both coded in Matlab scripting language. Since Matlab is quite slow for loops, GKMODE is also coded in C++ programming language. Our experiments were conducted on a PC with 2.2 Hz CPU and 512M RAM.

4.1 Data Sets

The soybean disease data[14] is used to test our algorithm. We choose this data set to test for the algorithm for three reasons. First, all attributes of the data set can be treated as categorical; Second, the true clustering of the data set is known; Third, the value of the objective function corresponding to the true clustering is the global minimum.

We also tested the algorithm on the Mushroom data, the Congress Voting data and the Zoo data[14]. The true clusterings of the Congress Voting data and the Zoo data have objective function values 1988 and 149, respectively, while the clusterings produced by GKMODE(with parameters $G_{max} = 10, P_m = 0.4, N = 10$) have objective function values 1701 and 132, respectively. The Mushroom data is big, and the algorithm did not stop in 5 hours. Due to the space limit, the results for these three data sets are not presented here.

4.2 Clustering Quality Measures

We used the corrected Rand index[15] to assess the recovery of the underlying cluster structure. Let $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$ be a data set, and let $\mathcal{P} = \{C_1, C_2, ..., C_{k_1}\}$ and $\mathcal{P}' = \{C'_1, C'_2, ..., C'_{k_2}\}$ be two clusterings of D. Denote by n_{ij} the number of points simultaneously in C_i and C'_j, i.e. $n_{ij} = |C_i \cap C'_j|$, then the corrected Rand index is defined as

$$\gamma = \frac{\binom{n}{2}\sum_{i=1}^{k_1}\sum_{j=1}^{k_2}\binom{n_{ij}}{2} - \sum_{i=1}^{k_1}\binom{|C_i|}{2}\sum_{j=1}^{k_2}\binom{|C'_j|}{2}}{\frac{1}{2}\binom{n}{2}\left[\sum_{i=1}^{k_1}\binom{|C_i|}{2} + \sum_{j=1}^{k_2}\binom{|C'_j|}{2}\right] - \sum_{i=1}^{k_1}\binom{|C_i|}{2}\sum_{j=1}^{k_2}\binom{|C'_j|}{2}}.$$

The corrected Rand index γ ranges from 0 when the two clusterings have no similarities(i.e. when one consists of a single cluster containing the whole data set and the other only clusters containing single points), to 1 when the two clusterings are identical. Since we know the true clustering of the data set, the true clustering and the resulting clustering are used to calculate γ.

4.3 Results

In the following tests, we select the constants $c = 1.5$, $c_m = 1.5$ and the input number of clusters $k = 4$ for GKMODE. We tested the algorithm for different values of the following parameters: mutation probability P_m, population size N and maximum number of generations G_{max}.

To compare the k-Modes algorithm and GKMODE, we run each of them 100 times. All objects are correctly clustered into the 4 given clusters by GKMODE for these 100 runs. The average clustering accuracy of GKMODE is 100%. However, the average clustering accuracy of the k-Modes algorithm is about 71% and the number of correct clusterings is 26 out of 100. The results show that the GKMODE produces a more accurate clustering result than the k-Modes algorithm. GKMODE is also better than the tabu search based k-Modes algorithm[16], in which the number of correct clusterings is 67 out of 100.

Table 1 gives the clustering results of GKMODE under different sets of parameters. For each set of the parameters (N, P_m, G_{max}), GKMODE is ran 100 times. In these tests, we choose a wide range of the mutation probability, and we see from the table that the average clustering accuracy of GKMODE is above 88% and the number of correct clusterings is at least 49 out of 100. Because of the limit of the number of generations, the algorithm stops before achieving the global optimum in some cases. Even in the worst case, GKMODE is better than the k-Modes algorithm.

From Table 1, we have following observations: **(a)** When N and G_{max} are fixed, the average clustering accuracy tends to decrease when the mutation probability P_m increases except for some cases. **(b)** When N and P_m are fixed, the average clustering accuracy increases when the maximum number of generations increases except for two cases. This makes sense. But larger values of G_{max} make the algorithm run longer. Therefore, there is a trade-off between the run-

Table 1. Clustering results of GKMODE for different parameters, the algorithm runs 100 times for each parameter setting. The input number of clusters is 4. $\bar{\gamma}$ is the average accuracy, $N_{\gamma=1.0}$ is the number of runs that $\gamma = 1.0$

N	P_m	G_{max}	$\bar{\gamma}$	$N_{\gamma=1.0}$	N	P_m	G_{max}	$\bar{\gamma}$	$N_{\gamma=1.0}$
10	0.2	5	0.9982	99	20	0.2	5	1.0	100
	0.2	10	0.9988	99		0.2	10	1.0	100
	0.3	5	0.9962	95		0.3	5	1.0	100
	0.3	10	1.0	100		0.3	10	1.0	100
	0.4	5	0.9212	59		0.4	5	1.0	100
	0.4	10	1.0	100		0.4	10	0.9995	99
	0.5	5	0.9013	56		0.5	5	1.0	100
	0.5	10	1.0	100		0.5	10	1.0	100
	0.6	5	0.8868	52		0.6	5	1.0	100
	0.6	10	1.0	100		0.6	10	0.9977	99
	0.7	5	0.9785	76		0.7	5	0.9962	96
	0.7	10	0.9994	99		0.7	10	0.9973	99
	0.8	5	0.9344	49		0.8	5	0.9673	71
	0.8	10	0.9863	86		0.8	10	0.9961	94

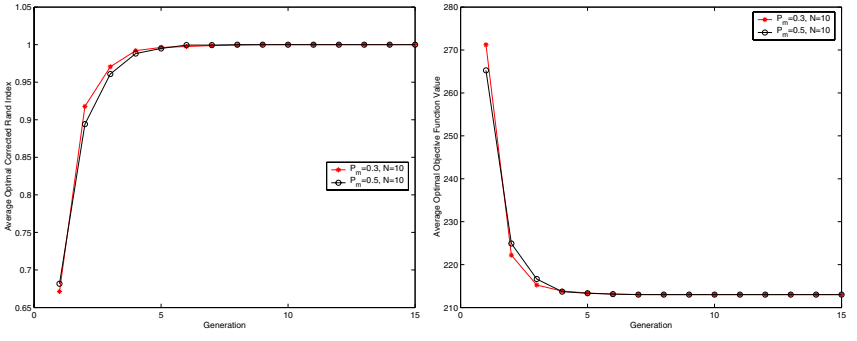

Fig. 1. Average optimal corrected rand index changes(Left) and Average optimal objective function value changes(Right) over generations for 100 runs

ning time and the maximum number of generations. **(c)** When P_m and G_{max} are fixed, the average clustering accuracy of a relatively large population size N is in general higher than that of a relatively small population size N.

We also study the average convergence of the clustering accuracy and the objective function value over generations for two different mutation probabilities. In both cases, GKMODE converges very fast to the extent that it will reach the global optimal clustering in five generations. The convergence of clustering accuracy and the convergence of objective function value are shown in Figure 1.

5 Conclusions

We have introduced the genetic k-Modes algorithm(GKMODE) for finding a globally optimal partition of a given categorical data set into a specified number of clusters. This incorporates the genetic algorithm into the k-Modes algorithm, and our experimental results show that GKMODE is very effective in recovering the underlying cluster structures from categorical data if such structures exist. Note that GKMODE requires the number of clusters k as an input parameter, how to incorporate validity indices for selecting k into GKMODE remains an interesting and challenging problem.

References

[1] Jain, A., Murty, M., Flynn, P.: Data clustering: A review. ACM Computing Surveys **31** (1999) 264–323
[2] Murtagh, F.: A survey of recent advances in hierarchical clustering algorithms. The Computer Journal **26** (1983) 354–359
[3] Cormack, R.: A review of classification. Journal of the Royal Statistical Society. Series A (General) **134** (1971) 321–367
[4] Gordon, A.: A review of hierarchical classification. Journal of the Royal Statistical Society. Series A (General) **150** (1987) 119–137
[5] Hartigan, J.: Clustering Algorithms. John Wiley & Sons, Toronto (1975)
[6] Chaturvedi, A., Green, P., Carroll, J.: k-modes clustering. Journal of Classification **18** (2001) 35 – 55
[7] Huang, Z.: Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery **2** (1998) 283–304
[8] Filho, J., Treleaven, P., Alippi, C.: Genetic-algorithm programming environments. IEEE Computer **27** (1994) 28–43
[9] Holland, J.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI (1975)
[10] Maulik, U., Bandyopadhyay, S.: Genetic algorithm-based clustering technique. Pattern Recognition **33** (2000) 1455–1465
[11] Hall, L., Özyurt, I., Bezdek, J.: Clustering with a genetically optimized approach. IEEE Trans. on Evolutionary Computation **3** (1999) 103–112
[12] Krishna, K., Narasimha, M.: Genetic k-means algorithm. Systems, Man and Cybernetics, Part B, IEEE Transactions on **29** (1999) 433–439
[13] Lu, Y., Lu, S., Fotouhi, F., Deng, Y., Brown, S.: FGKA: a fast genetic k-means clustering algorithm. In: Proceedings of the 2004 ACM symposium on Applied computing, ACM Press (2004) 622–623
[14] Blake, C., Merz, C.: UCI repository of machine learning databases (1998) `http://www.ics.uci.edu/~mlearn/MLRepository.html`.
[15] Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification **2** (1985) 193–218
[16] Ng, M., Wong, J.: Clustering categorical data sets using tabu search techniques. Pattern Recognition **35** (2002) 2783–2790