An Information-Theoretical Approach to Clustering Categorical Databases using Genetic Algorithms

Dana Cristofor Dan A. Simovici*

Abstract

Clustering categorical databases presents special difficulties due to the absence of natural dissimilarities between objects. We present a solution that overcomes these difficulties that is based on an information-theoretical definition of dissimilarities between partitions of finite sets (applied to partitions of the set of objects to be clustered which are determined by categorical attributes) and makes use of genetic algorithms for finding an acceptable approximative clustering. We tested our method on databases for which the clustering of the rows is known in advance and we show that our proposed method finds the natural clustering of the data with a good classification rate, better than that of the classical algorithm k-means.

Keywords: categorical attributes, classification, clustering, genetic algorithm, partitioning

1 Introduction

Clustering a set of objects with categorical attributes presents a special challenge since natural distances (like Euclidean or Manhattan distances) between objects are not available. As we shall see, the Hamming distance (that gives the number of disagreements between the values associated with two objects) is a blunt instrument that generates clusterings of poor quality. Previous research in clustering categorical data include [VJR99, SRK99, Hua97, HGV97].

The application of genetic algorithms to grouping problems was investigated in [Hol92, Mic99, Fal99, Mit97, JB91, ECM97]. Our contribution, which results in better performing algorithms, is centered around a measure of dissimilarity between partitions of a set of objects, introduced in [SCC00b, SCC00a]. To search for the best clustering of the data we use a genetic algorithm approach, where the chromosomes represent possible clustering solutions and the fitness measure captures the dissimilarity between the partition associated with a chromosome and the partitions determined by the attributes of the input database. The proposed clustering process involves two phases:

1. Initially, we estimate the influence of each attribute on the decision to place an object in a specific cluster. This estimation can be obtained from a domain expert or by using a training set of objects.

2. In the second phase, our algorithm searches for a clustering of the entire set of objects such that the attributes of the objects influence the clustering to the extent obtained in the first phase.

To verify the quality of the clustering obtained through our techniques and to expedite the first phase, we tested our algorithms on databases for which a classification of the set of objects is already known. For a broad class of databases, which we characterize in the paper, our approach yields results that are superior to clusterings obtained with classical clustering methods (such as k-means).

A clustering of the tuples of a table can be regarded as a partition of the set of tuples. Also, each set of attributes X of a table T determines a partition π_X^T of the set of tuples such that two tuples belong to the same block of the partition if they have equal projections on X. A dissimilarity between partitions of a set

*University of Massachusetts at Boston, Department of Computer Science, Boston, Massachusetts 02125, USA, e-mail: {dana, dsim}@cs.umb.edu

is defined using a generalization of conditional entropy. This dissimilarity helps us estimate the influence of the attributes on the clustering.

We obtain good results on databases for which there is a strong relationship between attributes and the "natural" clustering of the data, and we propose a method to determine if the given database is suitable or not to be well-classified by our approach.

2 Tables and Generalized Entropy

The set of partitions of a set R is denoted by $\mathsf{PART}(R)$. For $\pi, \sigma \in \mathsf{PART}(R)$ we write $\pi \leq \sigma$ if every class of σ is a union of classes of π.

Let $\mathcal{U}$ be a set whose elements are referred to as *attributes*. We assume that for every element A of $\mathcal{U}$ there is a set denoted by $\mathsf{Dom}(A)$ referred to as the *domain of* A such that $|\mathsf{Dom}(A)| \geq 2$.

A *table* is a function $T : R \times H \longrightarrow \bigcup D_H$, where R, a finite set, is referred to as the *set of rows of* T, H is a finite subset of $\mathcal{U}$ (called the *heading of* T), $D_H = \{\mathsf{Dom}(A) \mid A \in H\}$, and $T(r, A) \in \mathsf{Dom}(A)$ for every $r \in R$ and $A \in H$.

The *projection of the table* $T : R \times H \longrightarrow \bigcup D_H$ *on the set of attributes* $L \subseteq H$ is the table $T[L] : R \times L \longrightarrow \bigcup D_L$ given by $T[L](r, A) = T(r, A)$ for every $r \in R$ and $A \in L$. If r is a row of the table T, then the set $\{T(r, A) \mid A \in H\}$ is the *content* of the row r. This definition of a table allows for the existence of multiple rows having the same content. The *projection of the row* r *of the table* T *on* L is the set $r[L] = \{T(r, A) \mid A \in L\}$. For $A \in H$, $Q_T[A] = \{T(r, A) \mid r \in R\}$ represents the content of the column associated with the attribute A in the table T.

The *active domain of a set of attributes* L *in the table* T is the set $\mathsf{aDom}_T(L) = r[L]$. The cardinality of the active domain of L in the table T is denoted by a_L^T, or, when there is no risk for confusion, just by a_L.

Let $T : R \times H \longrightarrow \bigcup D_H$ be a table with n attributes and $H = A_1 A_2 \ldots A_n$. Every attribute set $L \subseteq H$ determines a partition $\pi_L^T = \{D_1, \ldots, D_{a_L}\}$ of the set R of rows in table T, where $\mathsf{aDom}_T(L) = \{v_1, \ldots, v_{a_L}\}$ and $D_i = \{r \in R \mid T(r, L) = v_i\}$ for $1 \leq i \leq a_L$. Clearly, if $L = \{A_{i_1}, \ldots, A_{i_\ell}\}$, then $\pi_L^T = \bigcap\{\pi_{A_{i_j}}^T \mid 1 \leq j \leq \ell\}$.

We introduce now a generalization of the notion of entropy that plays an essential role in defining partition dissimilarities.

Let $\mathbb{R}$ be the set of reals. The k-dimensional simplex is the set

$$\mathsf{SIMPLEX}_{k-1} = \{(p_1, \ldots, p_k) \in \mathbb{R}^k \mid p_i \geq 0 \text{ and } p_1 + \cdots + p_k = 1\}.$$

A function $f : \mathbb{R} \longrightarrow \mathbb{R}$ is *concave on a set* $S \subseteq \mathbb{R}$ if $f(\alpha x + (1-\alpha)y) \geq \alpha f(x) + (1-\alpha) f(y)$ for $\alpha \in [0, 1]$ and $x, y \in S$. The function f is *sub-additive* (*supra-additive*) on S if $f(x+y) \leq f(x) + f(y)$ $(f(x+y) \geq f(x) + f(y))$ for $x, y \in S$.

A *generator* is a concave, subadditive function $f : [0, 1] \longrightarrow \mathbb{R}$ such that $f(0) = f(1) = 0$ and

$$f(\theta a_1) + \cdots + f(\theta a_n) \leq \theta(f(a_1) + \cdots + f(a_n)) + f(\theta). \tag{1}$$

for every $(a_1, \ldots, a_n) \in \mathsf{SIMPLEX}_{n-1}$ and $\theta \in [0, 1]$.

The impurity measure induced by f is $i(p_1, \ldots, p_k) = f(p_1) + \cdots + f(p_k)$, for every $(p_1, \ldots, p_k) \in \mathsf{SIMPLEX}_{k-1}$.

It is easy to verify that such functions as $f_{\rm gini}(p) = p - p^2$ (the Gini index), $f_{\rm sq}(p) = \sqrt{p} - p$, $f_{\rm ent}(p) = -p \log p$ (the Shannon entropy), or $f_{\rm peak}$ (given by $f_{\rm peak}(p) = p$ for $0 \leq p \leq 0.5$ and $f_{\rm peak}(p) = 1 - p$ for $0.5 < p \leq 1$) are generators.

Definition 2.1 Let f be a generator, R be the set of rows of a table T, and let $\pi = \{B_1, \ldots, B_n\}$, $\sigma = \{C_1, \ldots, C_m\}$ be two partitions of R.

The *f-impurity of partition* π (called also the *f-entropy* of π) is $\mathcal{H}^f(\pi) = \sum_{i=1}^n f\left(\frac{|B_i|}{|R|}\right)$.

The *specific f-impurity of a subset* L *of* R *relative to a partition* π is the quantity

$$\mathsf{imp}_\pi^f(L) = f\left(\frac{|L \cap B_1|}{|L|}\right) + \cdots + f\left(\frac{|L \cap B_n|}{|L|}\right).$$

The *f-impurity of π relative to σ* is

$$\mathcal{H}^f(\pi|\sigma) = \sum_{j=1}^{m} \frac{|C_j|}{|R|}\mathsf{imp}^f_\pi(C_j) = \frac{1}{|R|}\sum_{j=1}^{m}|C_j|\sum_{i=1}^{n} f\left(\frac{|B_i \cap C_j|}{|C_j|}\right).$$

This quantity is also called the *f-conditional entropy of π relative to σ.*

In other words, the conditional entropy $\mathcal{H}^f(\pi|\sigma)$ is the average value of the specific impurity of the classes of the partition σ relative to the partition π.

We have $\sigma \leq \pi$, if and only if $\mathcal{H}^f(\pi|\sigma) = 0$.

If $\pi, \sigma \in \mathsf{PART}(R)$, then

$$\mathcal{H}^f(\pi) \geq \mathcal{H}^f(\pi|\sigma) \geq \mathcal{H}^f(\pi \wedge \sigma) - \mathcal{H}^f(\sigma).$$

If $\pi_1, \pi_2, \sigma \in \mathsf{PART}(R)$ are such that $\pi_1 \leq \pi_2$, then

$$\mathcal{H}^f(\pi_1|\sigma) \geq \mathcal{H}^f(\pi_2|\sigma).$$

Similarly, if $\pi, \sigma_1, \sigma_2 \in \mathsf{PART}(R)$ are such that $\sigma_1 \leq \sigma_2$, then

$$\mathcal{H}^f(\pi|\sigma_1) \leq \mathcal{H}^f(\pi|\sigma_2).$$

This allows us to conclude that if $\pi_1, \pi_2, \sigma \in \mathsf{PART}(R)$ are such that $\pi_1 \leq \pi_2$, then

$$\mathcal{H}^f(\pi_1) \geq \mathcal{H}^f(\pi_2).$$

3 Clustering using Genetic Algorithms

Let $T : R \times H \longrightarrow \bigcup D_H$ be a table. A *training table for T* is a table $T_s : R_s \times (H \cup \{C\}) \longrightarrow \bigcup D_H \cup D_C$, where C is an additional attribute that specifies the class of each row of R_s.

If $R_s \subseteq R$, then T_s is referred to as a *sample table.*

The training table can be used to determine the influence of each attribute on the clustering. We adopt as a measure of this influence the weight

$$w_{\pi_C^{T_s}}(A) = \mathcal{H}^f(\pi_A^{T_s}|\pi_C^{T_s}) + \mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s}).$$

The weight $w_{\pi_C^{T_s}}(A)$ captures the dissimilarity between the partitions $\pi_A^{T_s}$ and $\pi_C^{T_s}$ and thus, the influence of attribute A on the classes of the "natural" clustering. Attributes with small values of $w_{\pi_C^{T_s}}(A)$ have a larger impact on the "natural" clustering. This is justified by the fact that small values for $w_{\pi_C^{T_s}}(A)$ indicate that each class of the partitions $\pi_A^{T_s}$ and $\pi_C^{T_s}$ has a high degree of "purity" relative to the other partition.

Our goal is to find a partition π of the set of rows R, that has no more classes that a certain prescribed limit k, such that the weights of the attributes $w_\pi(A) = \mathcal{H}^f(\pi_A^T|\pi) + \mathcal{H}^f(\pi|\pi_A^T)$ are as close as possible to the estimated weights $w_{\pi_C^{T_s}}(A)$.

A *k-chromosome* on a table $T : R \times H \longrightarrow \bigcup D_H$ is a function $K : R \longrightarrow \{1, \ldots, k\}$. An element of the set $\{1, \ldots, k\}$ is called a *class identifier.*

The partition π_K^T of the set of rows R determined by the k-chromosome K is $\pi_K^T = \{B_1, \ldots, B_k\}$, where $B_j = \{r \in R \mid K(r) = j\}$ for $1 \leq j \leq k$.

The chromosomes represent possible partitions of the set of rows R into k classes. The population of chromosomes consists of M k-chromosomes, where M is another input parameter.

The idea of genetic evolution is to modify the chromosomes in the current population by using mutation and crossover as genetic operators such that in the new population we have chromosomes that will be increasingly closer to the partition determined by the "natural" clustering of the data.

We use the classical single point crossover operator. Starting from two chromosomes, a random crossing point (called a *crossover site*) is selected as a number l between 1 and N (the number of rows in the table T).

The offsprings will contain the first 1 to l positions from the first parent and the last $l+1$ to N positions from the second parent and vice versa. Also, the classical mutation operator which involves changing randomly a number of $\max\{1, 0.1N\}$ positions in the chromosomes is used. The new value for each chromosome position is chosen randomly from 1 to k.

The input parameters of the genetic algorithm are summarized in Figure 1.

M	cardinality of the chromosomial population
N	number of rows in the table T
n	number of columns in the table T
k	number of classes in the median partition
r	percent of chromosomes that are used for crossover
m	percent of chromosomes that undergo a mutation
N_{max}	maximum number of consecutive iterations without improvement
ϵ	margin error for the fitness function

Figure 1: Genetic algorithm input parameters

Beside these parameters, the genetic algorithm is characterized also by the measure fitness(K), used in assessing the fitness of chromosomes, that is discussed in the next section.

If in the chromosomial population there exists a chromosome with sufficiently large fitness, then the algorithm stops and produces the natural clustering generated by this chromosome.

The initial population of chromosomes is filled randomly with values between 1 and k. For each chromosome K we compute the value fitness(K). In each population exists a chromosome which has the largest fitness(K) value. We denote this chromosome by K_{best} and its fitness by $\text{fitness}_{\text{best}}$.

If in N_{max} consecutive iterations there is no improvement in the best fitness, then the algorithm will halt and the resulting partition is the one corresponding to K_{best}.

If the above exit condition does not hold, a new population is computed using the elite selection (see [Mic99]). The fittest $(1 - r - m)M$ chromosomes are copied directly in the new population, since the fitness value indicates how close is the chromosome partition to the "natural" clustering of the data, and we want to preserve the best partitions discovered so far. This selection ensures that we will never lose the best chromosome from the old population.

Next, a number of $\max\{2, rM\}$ chromosomes from the old generation are selected probabilistically, to be used in generating new offsprings by crossover. The selection method used is a roulette wheel strategy (or fitness-proportionate selection) in which the chromosomes having greater values of their fitness have also a greater chance of being selected. The newly generated offsprings are copied into the new population.

Finally, a number of $\max\{1, mM\}$ chromosomes are selected with uniform probability and subjected to mutation. The mutation operator is not biased toward the fittest chromosomes and therefore the chromosomes that will suffer a mutation are selected with uniform probability.

The pseudocode of the algorithm is the following:

```
initialize the population of genetic algorithm
while (true)
    compute the fitness of chromosomes in the population;
    if (there has been no relative
      improvement in best fitness value for $N_{max}$ iterations)
      then
          output the partition of $K_{\text{best}}$;
          exit;
    copy fittest $(1 - r - m)M$ chromosomes to new population;
    select probabilistically $\max\{2, rM\}$ chromosomes to cross over;
    apply crossover operator to the selected chromosomes
      and copy the offsprings to the new population;
    select with uniform probability $\max\{1, mM\}$ chromosomes to mutate;
```

```
apply mutation operator to the selected chromosomes
  and copy the modified chromosomes to the new population;
replace old population by the new one;
```

3.1 Genetic Algorithm Fitness Measure

The fitness of a chromosome K is based on the similarity between the weights $w_{\pi_K^T}(A)$ of attributes $A \in H$ and the estimated weights $w_{\pi_C^{T_s}}(A)$, where π_K^T is the partition determined by each chromosome K. The closer the weights $w_{\pi_K^T}(A)$ are to $w_{\pi_C^{T_s}}(A)$, the closer the partition π_K^T is to the "natural clustering" of the data. Thus, we seek to determine a partition π of the set of rows R of the table T such that:

$$\frac{w_\pi(A_1)}{w_{\pi_C^{T_s}}(A_1)} = \frac{w_\pi(A_2)}{w_{\pi_C^{T_s}}(A_2)} = \ldots = \frac{w_\pi(A_n)}{w_{\pi_C^{T_s}}(A_n)} = \frac{\sum_{A\in H} w_\pi(A)}{\sum_{A\in H} w_{\pi_C^{T_s}}(A)}.$$

If we normalize the values $w_\pi(A)$ by replacing them with

$$w_\pi(A) = \frac{w_\pi(A)}{\sum_{A\in H} w_{\pi_C^{T_s}}(A)}$$

the last series of equalities can be written as:

$$\frac{w_\pi(A_1)}{w_{\pi_C^{T_s}}(A_1)} = \frac{w_\pi(A_2)}{w_{\pi_C^{T_s}}(A_2)} = \ldots = \frac{w_\pi(A_n)}{w_{\pi_C^{T_s}}(A_n)} = \sum_{A\in H} w_\pi(A).$$

We denote by $\mathcal{W}^f(\pi) = \sum_{A\in H} |w_\pi(A) - w_{\pi_C^{T_s}}(A) \times \sum_{A\in H} w_\pi(A)|$. The smaller the quantity $\mathcal{W}^f(\pi)$, the closer the partition π is to the "natural" clustering of the data. As fitness measure we use the measure:

$$\text{fitness}^f_{\mathcal{W}}(K) \quad = \quad \frac{\max\{\mathcal{W}^f(\pi_{K_i}^T) | 1 \le i \le n\}}{\mathcal{W}^f(\pi_K^T)},$$

for every chromosome K, whose associated partition is π_K^T.

3.2 Training for Clustering

Our clustering approach is suitable for databases that have a strong relationship between the attribute partitions and the "natural" clustering of the data. To evaluate this relationship, we compute the weights $w_{\pi_C^{T_s}}(A)$ from the training table T_s. The value $w_{\pi_C^{T_s}}(A)$ consists of the sum of two quantities $\mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s})$ and $\mathcal{H}^f(\pi_A^{T_s}|\pi_C^{T_s})$. We will focus on the first quantity $\mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s})$ that represents the generalized conditional entropy between the partitions $\pi_A^{T_s}$ and $\pi_C^{T_s}$. We have $0 \le \mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s}) \le \mathcal{H}^f(\pi_C^{T_s})$. The closer this value is to 0, the closer the partition $\pi_A^{T_s}$ is to the partition $\pi_C^{T_s}$. The closer this value to $\mathcal{H}^f(\pi_C^{T_s})$, the more distant the partition $\pi_A^{T_s}$ is from the partition $\pi_C^{T_s}$. Thus, when the expression $\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s})$ is closer to 0, the importance of attribute A in the finding of the natural clustering of the data is smaller. Another interpretation of the value $\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s})$ is the following: if attribute A has no relationship with the natural classes, then knowing the class of the attribute A brings no information on the class of the partition $\pi_C^{T_s}$, and in this case $\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s})$ is close to 0.

Thus, given a training database, we compute for each attribute the values $\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s})$ and $\text{ind}_{T^s} = \frac{\max_{A\in H}\{\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s})\}}{\mathcal{H}^f(\pi_C^{T_s})}$. This last quantity is an indicator of how suitable is the given database to the proposed clustering algorithm. If the value ind_{T^s} is very small, than there is no point in applying our algorithm, since there is no relationship between the attribute partitions and the partition associated with the "natural" clustering of the data.

4 Experimental results

We tested the proposed clustering algorithm on different categorical databases from the UCI Machine Learning Repository (see [BM98]). These databases have an attribute, denoted *target* attribute, whose values represent the classes of the "natural" clustering of the rows. We extracted the sample database, from the input database, by randomly choosing a certain number of rows, computed in terms of an input parameter of our algorithm, which specifies the percent of the original database to be used for sampling. In all our experiments, we used a sample database of 40% of the input database.

To test the proposed algorithm we first analyzed the datasets and then we continued with the search for the "natural" clustering only on the suitable databases.

Using the sample database, we estimate the relationship between the partition determined by each attribute and the partition determined by the *target* attribute. As it was explained in the previous section, for each attribute A different than the *target* attribute, we compute the value $\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s})$ and the value $w_{\pi_C^{T_s}}(A) = \mathcal{H}^f(\pi_A^s|\pi_C^{T_s}) + \mathcal{H}^f(\pi_C^{T_s}|\pi_A^s)$. The larger the value $\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s}|\pi_A^{T_s})$, the stronger the relationship between attribute A and the *target* attribute, and the more important is attribute A in the search for the "natural" clustering.

attribute and its value $\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s}\|\pi_A^{T_s})$			
hair	0.258	feathers	0.224
eggs	0.264	milk	0.281
airborne	0.112	aquatic	0.110
predator	0.03	toothed	0.203
backbone	0.117	breathes	0.161
venomous	0.031	fins	0.185
legs	0.465	tail	0.095
domestic	0.016	catsize	0.115

Figure 2: Estimated attribute influence for zoo.data

The first experiment involved the database *zoo.data* (from the UCI Repository), that contains the records of 101 animals, characterized by 17 attributes and classified in 7 classes. We removed the *name* attribute, since it has unique values for each row in the database, and it does not bring any useful information to the clustering process.

attribute and its value $\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s}\|\pi_A^{T_s})$					
date	0.296	plant-stand	0.157	precip	0.232
temp	0.309	hail	0.040	crop-hist	0.05
area-damaged	0.264	severity	0.117	seed-tmt	0.018
germination	0.101	plant-growth	0	leaves	0.271
leafspots-halo	0	leafspots-marg	0	leafspot-size	0
leaf-shread	0	leaf-malf	0	leaf-mild	0
stem	0	lodging	0.046	stem-cankers	0.507
canker-lesion	0.507	fruiting-bodies	0.196	external decay	0.158
mycelium	0.121	int-discolor	0.136	sclerotia	0.136
fruit-pods	0.209	fruit-spots	0	seed	0
mold-growth	0	seed-discolor	0	seed-size	0
shriveling	0	roots	0.212		

Figure 3: Estimated attribute influence for soybean-small.data

The results of the analysis process for *zoo.data* are summarized in Figure 2. Note that attribute *legs* plays the most important role in the classification of animals and attribute *domestic* has the weakest influence on determining the class of an animal. For this dataset, the value $ind_{\text{zoo.data}} = \frac{0.465}{0.777} = 0.59$ reflects a strong relationship between attribute partitions and the *target* partition.

attribute and its value $\mathcal{H}^f(\pi_C^{T_s}) - \mathcal{H}^f(\pi_C^{T_s} \mid \pi_A^{T_s})$			
handicapped infants	0.063	water project cost sharing	0.069
adoption of the budget resolution	0.006	physician fee freeze	0.299
el-salvador aid	0.403	religious groups in schools	0.223
anti satellite test ban	0.093	aid to nicaraguan contras	0.120
mx-missile	0.165	immigration	0.177
synfuels corporation cutback	0.011	education spending	0.054
superfund right to sue	0.221	crime	0.105
duty-free exports	0.147	export administration act south-africa	0.089

Figure 4: Estimated attribute influence for house-votes-84.data

Next, we analyzed *soybean-small.data*, documenting 4 soybean diseases, having 47 rows and 35 attributes, including also the *target* attribute. The estimated attribute influences in the clustering process are summarized in Figure 3. Attributes *leafspots-halo, leafspots-marg, leafspot-size, leaf-shread, leaf-malf, leaf-mild, stem, fruit-spots, seed, mold-growth, seed-discolor, seed-size, shriveling* have only one value in the sample database, so they play no role in the clustering process and in Figure 3 we see that their influence value is 0. Attributes *stem-cankers* and *canker-lesion* play the most important role in determining the type of soybean disease. The value $ind_{\text{soybean-small.data}} = \frac{0.507}{0.685} = 0.74$ indicates also a strong connection between attributes and the *target* attribute.

We also analyzed the database *house-votes-84.data*, documenting Congressional Voting Records in 1984 and having 435 votes on 16 issues. Each row is characterized by a classification label of democrat or republican. The results of the analysis are summarized in Figure 4. Attribute *el-salvador-aid* has the greatest importance in the democrat/republican classification and attribute *adoption-of-the-budget-resolution* the smallest importance. The value $ind_{\text{house-votes-84.data}} = \frac{0.403}{0.483} = 0.83$ reflects a strong connection between attributes and the target attribute.

The second phase of our algorithm represents the actual search for the partition reflecting the "natural" clustering of the data. In this phase, we applied the proposed clustering method based on a genetic algorithm approach, to the input databases, from which we eliminated the *target* attribute. We denote by π_{target} the partition associated with the *target* attribute as it appears in the input database.

For all the following experiments, the parameters of the genetic algorithm are as follows. The crossover and mutation rate are set to 0.8 and 0.1 respectively. The number of consecutive iterations without improvement is set to 500.

For each dataset, we executed 10 times our genetic algorithm using the Gini index and the Shannon entropy, and the k-means clustering algorithm from the WEKA package, for each experiment using different values for the random number generator. We searched for partitions π_k with $|\pi_k| = k$ classes with $k = 7$ for *zoo.data*, $k = 4$ for *soybean-small.data* and $k = 2$ for *house-votes-84.data*.

The results for the *zoo.data* with 100 chromosomes are presented in Figure 5, where we specify the average classification rate obtained over all 10 runs, and characteristics of the best partition found, the number of classes, the associated classification rate and the number of elements classified as in the *target* attribute.

The best classification rate obtained for the Gini index is 0.84, obtained for a partition that grouped together 39 out of 41 mammals, all 20 birds, all 13 fishes, all 8 insects, 2 out of 4 amphibians and 3 out of 10 invertebrates. For Shannon entropy the best partition grouped together 39 out of 41 mammals, all 20 birds, all 13 fishes, all 10 invertebrates and 7 out of 8 insects, leading to a classification rate of 0.89. The best clustering obtained by k-means classified correctly 85 out of 101 animals, leading to a classification rate of 0.84.

The best result of k-means is similar to the best result obtained using the Gini index, and is worst than the best result obtained using the Shannon entropy. Overall, in these experiments our approach found a more

Gini index as generator	
Average classification rate: 0.805	
Best Partition Characteristics	
Cardinalities: 40 23 16 13 4 4 1	
Classification rate: 0.84	
Elements classified as in π_{target}:	
39 out of 41	13 out of 13
20 out of 20	3 out of 10
8 out of 8	2 out of 4

Shannon entropy as generator	
Average classification rate: 0.783	
Best Partition Characteristics	
Cardinalities: 39 20 20 11 9 2	
Classification rate: 0.89	
Elements classified as in π_{target}:	
39 out of 41	13 out of 13
20 out of 20	10 out of 10
7 out of 8	1 out of 5

k-means	
Average classification rate: 0.754	
Best Partition Characteristics	
Cardinalities: 39 19 18 13 6 6	
Classification rate: 0.84	
Elements classified as in π_{target}:	
39 out of 41	13 out of 13
19 out of 20	4 out of 10
6 out of 8	4 out of 4

Figure 5: Results for zoo.data

robust clustering. The average classification rate is 0.805 for the Gini index, 0.783 for the Shannon entropy, and 0.754 for k-means. Thus, we can conclude that on average our algorithm finds a better clustering of the input data.

We repeated the previous tests for *soybean-small.data*, with the number of chromosomes set to 50. The results for Gini index, Shannon entropy and k-means are presented in Figure 6.

For the Gini index and the Shannon entropy algorithms, the classification rate was 0.97 (only one row was misclassified) and was obtained in all 10 experiments with a single exception in the case of the Shannon entropy. The best k-means clustering has a classification rate of 1. The average classification rate is 0.97 for Gini index, 0.945 for Shannon entropy and 0.882 for k-means. Again, we can conclude that on average our algorithm performed better than k-means.

Last, we repeated our experiments on *house-votes-84.data*, with the number of chromosomes set to 100. The results are presented in Figure 7.

On this dataset the best classification rate obtained using the Gini index was 0.91, and resulted in a partition which grouped together 160 out of the 168 republicans, and 237 out of 267 democrats, so only 38 rows were misclassified. The best partition for Shannon entropy as generator had a classification rate of 0.92, it grouped together 154 out of the 168 republicans and 250 out of 267 democrats, leading to only 31 rows misclassified. The best partition found by the k-means algorithm had a classification rate of 0.87, since 54 rows are misclassified. The average classification rate is 0.904 for Gini index, 0.911 for Shannon entropy and 0.860 for k-means. We can conclude again that our algorithm performed better than k-means.

5 Conclusions

We proposed a clustering method based on a genetic algorithm, using information theoretical measures to determine the quality of the resulting clustering. Our algorithm involves two phases. The first phase consists in estimating the degree of influence of each attribute on the "natural" classification of the rows of the database; the second phase consists in searching for a partition that provides the same degree of influence of the attributes. We showed that the generalized entropy is a powerful tool in assessing the similarity between two partitions, and that we obtained better results than the classical k-means algorithm for both Gini index and Shannon entropy generators. Our proposed method produces best results for databases for which there

Gini index as generator	
Average classification rate: 0.97	
Best Partition Characteristics	
Cardinalities: 16 11 10 10	
Classification rate: 0.97	
Elements classified as in π_{target}:	
10 out of 10	10 out of 10
10 out of 10	16 out of 17

Shannon entropy as generator	
Average classification rate: 0.945	
Best Partition Characteristics	
Cardinalities: 17 11 10 9	
Classification rate: 0.97	
Elements classified as in π_{target}:	
10 out of 10	10 out of 10
9 out of 10	17 out of 17

k-means	
Average classification rate: 0.882	
Best Partition Characteristics	
Cardinalities: 17 10 10 10	
Classification rate: 1	
Elements classified as in π_{target}:	
10 out of 10	10 out of 10
10 out of 10	17 out of 17

Figure 6: Results for soybean-small.data

is a strong relationship between attributes and the "natural" clustering of the data. To determine this relationship we proposed a measure based also on a generalized conditional entropy. This measure can be used also as an indication of how useful is an attribute in the process of determining the class associated with a row. Attributes that have small influence in the natural clustering can be ignored from the clustering process, and thus we have a method of reducing the dimensionality of the data without diminishing the quality of the clustering. In our experiments we did not ignore the weakest influence attributes, but this can be done easily, by adding a minor modification to the proposed algorithm.

References

[BM98] C.L. Blake and C.J. Merz. Uci repository of machine learning databases, 1998. http://www.ics.uci.edu/~mlearn/MLRepository.html.

[ECM97] V. Estivill-Castro and A. Murray. Spatial clustering for data mining with genetic algorithms. Technical Report FIT-TR-97-10, Queensland University of Technology, 1, 1997.

[Fal99] Emanuel Falkenauer. *Genetic algorithms and grouping problems.* Jon Wiley, New York, 1999.

[HGV97] Han Eui-Hong (Sam) Han, Karypis George, and Kumar Vipin. Clustering based on association rule hypergraph. *Proc. Workshop on Research Issues on Data Mining and Knowledge Discovery*, 1997.

[Hol92] John H. Holland. *Adaptation in Natural and Artificial Systems.* The MIT Press, Cambridge, Massachusetts, 1992.

[Hua97] Zhexue Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In *Research Issues on Data Mining and Knowledge Discovery*, pages 0–, 1997.

[JB91] Donald R. Jones and Mark A. Beltramo. Solving partitioning problems with genetic algorithms. *Proceedings of the fourth International Conference on Genetic Algor hms*, pages 442–449, 1991.

[Mic99] Zbigniew Michalewicz. *Genetic algorithms + Data Structures = Evolution Programs.* Springer, New York, 1999.

[Mit97] Tom M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[SCC00a] Dan Simovici, Dana Cristofor, and Laurentiu Cristofor. Generalized entropy and projection clustering of categorical data. Technical Report 00-3, University of Massachusetts, Boston, Massachusetts, 2000.

[SCC00b] Dan Simovici, Dana Cristofor, and Laurentiu Cristofor. Mining for purity dependencies in databases. Technical Report 00-2, University of Massachusetts, Boston, Massachusetts, 2000.

Gini index as generator
Average classification rate: 0.904
Best Partition Characteristics
Cardinalities: 245 190
Classification rate: 0.91
Elements classified as in π_{target}:
160 out of 168   237 out of 267

Shannon entropy as generator
Average classification rate: 0.911
Best Partition Characteristics
Cardinalities: 264 171
Classification rate: 0.92
Elements classified as in π_{target}:
154 out of 168   250 out of 267

k-means
Average classification rate: 0.860
Best Partition Characteristics
Cardinalities: 226 209
Classification rate: 0.87
Elements classified as in π_{target}:
162 out of 168   220 out of 267

Figure 7: Results for house-votes-84.data

[SRK99] Guha Sudipto, Rastogi Rajeev, and Shim Kyuseok. Rock: A robust clustering algorithm for categorical attributes. *Proceedings of the IEEE International Conference on Data Engineering, Sydney, March 1999*, March 1999.

[VJR99] Ganti Venkatesh, Gehrke Johannes, and Ramakrishnan Raghu. Cactus - clustering categorical data using summaries. *KDD-99 San Diego CA USA*, 1999.