ELSEVIER

Intelligent Data Analysis 2 (1998) 265–286

IDA
INTELLIGENT DATA ANALYSIS

www.elsevier.com/locate/ida

ARCADE: A Prediction Method for Nominal Variables

J.F.P. Costa [a,*], I.C. Lerman [b,1]

[a] *Departamento de Matemática Aplicada (Fac. de Ciências) LIACC, Univ. do Porto, Rua das Taipas, 135, 4050 Porto, Portugal*

[b] *IRISA-INRIA-Rennes, France*

Received 28 February 1998; revised 3 May 1998; accepted 28 May 1998

Abstract

The main problem considered in this paper consists of binarizing categorical (nominal) attributes having a very large number of values (20^4 in our application). A small number of relevant binary attributes are gathered from each initial attribute. Let us suppose that we want to binarize a categorical attribute v with L values, where L is large or very large. The total number of binary attributes that can be extracted from v is $2^{L-1}-1$, which in the case of a large L is prohibitive. Our idea is to select only those binary attributes that are predictive; and these shall constitute a small fraction of all possible binary attributes. In order to do this, the significant idea consists in grouping the **L** values of a categorical attribute by means of an hierarchical clustering method. To do so, we need to define a similarity between values, which is associated with their predictive power. By clustering the **L** values into a small number of clusters (J), we define a new categorical attribute with only **J** values. The hierarchical clustering method used by us, AVL, allows to choose a significant value for **J**. Now, we could consider using all the $2^{L-1}-1$ binary attributes associated with this new categorical attribute. Nevertheless, the J values are tree-structured, because we have used a hierarchical clustering method. We profit from this, and consider only about $2 \times \mathbf{J}$ binary attributes. If **L** is extremely large, for complexity and statistical reasons, we might not be able to apply a clustering algorithm directly. In this case, we start by "factorizing" v into a pair (v^2, v^2), each one with about $\sqrt{L(v)}$ values. For a simple example, consider an attribute v with only four values m_1, m_2, m_3, m_4. Obviously, in this example, there is no need to factorize the set of values of v, because it has a very small number of values. Nevertheless, for illustration purposes, **v** could be decomposed (factorized) into 2 attributes with only two values each; the correspondence between the values of **v** and (v^2, v^2) would be

v	$(v^1,$	$v^2)$
m_1	1	1
m_2	1	2
m_3	2	1
m_4	2	2

Now we apply the clustering method to both sets of values of v^1 and v^2, defining therefore a new synthetic pair $(\tilde{v}^1, \tilde{v}^2)$. Then, we "multiply" these new attributes and get another attribute v^{10} with $J \times J$ values; J_1 (resp. J_2) is the number of values

* Corresponding author. E-mail: jpcosta@ncc.up.pt.

[1] E-mail: lerman@irisa.fr.

PII: S1088-467X(98)00031-6

of $\bar{v}^1$(resp. $\bar{v}^2$). Now, we apply a final clustering to the values of v^{10}, and proceed as above. The solution that we propose is independent of the number of classes and can be applied to various situations. The application of ARCADE to the protein secondary structure prediction problem, proves the validity of our approach.

Keywords: Decision trees; Binarization; Complexity reduction; Categorical attributes; Hierarchical clustering

1. Introduction

Binary decision trees are a representative logic tool for concept discrimination. The methodological and inferential problems related to their construction concern two domains: statistics and machine learning [3,30,33,39]. Their construction, which depends on a set **A** of binary description attributes, is based on a learning set **O**. The concept (or class) to be recognised is expressed as an attribute **c** taking **K** different values, where **K** is not necessarily very small.

Predictive descriptive attributes are seldom binary at the beginning. Although binarization is often an easy task (for ordered variables or nominal with a small number of values), in our application this is not at all the case. Let $E(v)$ be the set of values of an attribute **v**. A fundamental research issue has been how to construct two-class partitions of $E(v)$, in order to obtain the binary attributes associated with **v**. This construction must depend on the structure (or semantics) of $E(v)$. Lately, the concern has mainly been on finding ways to discretize the continuous attributes [3,8,9,13,14,29, 40]. In the methodological developments that we consider in this article, each predictive attribute **v** is qualitative nominal (categorical) and takes a very large number of different values. In other words, $L = L(v) = card[E(v)]$ is large and in our data it can reach 20^4 values for each predictive attribute **v**. Then, even a complexity $O[L \cdot \log(L)]$, as reached in the program CART [3] for $\mathbf{K} = 2$ classes, is not acceptable in our case. Nevertheless, as we shall see, this is not the main reason why CART is not applicable to our data.

Our method, ARCADE, by grouping the values of each predictive attribute **v** into clusters, allows the definition of another qualitative attribute with much fewer values. For example, let us consider an attribute **v** with 10 values $m_1, m_2, \ldots, m_{10}$. By grouping these values into four clusters, like for instance $\{m_1, m_5, m_8\}$, $\{m_2, m_3\}$, $\{m_4, m_6, m_7\}$, $\{m_9, m_{10}\}$, we can define a new nominal (categorical) attribute, $\bar{v}$, with four values. To this new attribute we associate $2^{4-1} - 1 = 7$ binary attributes, each one corresponding to a bi-partition of the 4 new values. In order to make significant these latter binary attributes, the values of the attribute **v** grouped in the same cluster have to be similar with respect to their statistical behaviour on the classes to be predicted. To achieve this, we start by establishing a contingency table, which crosses the predictive attribute **v** with the attribute **c** to be predicted. Then, the application of an automatic clustering method to the rows of this contingency table, which represent the values of **v**, allows the definition of a new categorical attribute $\bar{v}$. Let us now denote by $E(v)$ the number of values of $L(\bar{v})$. The number of binary splittings of $\bar{v}$ that have to be considered in the decision tree construction is $(2^{L-1} - 1)$. This number is equal to 7 in the above example. It becomes too large in the case where $L(\bar{v})$ is not small enough.

For reasons that we shall explain later, instead of a non-hierarchical clustering approach, we apply a hierarchical clustering method to the rows of the contingency table. The binary splittings that are retained are then associated with the leaves or nodes of the clustering tree on the set of values of the new categorical variable $\bar{v}$. This technique reduces very considerably the complexity of finding "good" predictive attributes.

In our application the number of rows is too large (20^4), and so we can not apply directly the clustering technique. The direct utilisation of a clustering method on the row set of the contingency table becomes—for complexity and statistical reasons—neither tractable nor significant. To solve this problem we start by **factorizing** an attribute **v** with $L(v)$ values into a pair of two simpler attributes, (v^1, v^2), each one with about $\sqrt{L(v)}$ values. For a simple example, consider an attribute **v** with only four values m_1, m_2, m_3, m_4. Obviously, in this example, there is no need to factorise the set of values of **v**, because it has a very small number of values (4). Nevertheless, for illustration purposes, **v** could be decomposed (factorized) into 2 attributes with only two values each; the correspondence between the values of **v** and (v^1, v^2) would be:

v	$(v^1$	$v^2)$
m_1	1	1
m_2	1	2
m_3	2	1
m_4	2	2

Fig. 1. Simple example of the factorization procedure.

Each of the latter attributes v^1 and v^2 gives rise to a contingency table with $\sqrt{L(v)}$ rows. Therefore, by applying a clustering method to the rows of both contingency tables, the pair of attributes (v^1, v^2) is reduced to a pair of synthetic attributes that we denote by $(\bar{v}^1, \bar{v}^2)$. In order to get back as much as possible our initial attribute **v**, we "multiply" these two synthetic attributes; that is, we consider their joint distribution. This operation of **"multiplication"** of categorical attributes is the inverse of the factorization operation. It allows the definition of a new attribute, $\bar{v}^{1.2}$, that is more informative than the two attributes taken separately. Finally, we consider the contingency table that crosses the set of values of $\bar{v}^{1.2}$ with the set of values of the attribute to be predicted, **c**.

In the construction of a binary decision tree, at each node, one must choose a binary attribute to split that node into two descendent nodes. This choice is based on a measure of association, or coefficient, between the binary attribute and the concept (attribute) to be predicted. It has already been demonstrated [4,7,22,28], that a "good coefficient" must be used. The ARCADE method presents a large choice of coefficients, including the Gini coefficient (used in CART), the information quantity of Shannon, and the Lerman coefficients [19,20]. Costa [7] has demonstrated that the use of the Gini coefficient in decision trees is equivalent to the use of the coefficients of Goodman and Kruskal [12], Light and Margolin [27]. The fact that these coefficients have been conceived to predict, is another explanation for the good results that are usually obtained with the Gini coefficient.

We will give now a brief description of our data.

1.1. Prediction of Protein Secondary Structure

The primary structure of a protein is formally a word whose letters are taken from an alphabet of 20 letters (corresponding to the 20 amino acids $A, C, \ldots, Y$). The length of such a word can reach many hundred letters. The associated secondary structure of a protein can also be formally defined as a word of the same length. The corresponding alphabet contains three letters, **E**, **H** and **X**, which are the names of the three concepts to be discriminated (**H** corresponds to an **helix alpha**, **E** to a **beta sheet** and **X** stands for **turn**):

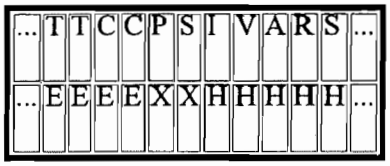

Fig. 2. Primary and secondary structures of a protein.

Many researchers have attacked this prediction problem over the last two decades, although we are not aware of anyone using decision trees. For instance, more recently, Qian and Sejnowsky (1988) [32] have used neural networks for treating this problem; Rost and Sander (1993) [35] have also used neural networks, and they have reached, for the first time, 70.1% of correct classifications; Cost and Salzberg (1993) [6] have used nearest-neighbor methods; Zhang, Mesirov and Waltz (1993) [42] have used a combined neural-net/nearest-neighbor/Bayesian approach; Solovyev and Salamov (1994) [37] have used four linear discriminant functions for predicting secondary structure segments, instead of predicting one residue at a time.

The protein structure prediction problem is a very difficult one. As Friesner and Gunn (1996) [11] pointed out, the determination of protein structure from amino acid sequence via computational methodology is a major goal of theoretical molecular biology. However, there are still formidable problems, both conceptual and numerical, in making this objective a reality. Biologists generally believe that protein secondary structure is a global property of the fold, emerging from energy considerations. Protein folding is a highly co-operative phenomenon in which the final secondary structure pattern and the tertiary architecture are determined by global optimisation of the hydrophobic effect and backbone hydrogen bonding. The accuracy of the prediction of secondary structure, from amino acid sequences, is at present on the order of 70%. Reliably predicting up to 80% of the secondary structure would be quite an achievement.

Although the number of known primary sequences increases extremely fast, the same is not true for the secondary structure. So, there are a widening number of protein sequences whose secondary structure needs to be predicted. This is not a difficult problem when there are proteins, of known secondary structure, which are homologous to the protein being predicted. Nevertheless, for 85% of the new proteins, there does not exist homologues; and the problem becomes very difficult. Most of the existing prediction methods try, for a *description* of each letter of the first word (primary structure), to predict the corresponding letter of the second word (secondary structure). Yet, it is crucial to find a pertinent description of the data for this difficult recognition problem. Indeed, the prediction of the letter in the second word does not depend only on the corresponding position in the first word, but also on its neighborhood.

The predictive attributes that we have adopted will be defined in Section 3. We will describe in this Section the different stages of the application of the ARCADE method to this difficult prediction problem. As a matter of fact, it was in the course of solving this problem that we have set up the methodology presented in this paper. This has been validated by the obtained results.

More autonomous and formal definition of ARCADE will be presented in Section 2. Section 4 will be devoted to conclusions and perspectives and also to a comparison with CART method in terms of computational complexities.

In order to aggregate the rows of a contingency table, according to a hierarchical scheme, the AVL ("Analyse de la Vraisemblance des Liens") hierarchical clustering method is employed. This method is extremely general with respect to the logical or mathematical structure of the data to be organised, according to their resemblances.

2. The ARCADE Method

2.1. Introduction

The ARCADE method is a hybrid of the CART method [3]. It consists mainly in a binarization technique of predictive categorical attributes; this binarization technique is particularly useful in the case where the categorical attributes have a very large number of values (modalities, categories). This hybridization includes also a large family of splitting criteria [7,22]. The binarization procedure can be decomposed into five steps:

1. Factorization of the predictive categorical attribute **v** into two or more than two factors.
2. Clustering of the set of values of each "factor" variable on the basis of the contingency table, crossing this factor with the categorical attribute to be predicted, **c**. This clustering reduces the number of values of each factor variable.
3. "Multiplication" of the reduced factors and setting up the contingency table which crosses the "multiplied" attribute with the attribute to be predicted.
4. Hierarchical clustering of the set of values of the new multiplied attribute represented by the rows of the latter contingency table. This last clustering produces the final reduced predictive attribute, whose values are tree-structured.
5. The final predictive binary attributes are chosen according to the tree-structure on the values of the reduced attributes.

Notice that only the fourth and fifth step above have to be considered in the case where the number $L(v)$ of values of the predictive categorical attribute **v**, is not too large. This is because the contingency table crossing **v** with the attribute to be predicted becomes statistically consistent. Moreover, the size of the row set of the contingency table becomes tractable by a clustering algorithm.

2.2. The Binarization Method

The binarization procedure will be exemplified (see Section 3.2) in the context of our application. Formal and general description of this method will be given here. Different cases and sub-cases have to be distinguished according to the mathematical structure of the set of values $E(v)$ of the categorical predictive attribute **v**. The first case to be considered is that where $L - L(v) - card[E(v)]$ is not too large,

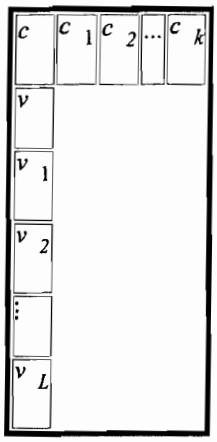

Fig. 3. Contingency table crossing v with c.

in order to make it statistically consistent the contingency table crossing **v** with the attribute to predict, **c**. In the case where **L** is too large, two sub-cases will be distinguished. For the former sub-case, the attribute **v** is multidimensional by nature and can be defined by a sequence $(v^1, v^2, \ldots, v^3, \ldots, v^q)$ of **q** attributes. For example, in our application (see figure 2), an attribute taking into account 4 consecutive amino acids is defined as a sequence of 4 simpler attributes, (v^1, v^2, v^3, v^4); this attribute takes $20^4 = 160,000$ values, because there are 20 different amino acids. For this former sub-case the set of values $E(v)$ can be decomposed as a cartesian product of sets. For the latter sub-case $E(v)$ has not any structure.

Let us denote by $\{c_1, c_2, \ldots, c_k, \ldots, c_K\}$ the value set of the categorical (qualitative nominal) attribute **c** to be predicted. c_k codes one class or concept to recognise from the description given by the predictive attributes, $1 \leqslant n \leqslant K$. In our application the set of values of **c** is $\{X, H, E\}$ and $\mathbf{K} = 3$. If **v** is one of the categorical predictive variables, let us denote by $\{v_1, v_2, \ldots, v_L\}$ its set of values, indicated above by $E(v)$. The statistical basic information used for clustering $E(v)$ is the contingency table crossing **v** and **c** (see Fig. 3)

2.2.1. The Case Where **L** *Is Not Too Large*

In this case—where the above contingency table is statistically consistent—the binarization procedure of ARCADE begins by grouping the set of the values of the predictive attribute **v** in order to reduce the number of binary splits to be considered. These values are grouped into clusters of similar values, so that each cluster will constitute a new value of a new nominal attribute $\bar{v}$. More precisely, the similarity between the values is relative to the discrimination of the concept attribute, **c**. That is the reason for applying a clustering method to the rows of the contingency table above. For simplicity, we will denote the set of values of **v**, $\{v_1, v_2, \ldots, v_L\}$, by $\{1, 2, \ldots, L\}$.

Definition 1. Consider the set G^L of all partitions of $\{1, 2, \ldots, L\}$, into more than one cluster and for which, at least, one cluster includes more than one element. We define a *grouped attribute* $\bar{v}_g$, associated with the partition **g** (belonging to G^L), as the qualitative attribute whose values are defined by the **g**-clusters.

More precisely, let us denote by $j_k = \{i_1, i_2, \ldots, i_u\}$ a given cluster of the partition **g**. j_k is the value of $\bar{v}_g$ on a given object, if and only if i_1 or $i_2, \ldots,$ or i_g, is the value of **v** on this object.

Obviously, this new attribute $\bar{v}$ has fewer values than the initial attribute **v**. Indeed, we can choose, in a proper way, the number of values to use in $\bar{v}$ according to the complexity that we can accept to binarize it. To these new synthetic values, we shall call macro-values.

There are essentially two types of automatic clustering; hierarchical and non-hierarchical. In our case we have employed the hierarchical clustering method AVL ("Analyse de la Vraisemblance des Liens"). This method has been introduced by Lerman [15,17–21,25,31] and developed by him and his collaborators. It is implemented in the computer program CHAVL [24]. We have used a specific form for the aggregation criterion, called $AVL_{0.5}$ [2]. The relevancy of this methodology is motivated by the following reasons:

(i) It is underlined by a very general constructive method of probabilistic similarity indices between combinatorial or logical structures, on a given object set **O**. These structures are interpreted in terms of relations on **O**, eventually weighted; and then, the numerical case is comprised. Description of concepts observed on **O** is also taken into account in this approach, which is extremely general with respect to the data structure. On the other hand, it introduces the Likelihood concept into the Resemblance notion.

(ii) This method allows (see [16,23]) the identification of the significant levels and nodes of the hierarchical clustering tree. A significant level corresponds to a stable partition; and a significant node to the completion of a cluster.

(iii) The hierarchical tree (on the set of values of a given attribute) determines an ultrametric distance between the macro-values.

The two latter points are very important to define, in a relevant and reduced manner, the binary attributes that are going to be used in the decision tree.

Thus, the ARCADE method performs a $AVL_{0.5}$ hierarchical clustering on the set $E(v)$ of the **L** values of our predictive attribute **v** and then chooses a significant partition with **J** clusters. As an example, suppose that our attribute **v** has 400 values (this happens in our application), and assume $\mathbf{J} = 12$. This allows the definition of a new attribute $\bar{v}$ with **J** macro-values. Thus, the first stage of ARCADE reduces significantly the complexity of the problem by replacing a predictive attribute with **L** values by another one with **J** values (**J** much less than **L**). Each of the **J** values of this new attribute is a cluster of the initial set $E(v)$ of **L** values.

Additional and very important complexity reduction is provided by the ultrametric structure of the restriction of the clustering tree on the **J** macro-values set. Let us consider, for instance, $\mathbf{J} = 12$ and then denote by $E = \{e_1, e_2, \ldots, e_{12}\}$ the set of the **J** macro-values of $\bar{v}$.

By itself the categorical attribute $\bar{v}$ provides $2^{11} - 1 = 2047$ binary attributes, corresponding to all the binary partitions of **E**. Nevertheless, most of them are somewhat arbitrary with respect to the prediction purpose. The 12 macro-values of $\bar{v}$ are tree-structured (see Fig. 4), since we have used a hierarchical clustering algorithm. This structure defines an ultrametric distance between these 12 macro-values (the ultrametric distance between any two values is the level at which they are joined, which is 4 in the case of e_5 and e_7, for instance (see Fig. 4). By considering this ultrametric dissimilarity, a partition as $(\{e_3, e_9, e_{12}\}, \{e_1, e_2, e_4, e_5, e_6, e_7, e_8, e_{10}, e_{11}\})$ is, intuitively speaking, much less predictive than a partition as $(\{e_4, e_5, e_7\}, \{e_1, e_2, e_3, e_6, e_8, e_9, e_{10}, e_{11}, e_{12}\})$. To avoid the unpredictive attributes, instead of using all of the $2^{12-1} - 1 = 2047$ binary attributes, we consider only those that correspond to the leaves and nodes of the hierarchy (see Fig. 4 and Fig. 5).

Thus, the binary attributes that ARCADE chooses for the above case, are $a_1, a_2, \ldots, a_{21}$ (see Fig 5).

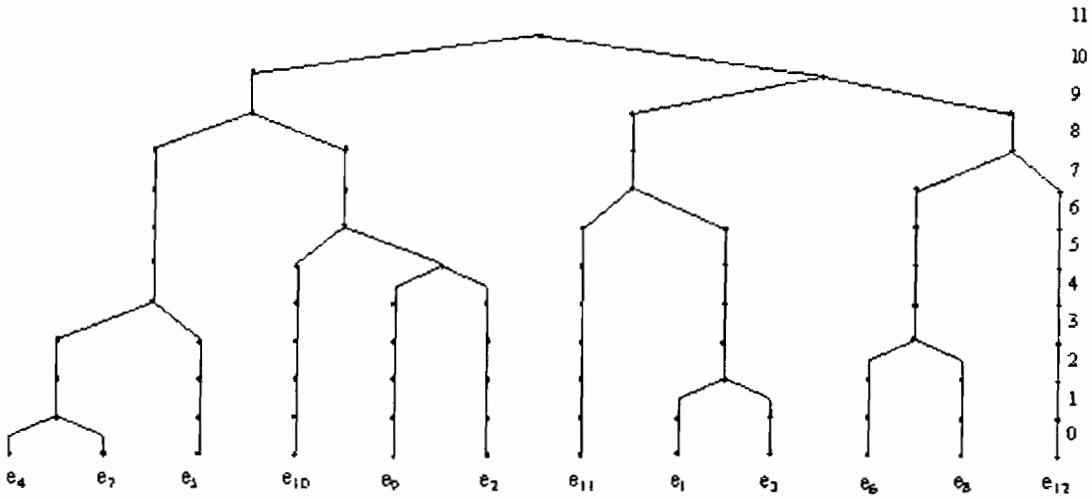

Fig. 4. Example of a clustering tree.

a_1:	$(a_1 = 1 \text{ in } \{e_1\})$ and $(a_1 = 0 \text{ in } \{e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}, e_{12}\})$	(leaf)
a_2:	$(a_2 = 1 \text{ in } \{e_2\})$ and $(a_2 = 0 \text{ in } \{e_1, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}, e_{12}\})$	(leaf)
⋮		⋮
a_{12}:	$(a_{12} = 1 \text{ in } \{e_{12}\})$ and $(a_{12} = 0 \text{ in } \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\})$	(leaf)
a_{13}:	$(a_{13} = 1 \text{ in } \{e_4, e_7\})$ and $(a_{13} = 0 \text{ in } \{e_1, e_2, e_3, e_5, e_6, e_8, e_9, e_{10}, e_{11}, e_{12}\})$	(node 1)
a_{14}:	$(a_{14} = 1 \text{ in } \{e_1, e_3\})$ and $(a_{14} = 0 \text{ in } \{e_2, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}, e_{12}\})$	(node 2)
⋮		⋮
a_{20}:	$(a_{20} = 1 \text{ in } \{e_6, e_8, e_{12}\})$ and $(a_{20} = 0 \text{ in } \{e_1, e_2, e_3, e_4, e_5, e_7, e_9, e_{10}, e_{11}\})$	(node 8)
a_{21}:	$(a_{21} = 1 \text{ in } \{e_4, e_7, e_5, e_{10}, e_9, e_2\})$ and $(a_{21} = 0 \text{ in } \{e_{11}, e_1, e_3, e_6, e_8, e_{12}\})$	(node 9)

Fig. 5. Binary attributes used by ARCADE.

This last part of our method is similar to the Bit-Per-Category Encoding scheme (see [1] for a description of this and other methods for dealing with tree-structured attributes). In our method, however, the main contribution is in the construction of a tree-structured attribute, from a nominal one. In the work of Almuallim et al. [1], and other methods, the initial predictive attributes are already tree-structured.

2.2.2. The Case Where **L** *Is Too Large*

Now we consider the case where the number of values of the predictive attribute **v** is too large for the above contingency table (see Figure 3) to be consistent. In other words, the size of the learning set O_s, is not large enough to assure the statistical consistency needed for the calculations. This is because the contingency table is too hollow. As expressed above, we are going to distinguish two sub-cases according to the structure of **v**; multidimensional or not.

The attribute v is multidimensional:

We assume here that the predictive attribute **v** is defined by a sequence of **q** attribute $(v^1, v^2, \ldots, v^p, \ldots, v^q)$. For instance, our attributes with 160,000 values described in the beginning of 2.2 are a sequence of 4 simpler attributes, each one with 20 values. Let us designate by $A_p = E(v^p)$ the set of values of the pth attribute v^p, $1 < p < q$. In these conditions, the set of values of **v** is given by the Cartesian product $= A_1 \times A_2 \times \cdots \times A_p \times \cdots \times A_q$.

In our application, the attributes are defined by **q**-letter words taking position in a window of a given size ($> q$); all the sets $A_1, A_2, \ldots, A_{q-1}$ and A_q are identical to a set **A** of 20 letters, representing the 20 amino acids, and then $E(v) = A^q$.

Consider the set $Q = \{1, 2, \ldots, p, \ldots, q\}$ of the above subscripts and let $p(Q) = \{Q_1, Q_2, \ldots, Q_r\}$ be a partition of **Q** into **r** non-empty clusters. One may denote Q_i as following:

$$Q_i = \{p_{ic_1}, p_{i_2}, \ldots, p_{ic_j}\}, \quad 1 \leqslant j \leqslant r, \text{ where } c_1 + c_2 + \cdots + c_j = q \text{ and } p_{ij} \in \{1, 2, \ldots, q\}.$$

We define the $\pi(Q)$ -FACTORIZATION of order **r** of **v** as the grouping of the components of **v** into **r** sub-sequences according to the partition $\pi(Q)$. More precisely the attribute **v** is viewed as the following sequence of **r** attributes

$$(u^1, u^2, \ldots, u^i, \ldots, u^r) \quad \text{where } u^1 = (v^{p_{i1}}, v^{p_{i2}}, \ldots, v^{p_r}), \ 1 < i < r.$$

Now the set of values of u^2 is defined by $A_{p,1} \times A_{p,2} \times \cdots \times A_{p,L}$, $1 \leqslant i \leqslant r$.

The cardinality of the latter is given by $c(p_{11}, \ldots, p_{1q}) = \prod_{1 \leqslant j \leqslant q} card(A_{p_{ij}})$ where $card(A_{p_{ij}})$ is the cardinality of $A_{p_{ij}}$.

The above factorization is considered well balanced if the integer numbers $c(p_{i1}, \ldots, p_i)$ are close. A measure of the balance of the factorization can be

$$\sum_{\{(i,i'),1 \leqslant i' \leqslant r\}} |c(p_{i1} + \cdots + p_{ic}) - c(p_{i'1} + \cdots + p_{i'c'_j})|.$$

In the context of our application, reconsider the example of the attribute (with 4 letters) occupying four consecutive positions, which we denote by 1, 2, 3 and 4. The initial attribute **v** is defined by (v^1, v^2, v^3, v^4). **Q** is equal to $\{1, 2, 3, 4\}$ and the partition $\pi(Q)$ is into two clusters comprising two consecutive elements each:

$$\pi(Q) = \{\{1, 2\}, \{3, 4\}\}.$$

Hence, $v = (u^1, u^2)$ where $u^1 = (v^1, v^2)$ and $u^2 = (v^3, v^1)$. In these conditions $c(1,2) - c(3,4) - 20 \times 20 - 200$, and so, the above measure of balance is zero.

The ordered partition of the sequence $(v^1, v^2, \ldots, v^3, \ldots, v^q)$ into the sequence $(u^1, u^2, \ldots, u^r)$ of sub-sequences (see above) can be guided by formal considerations. Thus, as in our application, u^i can be associated with a sequence of consecutive attributes vp. For this case ui is defined as follows:

$$u^1 = (v^{c_{i-1}+1}, v^{c_{i-1}+2}, \ldots, v^{c_{i-1}+n}).$$

Let us suppose that there are no formal constraints for determining the sub-sequences ui from the sequence $\mathbf{v}$ of the $\mathbf{q}$ attributes vp. Then, the most interesting consists in determining a clustering of $(v^1, v^2, \ldots, v^\beta, \ldots, v^q)$ according to the observed similarities between the predictive descriptive attributes $v^\beta \leqslant P \leqslant q$. The hierarchical clustering methodology AVL enables to obtain such a clustering.

We associate now with each attribute u^2, $1 \leqslant i \leqslant r$, the contingency table crossing its set of values with the set of values $\{c_1, c_2, \ldots, c_R, \ldots, c_K\}$, of the attribute $\mathbf{c}$ to be predicted. To this contingency table, which has $c(p_{i1}, \ldots, p_{iL})$ rows and K columns, we apply the hierarchical clustering method $AVL0.5$ on the row set. The purpose consists in obtaining from each clustering tree, resulting from each contingency table, a significant partition into k_i clusters of the values of u_i (the values k_i are usually close). A cluster of this partition is a subset of $A_{p.1} \times A_{p.2} \times \cdots \times A_{p.r}$ (see above). Each such subset determines one single value of the reduced grouped attribute $\bar{u}^2$ deduced from u_i (see Definition 1).

Let us designate by

$$\pi(u^i) = (D_1^i, D_2^i, \ldots, D_r^i, \ldots, D_h^i)$$

the partition of the set of values of u^2, $1 \leqslant i \leqslant r$. One may denote by $\{b_1^i, b_2^i, \ldots, b_j^i, \ldots, b_k^i\}$ the set of values of $\bar{u}^i$, where b_j^i represents the set D_j^i, $1 \leqslant j \leqslant k_i$. For simplicity, in general we consider $b_j^i = j$ and so, the set of values of $\bar{u}^i$ is $(1, 2, \ldots, k_i)$.

Now, we consider a MULTIPLICATION in a given order of the set $\{\bar{u}^1, \bar{u}^2, \ldots, \bar{u}^2, \ldots, \bar{u}^i\}$ of the reduced attributes. This operation can be considered the inverse of the FACTORIZATION operation. More explicitly, we consider here the attribute, that we denote, $\bar{u}$, and defined by the ordered sequence $\{u^1, u^2, \ldots, u^2, \ldots, u^i\}$. A given value of π is defined by a sequence of values b_j^i having the following form:

$$\{b_1^i, b_2^i, \ldots, b_j^i, \ldots, b_k^i\}, \quad 1 \leqslant j_i \leqslant k_i, \ 1 \leqslant i \leqslant r.$$

This value represents the Cartesian product $D_{j_1}^1 \times D_{j_2}^2 \times \cdots \times D_{j_i}^i \times \cdots \times D_{j_r}^r$.

Hence the reduction obtained at this level for the initial attribute $v - (v^1, v^2, \ldots, v^q)$, with $card(E(v))$ values, is the attribute $\bar{u}$, which has $k_1 \times k_2 \times \cdots \times k_s$ values. In general, the set of values of $\bar{u}$ is $E(\bar{u}) = (1, 2, \ldots, k_1 \times k_2 \times \cdots \times k_i)$.

For the following reduction stage, the contingency table crossing the set of values of the above multiplied attribute $\bar{u}$ with the set of values of the attribute $\mathbf{c}$ to predict, is considered. This table has $k_1 \times k_2 \times \cdots \times k_i \times \cdots \times k_i$ rows and $\mathbf{K}$ columns.

Finally, the hierarchical clustering method $AVL_{0.5}$ is performed on the row set of the last contingency table. By retaining a significant partition [23], we define a new synthesized categorical attribute $\bar{v}$ whose values are associated with the clusters of this partition. Note that the set of values of $\bar{v}$ has been illustrated above by a set of 12 macro-values, designated by $(e_1, e_2, \ldots, e_{12})$.

As described above, the binary attributes that we keep correspond to the leaves or nodes of the restriction of the clustering tree on the clusters of the retained previous partition (see figures 4 and 5).

Thus, if **J** is the number of clusters of the last partition, there will be at most $(2J-1)$ binary attributes, associated with the initial attribute **v**, which will intervene in the tree decision building.

The attribute v is not multidimensional:

We assume here that the set of values $E(v)$ of **v** has not any specific structure. We code by the first **L** positive integers, $\{1, 2, \ldots, l, \ldots, L\}$, this set of values. We are now going to describe how to apply the FACTORIZATION procedure to $E(v)$. Let us start by a very small and simple example where $\mathbf{L} = 4$ and $\mathbf{r} = 2$ (factorization of order 2). The sequence of values $(1, 2, 3, 4)$ is coded by a sequence of bidimensional vectors (i, j), lexicographically ordered, where each component is included in the set $\{1, 2\}$. The first and second components are respectively the values of two created attributes, v^1 and v^2. More explicitly, the correspondence between the values is:

v	v^1	v^2
1	1	1
2	1	2
3	2	1
4	2	2

A factorization of order 2 is generally sufficient for the current applications. We only consider here this case. A generalisation for any **r** is easy to conceive. Two cases have to be distinguished. For the former case, **L** is a perfect square and can be written as L^2, where L' is a positive integer. Then, the above correspondence table between the values of **v** and those of the factors v^1 and v^2, becomes:

v	v^1	v^2
1	1	1
2	1	2
⋮	⋮	⋮
L'	1	L'
$L'+1$	2	1
⋮	⋮	⋮
$2L'$	2	L'
⋮	⋮	⋮
L	L'	L'

By this technique the set of values $E(v)$ is decomposed into a Cartesian product $E(v^1) \times E(v^2)$. $E(v^1)$ and $E(v^2)$ are respectively the sets of values of the created attributes, v^1 and v^2. These factorise the initial attribute **v**. Nevertheless, this decomposition is arbitrary and depends on the coding of the set of values

of **v**. Since it is intuitively desirable to make v^1 and v^2 as independent as possible, we suggest to assign randomly the codes $\{1, 2, \ldots, L\}$ to the values of **v**.

Let us now consider briefly the case where the number **L** of the values of the attribute **v**, is not a perfect square. The factorisation procedure of order 2 can still be applied. For this purpose define $L' = [\sqrt{L}] + 1$, where $[x]$ designates the integer part of the positive number **x**. We have $L'^2 > L$. Consider now the following set of bidimensional vectors

$$\{(i, j)/1 \leqslant i \leqslant L',\ 1 \leqslant j \leqslant L'\}$$

that we suppose totally lexicographically ordered as in the above table. And now make one increasing correspondence between the sequence $(1, 2, \ldots, l, \ldots, L)$ and that defined by the ordered pairs. Since $L'^2 > L$, the latest part of the sequence of ordered pairs do not correspond to any of the original values of **v**. This should not bring any serious problem in the treatment of the factors v^1 and v^2, where the values of v^1 and v^2 are respectively defined by the first and the second components of the ordered pairs.

Now, according to the scheme described above, we associate with v^1 and v^2 a pair of contingency tables crossing respectively v^1 and v^2 with the attribute to be predicted, **c**. The common size of these tables is $L' \times K$ (L' rows and **K** columns). $AVL_{0.5}$ is then applied to the row set of each of the contingency tables. From the first clustering tree (resp. the second) we retain a significant partition with k_1 (resp. k_2) clusters (k_1 and k_2 are usually close). The values of k_1 and k_2, which are empirically chosen, are such that the contingency table with $k_1 \times k_2$ rows and **K** columns is accurate enough and statistically consistent. The retained partitions induce the reduction of v^1 and v^2 into synthesised attributes $\bar{v}^1$ and $\bar{v}^2$. By denoting $\{D_1^1, \ldots, D_j^i, \ldots, D_{k_1}^1\}$ and $\{D_1^2, \ldots, D_{j'}^2, \ldots, D_{k_2}^2\}$ these partitions, B_j^1 (resp. $B_{j'}^2$) corresponds to one single value of $\bar{v}^1$ (resp. $\bar{v}^2$), $1 \leqslant j' \leqslant k_1$ (resp. $1 \leqslant j' \leqslant k_2$).

As before, the next step consists in defining a new attribute $\bar{u}$ which results from the multiplication of $\bar{v}^1$ and $\bar{v}^2$: $\bar{u} = (\bar{v}, \bar{v}^2)$. A given value of $\bar{u}$ can be written as the cartesian product $D_j^1 \times D_{j'}^2$, $(1 \leqslant j \leqslant k_1,\ 1 \leqslant j' \leqslant k_2)$. Note here that some few values of (v^1, v^2) falling in some of the sets $D_j^1 \times D_{j'}^2$, $(1 \leqslant j \leqslant k_1,\ 1 \leqslant j' \leqslant k_2)$ might not correspond to any original value of the attribute **v**. This occurs when $L' > L$, for coding reasons. These non-representative values of (v^1, v^2) are simply ignored.

The following steps of the reduction process are continued as previously:

(i) setting up the contingency table ($k_1 \times k_2$ rows and **K** columns) which crosses the attributes $\bar{u} = (\bar{v}, \bar{v}^2)$ and c;
(ii) performing hierarchical clustering $AVL_{0.5}$ on the row set of the latter contingency table;
(iii) gathering a significant partition whose number of clusters is of a given order; considering the restriction of the clustering tree on the set of the clusters of this partition;
(iv) associating with each leave or node of the preceding truncated clustering tree a binary attribute, which will intervene in the binary decision tree construction.

3. Binary Attributes Used in ARCADE for the Prediction of Protein Secondary Structure

3.1. The Initial Descriptive Attributes

As said in the introduction, the description of the position to be predicted, must take into account all its neighborhood in the protein sequence. For this purpose, we begin by considering a window having an odd size $2p + 1$ (**p**: integer), centred on the position to be predicted. Different values of **p** have been

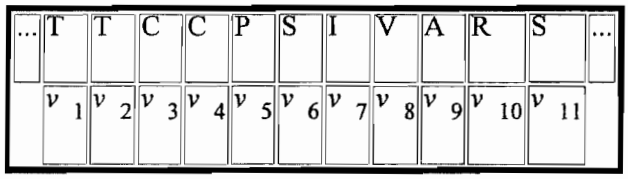

Fig. 6. The initial predictive attributes $v_1, v_2, \ldots, v_{11}$.

tried and the best prediction results have been obtained for the experimental value $\mathbf{p} = 5$. If we designate by **y** the position of the protein whose secondary structure we want to predict then, for a window of size eleven, the first descriptive attributes used by us were $v^1, v^2, \ldots, v^{11}$, see Figure 6.

Each one of these 11 nominal attributes takes 20 different values (because there are 20 different amino acids). We can consider therefore the application of a program like CART [3] or C4.5 [34] in order to predict the secondary structure of a protein. The data set contains 30983 data points and $11 + 1$ columns. The last colums is the concept to be predicted. Because there are 3 classes to be predicted (**alpha helix, beta sheet** and **turn**), the CART method is quite slow on this data, and therefore we have not used cross-validation. The results obtained with CART were 54.3% of correct predictions (using 63% of the data to train and 37% to test). In our data, the percentages of the three classes **X**, **H** and **E** are respectively, 46, 6%, 29% and 24, 4%. The results obtained with C4.5 were 52.5% of correct classifications (using 10-fold-cross-validation).

The CART methodology consists in grouping the values of the categorical attributes into two groups, defining thus a binary attribute. C4.5 has also an option, $-s$, which groups the values of the categorical attributes into **r** groups ($r > 2$). In [34, p.65] we have 151 proteins of variable size which totals 30983 observations (positions, amino acids) to be predicted. As said in the introduction, to binarize this predictive attribute with 160,000 values, we start by considering the contingency table crossing this attribute with the concept attribute (to be predicted). Thus, each of the 30983 observations adds one unity in the concerned entry of this contingency table, which size is $20^4 \times 3 = 480{,}000$ cases.

> "...The method used is based on iterative merging of value groups. The initial value groups are just the individual values of the attribute under consideration and, at each cycle, C4.5 evaluates the consequence of merging every pair of groups. (Any division of the attribute values into attribute groups is reflected in the split of the training cases by that attribute, and so in the corresponding split information and gain). The process continues until just two values groups remain, or until no such merger would produce a better partition of the training cases..."

This grouping option of C4.5 seems therefore to have a combinatorial complexity. The results obtained with C4.5 and the -*s* option were 47.3% of correct classifications (using 10-fold-cross-validation).

The 11 nominal attributes considered above are natural to consider. Nevertheless, since the order between the different amino acids is most important for prediction, we have decided to generalise and to make more accurate the above description.

For this purpose we consider a "Multiplication" operation of these elementary attributes, in a consecutive manner. More precisely, by associating two by two, in an adjacent way, the above eleven attributes, we define 10 composite new attributes. Each one of these new attributes has $20 \times 20 = 400$ values. Namely, these attributes are $(v^1, v^2), (v^2, v^3), (v^3, v^4), (v^4, v^5), (v^5, v^6), (v^6, v^7), (v^7, v^8), (v^8, v^9), (v^9, v^{10})$, and (v^{10}, v^{11}).

Each one of these new attributes can be expressed as a word with two letters. Its value set is defined by the cartesian square $A \times A$, where $\mathbf{A} = \{A, C, \ldots, Y\}$ and represents the 20 amino acids. Thus, the 400 values of these new attributes are $\Lambda\Lambda, \Lambda C, \ldots, YY$.

Again, although a priori CART and C4.5 might be inappropriate for these complex attributes, we have considered their application. The present version of CART is prohibitively complex for these attributes. In fact, as we have 3 classes to be predicted, at each node and for each categorical attribute with 400 values, CART will test $2^{400-1} - 1$ binary attributes, which can not be considered. Nevertheless, the CART program can be improved and choose the same binary attribute by testing only 3×400 attributes. The results obtained with C4.5 (without the options *-s*, which is prohibitive in this case) were 52.6% of correct classifications (10-fold-cross-validation).

In order to get a better description of our data, we have generalised the above descriptive attributes. Thus, one may consider attribute words with a number of letters varying between one and eleven. However note that the possible number of values of a word with 1 letters, is 20^l, and this number becomes extremely large if 1 is not small enough. Thus, for $\mathbf{l = 4}$, $20^l = 160{,}000$. This particular value of 1 was the one used by us:

Definition 2. A descriptive predictive attribute is a word of four consecutive letters having a given position in the sliding window of size eleven.

Therefore, there are $11 - 4 + 1 = 8$ attributes, and for each instantiation of the sliding window, we get one observation of each of the eight attributes. On the other hand, we associate with these attributes the letter of the secondary structure, at the center of the window.

With these attributes of 160,000 values each, neither CART nor C4.5 work, and, to our knowledge, there is no method capable of dealing with such complex attributes. This was the motivation for this work and the development of a method, ARCADE, capable of binarizing these complex attributes.

Now, consider one specific attribute **v** with 160,000 values. You may suppose that it concerns the word occupying the positions 5, 6, 7 and 8, of the window in figure 6. In our learning set [5] we have 151 proteins of variable size which totals 30983 observations (positions, amino acids) to be predicted. As said in the introduction, to binarize this predictive attribute with 160,000 values, we start by considering the contingency table crossing this attribute with the concept attribute (to be predicted). Thus, each of the 30983 observations adds one unity in the concerned entry of this contingency table, which size is $20^4 \times 3 = 480{,}000$ cases.

As it was mentioned in the Introduction, the general idea consists in clustering the rows of the contingency table in order to reduce the set of values of the descriptive attribute. Nevertheless, with these complex attributes, it is irrelevant to proceed directly. In fact, this table is too sparse to allow a significant clustering of its rows, which are mostly empty, and then, the whole 160,000 possible values of the specified attribute cannot be represented. Otherwise, due to computational complexity reasons, there is no clear solution for handling such a large data set, with known clustering algorithms.

3.2. Building the Binary Attributes Used in ARCADE

In the introduction, the solution that we have adopted was briefly described. It consists in starting by "Factorizing" the attribute **v**, which has 4 letters, into an ordered pair of 2-letter words attributes (u^1, u^2) where u^1 and u^2 are consecutive factors of the factorization of **v**. A clustering reduction procedure on the

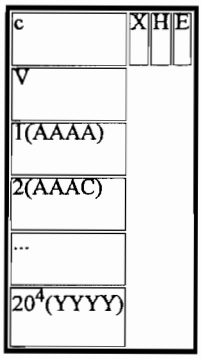

Fig. 7. Initial contingency table.

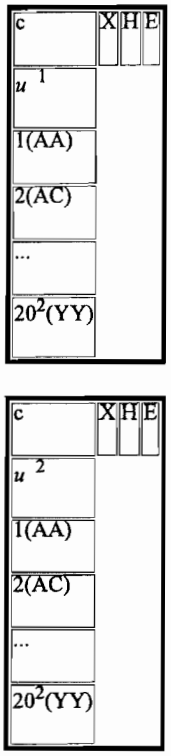

Fig. 8. Contingency tables for the factorised attributes.

value set of u^1 (resp. u^2) leads to the statistical reduction of the set of values of **v**. The method will be described in detail.

Let us consider a 2-letter word attribute, for instance u^1. In this case, the size of the contingency table crossing this attribute with the attribute to be predicted, **c**, is $20^2 \times 3 = 1200$. This means 30983 observations are distributed over 1200 entries of this table. This gives an average of $30983/1200 = 25.8$ observations for each entry, which shall make the contingency table statistically consistent. Consequently, in order to aggregate the set of values of the above attribute **v**, we begin by respectively aggregating the sets of values of u^1 and u^2. For this purpose we substitute the initial contingency table crossing **v** and **c**, by an ordered pair of contingency tables crossing u^1 and **c**, respectively, u^2 and **c**.

We have now to employ a clustering method on each row set of both contingency tables in figure 8, independently. Let us suppose for instance that the sets of values of u^1 and u^2 were clustered into, respectively, **l** and **m** clusters. Let us designate by $(V_1^1, \ldots, V_2^1, \ldots, V_3^1)$ (resp. $(V_1^2, \ldots, V_3^2, \ldots, V_m^2)$) the set of values resulting from the first (resp. the second) clustering. V_2^1, for instance, represents a cluster of values of u^1 and is therefore a value of a new attribute, $\bar{u}^1$, which is a reduced version of u^1. The set of values of the "joint attribute" $(\bar{u}, \bar{u}^3)$, which is given by the cross product $(V_1^1, \ldots, V_2^1, \ldots, V_l^1) \times (V_1^2, \ldots, V_3^2, \ldots, V_m^2)$, has cardinal $l \times m$. This new attribute $\bar{v} = (\bar{u}^1, \bar{u}^2)$ is a reduced version of **v**. More precisely, for a given (i, j), $1 \leqslant i \leqslant l$, $1 \leqslant j \leqslant m$, $V_i^1 \times V_j^2$ is considered as a single value of a reduced version $\bar{v}$ of **v**. Thus, all the words with 4 letters belonging to $V_i^1 \times V_j^2$ are considered as equivalent.

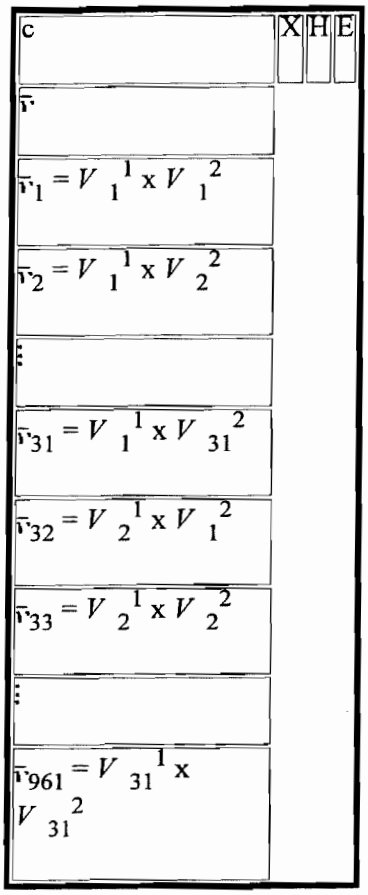

Fig. 9. Contingency table for the reduced attribute.

For comparable accuracy in both clusterings, **l** and **m** must be close. On the other hand, a new reduction procedure of the set of values of $\bar{v}$ is still necessary on the basis of the associated $(\bar{v}, c)$ contingency table. Therefore, for statistical and computing reasons, the size of $l \times M$ cannot be too large. By choosing this size about one thousand, the ratio between the number of observations and the number of the entries of the contingency table is around 3, for the above illustrated case of 30,983 observations. For instance if we consider $\mathbf{l} = \mathbf{m} = 31$; then, $l \times M = 961$.

In our approach the partition $\{V_1^1, V_2^1, \ldots, V_{31}^1\}$ (resp. $\{V_1^2, V_3^2, \ldots, V_{31}^2\}$) is gathered at a given level of the clustering tree. This clustering tree is built by the hierarchical clustering method $AVL_{0.5}$ applied to the rows of the contingency table (u^1, c) [resp. (u^2, c)]. Consider $\{V_1^1, V_2^1, \ldots, V_{31}^1\}$ (resp. $\{V_1^2, V_3^2, \ldots, V_{31}^2\}$) as the set of values of the reduced attribute $\bar{u}^1$ (resp. $\bar{u}^2$). We may define the new attribute $\bar{v}$ as the "Multiplication" of $\bar{u}^1$ and $\bar{u}^2$: $\bar{v} = \bar{u}^1 \times \bar{u}^2$.

This new categorical attribute $\bar{v}$ has 961 values that we can denote by $v_1, v_2, \ldots, v_{961}$. Clearly, the obtained reduction is not sufficient. Indeed, this attribute gives rise to $2^{960} - 1$ binary attributes. Therefore, a new application of $AVL_{0.5}$ is considered on the set of the rows of the contingency table $(\bar{v}, c)$, crossing this last attribute $\bar{v}$ with the attribute to be predicted, **c** (see Figure 9).

The hierarchical clustering methodology AVL enables to find "significant" levels of the hierarchical tree, that is, levels, which correspond to a stable partition. Usually, there are many such levels, and therefore, one can choose a significant partition whose number of clusters is close to what we want (for instance, 31, as above). The last application of $AVL_{0.5}$ enables also to find a significant partition of $\{v_1, v_2, \ldots, v_{961}\}$, for instance into 12 clusters. This partition determines the final reduction of the set of values of the initial attribute **v** (which had 160,000 values). The associated qualitative (categorical) attribute that we may denote by $\bar{v}$, has 12 values. Each one of them represents a cluster of the final application of the clustering method AVL.

Let us denote by $E = (e_1, e_2, \ldots, e_{12})$ the set of these 12 macro-values. Each one of these macro-values is a cluster of the set of 160,000 values of the initial attribute **v**. We can now consider the binarization of these categorical attributes; to each such attribute corresponds $2^{12} - 1 = 2047$ binary attributes. Nevertheless, we can still reduce this number to a much smaller one, by considering the clustering tree obtained by $AVL_{0.5}$ on the set of values E. In fact, the values of $\bar{v}$ are tree-structured (see Figure 4); we profit from this structure in order to reduce once more the final complexity. This structure defines an ultrametric distance between the 12 macro-values (the ultrametric distance between any two values is the level at which they are joined, which is 4 in the case of e_5 and e_7, for instance (see Figure 4)). By considering this ultrametric dissimilarity, a partition as $(\{e_3, e_9, e_{12}\}, \{e_1, e_2, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\})$ is, intuitively speaking, much less predictive than a partition as $(\{e_1, e_2, e_7\}, \{e_1, e_2, e_3, e_6, e_8, e_9, e_{10}, e_{11}, e_{12}\})$.

To avoid the unpredictive attributes, instead of using all of the $2^{12-1} - 1 = 2047$ binary attributes, we consider only those that correspond to the leaves and nodes of the hierarchy (see Figure 4 and Figure 5). The total reduction in complexity is therefore enormous:

$$(2^{20-1} - 1) \rightarrow (2^{961-1} - 1) \rightarrow (2^{12-1} - 1) \rightarrow \leqslant 22$$

3.3. *The Obtained Results for the Prediction of Protein Secondary Structure*

Our database is constituted by a set $S = \{s_i, 1 \leqslant i \leqslant 151\}$ of 151 non-homologous globular proteins [5], which result in 30983 amino acids. Our data set has therefore 30983 rows (objects), which are grouped in 151 subsets. Each protein has therefore an average of 205 residues. We have used the jacknife procedure to assess the accuracy of our method. Each one of the subsets corresponding to each of the 151 proteins $s_i, 1 \leqslant i \leqslant 151$, was successively chosen to be the test set. The training set for this protein was constituted by the remaining objects (amino acids), corresponding to the other 150 proteins $(O_2 = S - \{e_2\})$.

This set O_i was then separated into 10 subsets, $O_i^1, O_i^2, \ldots, O_i^{10}$, having each 15 proteins. Then, for $j = 1, 2, \ldots, 10$, the complementary set of O_i^j in O_i, $O_{i,j} (= O_i - O_i^j)$ is used to construct a decision tree $T_{\max i,j}$, where each leaf is pure (all of the observations belong to the same class) or contains no more than 5 observations. We have used various coefficients to choose the binary attribute that splits a node of the decision tree [7,22]. The results that we present here are for the Gini coefficient. The predictive attributes used have been defined above and are deduced from a word with 4 letters in a window of size 11. By associating with each one of these attributes 22 binary attributes, we have in all $8 \times 22 = 176$ binary attributes. The pruning strategy was the same as in CART. From the sequence of sub-trees obtained by pruning, the one that maximises the prediction accuracy for the set O_i^j,which has not yet been used, has been chosen. The corresponding pruning parameter is $\alpha(i, j)$.

For each $i = 1, 2, \ldots, 151$, we calculate the average $\alpha(i) = (\alpha(i, 1) \mid \alpha(i, 2) \mid \ldots \mid \alpha(i, 10))/10$. Afterwards, a decision tree $T_{\max(i)}$ is constructed on the set of 150 proteins O_i. After pruning, the tree corresponding to $\alpha(i)$ is selected. To estimate the accuracy of this tree, we use the sequence s_i, which has not yet been used. A final operation [7,22], which consists of a prediction correction index and is specific to this application, is used to improve the quality of the final prediction.

In the end, we have 151 percentages. The global quality of our method ARCADE is estimated as the average of these 151 values. This average can be an ordinary arithmetic mean (Q_{chain}), or a weighted mean, where the weights correspond to the respective lengths of the proteins (Q_{total}). The percentages of each of the three classes (**H**, **E**, and **X**) were also calculated. For instance, Q_H^{obs} is the percentage of those elements, which have been observed in the **H** class, that are correctly predicted. Q_H^{pred} is the percentage of those elements, which have been predicted in the **H** class, that are correctly predicted.

The final results, obtained with the Gini coefficient, are:

$Q_{chain}(\%)$	$Q_{total}(\%)$	$Q_H^{obs}(\%)$	$Q_H^{pred}(\%)$	$Q_E^{obs}(\%)$	$Q_E^{pred}(\%)$	$Q_X^{obs}(\%)$	$Q_X^{pred}(\%)$
65.6	65.1	64.6	62.4	60	55.8	68.1	72.6

With a random split, we have obtained $Q_{total} = 61.3$, which shows that a good prediction coefficient must be used. We give now the results of other methods, developed in this decade, for the protein secondary structure prediction problem:

Method	Q_{total}
Kneller et al., 1990	63%
Method SM from Zhang et al. 1992	63.5%
"Expert-NM" (Zhang et al. 1992)	63.1%
Method MBR from Zhang et al. 1992	64.5%
Hybrid Method from Zhang et al. 1992	66.4%
Salzberg and Cost, 1992	65.1%
"Reference Network" from Rost and Sander, 1993	61.7%
Yi and Lander, 1993	68%
Levin et al., 1993	68.5%
PHD from Rost and Sander, 1993	70.8%
SSP from Solovyev and Salamov, 1994	65.1%; 68.2%

The results that we have obtained cannot be compared with the ones obtained by other methods that use different data sets. Nevertheless, the performances that we have obtained are amongst the ones obtained by the best methods. On the other hand, as pointed out in [35], for some of these methods the actual performances may be lower if tested on non-homologous data sets or by using the jacknife procedure. The highest performance (70.8%) was obtained for the method PHD [35], which is a combination of several neural networks. This method uses an additional information, which we do not use, given by

the multiple sequence alignments of homologous proteins. If a new protein has no homologues, the performance of this method falls considerably.

4. Conclusion and Perspectives

In this work, we have set up a method for extracting a few numbers of relevant predictive binary attributes, from a categorical nominal attribute having a very large number of values. The aim of this technique is here situated in the context of the construction of a binary decision tree by the CART method. However, our binarization technique is independent of the prediction method used. Thus, any prediction method based on binary attributes can employ it.

In the case where the initial categorical attributes do not have a large number of values, we have shown how to form more informative attributes by "cartesian multiplication". This formation precedes the application of the binarization method, in which tree-structured attributes are built for prediction purposes.

The ARCADE method gives us a strategy, which enables to treat, for the first time, four amino acids taken together in the protein secondary structure prediction problem. By comparison with other methods applied to this recognition problem, our results are very competitive.

We plan to generalise the ARCADE method to the case of non-binary decision trees. This will allow us to introduce a new splitting rule taking into account the ultrametric structures of the clustering trees, which respectively organise the sets of values of the predictive categorical attributes.

4.1. Comparison between ARCADE and CART Methods

We shall now give some elements for the comparison between ARCADE and CART in terms of their respective complexities. In this comparison we will refer to our application (see Section 3). Let us start by considering the most favourable case for CART, in which the number **K** of classes to be predicted is equal to 2 and where the splitting criterion has an inertial nature; for instance the Gini coefficient [7]. Under these conditions, to a categorical predictive attribute **v** with **L** values, CART associates $\mathbf{L-1}$ binary attributes. These are obtained by successive cuttings according to a sorting of the set of values of **v**. This sorting is established at each node of the decision tree, according to the relative frequency of the class 1 (or 2) to be predicted on each value of **v**. These relative frequencies are empirically determined from the subset of the learning set corresponding to that node. The number of nodes of the decision tree is of order $O(n)$ (Landau notation), where **n** is the size of the learning set. On the other hand, the computational complexity for establishing the mentioned sorting of the **L** values, is in $O(L\log_2 L)$.

Therefore, the computational complexity for setting up the binary attributes that we have to evaluate is in $O(nL\log_2 L)$. This evaluation is based upon an association coefficient between categorical attributes as the Gini coefficient [7,22]. Another type of computational complexity concerns the number of binary attributes that we have to check at each node. This number is here $\mathbf{L-1}$. Under these conditions, it is obviously absurd, for statistical and complexity reasons, to apply the CART methodology to the binarization of our initial predictive attributes with 20^4 values.

There are 151 non-homologous proteins which constitute our data: $S = \{s_i, 1 \leqslant i \leqslant 151\}$. To each s_i we associate the complementary set $O_i = S - \{s_i\}$. The idea now is to learn with these sets of 150 proteins and to estimate the true performance of the final decision tree with s_i; this is done 151 times.

The ARCADE binarization method is applied only once, to the whole set of 151 proteins, instead of applying it 151 times to the sets O_i. This is because we have shown [7] that the difference in the final performances—between one application of our binarization method or applying it more rigorously 151 times—is smaller than 0.5%.

Consider a factorization of order **r** equals 2 and let us evaluate, in this case, the computational complexity of the ARCADE binarization method. The automatic clustering method AVL is of complexity $O(l^2\log_2(l))$ for a set of size **l**. This method has to be employed three times before the construction of the decision tree. For instance, in our application, the hierarchical clustering has been applied twice in order to cluster two sets of 400 rows of two contingency tables. Clearly, these two clusterings can be performed in parallel. The third clustering concerns the set of rows of a contingency table comprising 961 rows.

More generally, we have to apply the hierarchical clustering method AVL, in parallel, to two sets of $\sqrt{L}$ rows. The computational complexity for this treatment is $O(L\log_2(\sqrt{L}))$. The third and last application of the clustering algorithm AVL concerns a row set whose size is of the same order as $\sqrt{L}$. This complexity for setting up the binary attributes is lower than the one reached in the CART method, which is, as we have seen, $O(nL\log_2(L))$.

Now we could want to employ our idea of factorization in CART. For this purpose and for $\mathbf{r} = 2$, we could start by factorizing a given predictive attribute **v** with **L** values (e.g., $\sqrt{L} = 40$). Thus, in each node of the decision tree, there are two sets of $\sqrt{L}$ binary attributes to evaluate. The best binary attribute on each set is chosen and then these two binary attributes are multiplied giving rise to 4 binary attributes. The best among these is finally taken for splitting the concerned node. Therefore the complexity on each node, in terms of the number attributes to consider, is $2\sqrt{L} + 4$ (e.g., $2 \times 400 + 4 = 804$). Nevertheless there is a statistical consistency problem when we go down in the decision tree.

More deeply one may try to use in CART, in addition to the factorization technique, the clustering idea that we have also used in ARCADE. In CART however, the justification of the binary attributes determined from a categorical one is given by [11]. By using dynamic programming, the Fisher algorithm provides the best partition of the set of values of the categorical attribute, into a given number of groups and according to an inertial criterion. In CART two groups are requested. By demanding, after factorization, **s** groups for each factor, we may reproduce our reduction process (see Section 2.2.2), but by using Fisher's algorithm instead of the AVL method. Nevertheless, if **J** designates the final number of clusters obtained in the last application of the clustering algorithm, we have to consider $(2^{J-1} - 1)$ binary attributes. The advantage of ARCADE would be that, as the final macro-values are tree-structured, we only have to consider $(2J - 1)$ binary attributes in the decision tree construction.

Moreover, as mentioned above, it is important to notice that the use of the Fisher clustering algorithm is constrained by two conditions. The former consists of the inertial nature of the criterion employed for splitting the nodes of the decision tree. The latter, which is the most serious, fixes $\mathbf{K} = 2$ classes to predict. The solution proposed in CART for $\mathbf{K} > 2$ is of exponential complexity, whereas the complexity of ARCADE is independent of **K**.

Acknowledgements

Joaquim Costa's work was partly supported by program PRAXIS XXI, FEDER and program "Financiamento Plurianual de Unidades de I and D da JNICT"

References

[1] Almuallim, H., Akiba, Y. & Kaneda, S. (1995) On Handling Tree-Structured Attributes in Decision Tree Learning. *International Conference on Machine Learning*, 1995.

[2] Bacelar-Nicolau, H. (1985): The affinity coefficient in Cluster Analysis. *Methods of Operations Research*, vol. 53, 507–512.

[3] Breiman, L., Friedman, J. H., Olshen, A. and Stone, C. J. (1984): *Classification and Regression Trees (1984)*. Wadsworth, Belmont.

[4] Buntine, W. & Niblett, T. (1992): A further comparison of splitting rules for decision-tree induction. *Machine Learning*, 8, 75–86, 1992.

[5] Colloc'h, N., Etchebest, C., Thoreau, E., Henrissat, B. & Mornon J.P.: *Protein Engineering (1993)*, 6, 377–382.

[6] Cost, S. & Salzberg, S. (1993): A Weighted Nearest Neighbor Algorithm for Learning with Symbolic Features. *Machine Learning*, 10, 57–78.

[7] Costa, J.F.P. (1996): Coefficients d'association et binarization par la classification hiérarchique dans les arbres de décision. Application à l'identification de la structure secondaire des protéines. Thèse de doctorat, Université de Rennes I, Rennes, France, 1996.

[8] Dougherty, J., Kohavi, R., Sahami, M. (1995): Supervised and Unsupervised Discretization of Continuous features. In Armand Prieditis & Stuart Russell, eds. *Machine Learning: Proceedings of the Twelfth International Conference*, 1995, Morgan Kaufman Publishers, San Francisco, CA.

[9] Fayyad, U.M., Irani, K.B. (1993): Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. *Proc. of the IJCAI 93*, 13th International Joint C. On A.I., Chambéry, France, 1993.

[10] Fisher, W.D. (1958): On grouping for maximumhomogeneity. *Journal of the American Statistical Association*, 53, pp. 789–798.

[11] Friesner, R.A., Gunn, J.R. (1996): Computational studies of protein folding. *1996 Annual Review of Biophysics and Biomolecular Structure*, 25, pp. 315-342.

[12] Goodman, L.A. & Kruskal, W.H.: Measures of association for cross classifications, *Journal of the American Statistical Association*, 49 1954, pp. 732–764.

[13] Heath, D., Kasif, S. & Salzberg, S. (1993): Induction of Oblique Decision Trees. *Proc. of the IJCAI 93*, 13th International Joint C. On A.I., Chambéry, France, 1993.

[14] Krzanowsky, W.J. (1975): Discrimination and Classification Using Both Binary and Continuous Attributes. *Journal of The American Statistical Association*, Volume 70, Number 352, pp. 7.

[15] Lerman, I.C. (1981): Classification et analyse ordinale des données. Paris, Dunod.

[16] Lerman, I.C. (1983): Sur la signification des classes issues d'une classification automatique. *Numerical Taxonomy*, edited by J. Felsenstein, NATO ASI Series Vol. G1, Springer-Verlag, Berlin, pp. 179–198.

[17] Lerman, I.C. (1987): Construction d'un indice de similarité entre objets décrits par des variables d'un type quelconque. Application au problème du consensus en classification. *Revue de Statistique Appliquée*, xxxv(2), pp. 39–76.

[18] Lerman, I.C. (1991): Foundations of the Likelihood Linkage Analysis (LLA) classification method. *Applied Stochastic and Data Analysis*, Vol. 7, pp. 63–76.

[19] Lerman, I.C. (1992a): Conception et analyse de la forme limite d'une famille de coefficients statistiques d'association entre variables relationnelles I. *Revue Mathématique Informatique et Sciences Humaines*, 30è année, n° 118, Paris, pp. 35–52.

[20] Lerman, I.C. (1992b): Conception et analyse de la forme limite d'une famille de coefficients statistiques d'association entre variables relationnelles II. *Revue Mathématique Informatique et Sciences Humaines*, 30è année, n° 119, Paris, pp. 75–100.

[21] Lerman, I.C. (1993): Likelihood linkage analysis (LLA) classification method (Around an example treated by hand). *Biochimie 75*, Elsevier éditions, 1993, pp. 379–397.

[22] Lerman, I. C. & Pinto da Costa, J. F. (1997): Reducing the number of binary splits, in Decision Tree Induction, by means of an Hierarchical Classification. Technical Report 1138, IRISA (November 1997) and n. 3312, INRIA (December 1997). 38 pages.

[23] Lerman, I.C. & Ghazzali, N. (1991): What do we retain from a classification tree? an experiment in image coding. *Symbolic-Numeric data analysis and learning*, edited by E. Diday and Y. Lechevallier, Nova Science Publishers. Proceedings of the Conference of Versailles, September 18–20, 1991, pp. 27–42, 1991.

[24] Lerman, I.C., Peter, Ph. et Leredde, H. (1993–1994): Principes et calculs de la méthode implantée dans le programme CHAVL (Classification Hiérarchique par Analyse de la Vraisemblance des Liens). *La Revue de Modulad, no* 12, pp.33–70 et *n* 13, pp.63–90.

[25] Lerman, I.C. & Tallur, B. (1980): Classification des éléments constitutifs d'une juxtaposition de tableaux de contingence. *Revue de Statistique Appliquée*, n° 28, Paris, pp. 5–28.

[26] Levin, J.M., Pascarella, S., Argos, P. & Garnier, J.: Quantification of secondary prediction improvement using multiple alignments. *Prot. Engng.*, 6, 1993, pp. 849–854.

[27] Light, R.J. & Margolin, B.H.: An analysis of variance for categorical data, *Journal of the American Statistical Association*, 66, n 331, 1971.

[28] Liu, W.Z. & White, A.P. (1994): The importance of Attribute Selection Measures in Decision Tree Induction. *Machine Learning 15: 25–41, 1994*. Kluwer Academic Publishers—Manufactered in the Netherlands, 1994.

[29] Müller, W. & Wysotzki, F. (1994): A Splitting Algorithm, Based on a Statistical Approach in the Decision Tree Algorithm CAL5. *Proc. of the ECML 94*, 7th European Conference on ML, Catania, Sicily, April 1994.

[30] Nakhaeizadeh, G. (1994): Interaction Between Machine Learning and Statistics, An Overview. *Proc. of the ECML 94*, 7th European Conference on ML, Catania, Sicily, April 1994.

[31] Nicolau, F. (1981): Critérios de análise classificatória hierárquica baseados na função de distribuição. Ph.D. thesis, Science Faculty of Lisbon.

[32] Qian, N. & Sejnowsky, T. (1988): Predicting the Secondary Structure of Globular Proteins Using Neural Network Models. *Journal of Molecular Biology*, 202, 865–884.

[33] Quinlan, J.R. (1986): Induction of Decision Trees. *Machine Learning*, pp. 81–106.

[34] Quinlan, J.R. (1993): *C4.5: Programs for Machine Learning* (1993). Morgan Kaufman. California.

[35] Rost, B. & Sander, C. (1993): Prediction of protein secondary structure at better than 70% accuracy. *J. Mol. Biol.* 232, pp. 584–599, 1993.

[36] Salzberg, S. & Cost, S.: A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10, 1992, pp. 57–78.

[37] Solovyev, V. & Salamov, A. (1994): Predicting a-helix and b-strand segments of globular proteins. *CABIOS*. Vol. 10 n° 6, pp. 661–669, 1994.

[38] Tallur, B. (1988): Contribution à l'analyse exploratoire de tableaux de contingence par la classification. Thèse de Doctorat Sciences, Université de Rennes I.

[39] Taylor, C.C., (1994): Distance-based Decision Trees. *Proc. of the ECML 94*, 7th European Conference on ML, Catania, Sicily.

[40] Van de Merckt, T. (1993): Decision Trees in Numerical Attribute Spaces. *Proc. of the IJCAI 93*, 13th International Joint C. On A.I., Chambéry, France.

[41] Ward, J.H. (1963): Hierarchical grouping to optimise an objective function. *Journal of American Statistical Association*, 58, pp. 236–244.

[42] Yi, T.M. & Lander, E.S.: Protein secondary structure prediction using nearest-neighbor methods. *Journal of Molecular Biology*, 232, 1993: pp. 1117–1129.

[43] Zhang, X., Mesirov, J. & Waltz, D. (1992): A hybrid system for protein secondary structure prediction. *Journal of Molecular Biology*, 25, 1049–1063.