HCAIM: A Discretizer for the Hierarchical Classification Scenario Applied to Bioinformatics Datasets

Valter Hugo Guandaline[1], Luiz Henrique de Campos Merschmann[2]

[1] Federal University of Ouro Preto, Brazil
vhguandaline@gmail.com
[2] Federal University of Lavras, Brazil
luiz.hcm@dcc.ufla.br

Abstract. Discretization is one of the stages of data preprocessing that has been the subject of research in several works related to flat classification. Despite the importance of data discretization for a classification task, to the best of our knowledge, when it comes to the hierarchical classification scenario, where the classes to be predicted are organized according to a hierarchy, there are no discretization methods in the literature that take class hierarchy into account. The development of discretization methods capable of dealing with class hierarchy is extremely important to enable the use of global hierarchical classifiers that require discrete data. Therefore, in this work, we fill this gap by proposing and evaluating a supervised discretization method for the hierarchical classification context. Experiments with 17 bioinformatics datasets using a global hierarchical classifier showed that the proposed method allowed the classifier to achieve predictive performance superior to those obtained when other unsupervised discretization methods were used.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications; I.2.6 [**Artificial Intelligence**]: Learning

Keywords: Discretization, hierarchical classification, CAIM

1. INTRODUCTION

Data mining is an integral step in a larger process known as KDD (*Knowledge Discovery in Database*), which also includes data preprocessing and post-processing of the mined information [Fayyad et al. 1996]. Since some data mining techniques request discrete input data or achieve better results when dealing with discrete data, the procedure of converting continuous data into discrete ones, named discretization, is an important step of the KDD process. In this work, we propose and evaluate a discretization method tailored for hierarchical classification datasets.

The main purpose of the preprocessing step is to prepare the dataset so that it can be used by some data mining technique. One of the processes that can be performed in this step is discretization. Its purpose is to transform continuous attributes into discrete ones. This transformation is done by associating intervals of continuous values with new categorical values. Thus, discretization methods reduce and simplify data, making the learning process faster and the results more compact [Garcia et al. 2013].

Classification is one of the main tasks of data mining. Its objective is to be able to generate, from a dataset containing instances with known characteristics and classes, models capable of predicting the class(es) of new instances from their characteristics. Most of the classification problems addressed in the literature are considered flat classification problems, where the classes do not have relationships with each other. However, there are more complex classification problems, known as hierarchical clas-

The authors thank UFOP, FAPEMIG and CNPq for their financial support, and STILINGUE for sharing its computational resources.

sification problems, where the classes to be predicted are structured according to a hierarchy [Freitas and de Carvalho 2007]. For instance, hierarchical classification is very important in bioinformatics, specially in gene and protein function prediction, where such functions are often organized as a hierarchy. In this scenario, higher-level classes are related to general functions, while lower-level ones correspond to more specific functions.

In spite of real-world applications often involve continuous attributes, some classification algorithms deal only with discrete attributes. In addition, even though some classification methods are able to handle continuous attributes, they perform better when continuous attributes are previously discretized [Kurgan and Cios 2004].

Although the literature is still scarce in studies concerning data preprocessing tailored for hierarchical classification datasets, in recent years some works related to attribute selection and missing attribute value imputation have been published (e.g., [Naik and Rangwala 2016], [Galvão and Merschmann 2016], [Kamal et al. 2015] and [Paes et al. 2014]). Nonetheless, to the best of our knowledge, there are no discretization methods in the literature that take into account the relationships between existing classes in hierarchical classification problems. In bioinformatics area, studies that addressed hierarchical classification problems and required data discretization, such as [Merschmann and Freitas 2013] and [Silla Jr and Freitas 2009], had to use unsupervised discretization methods, since unsupervised methods can be used to the context of both flat and hierarchical classification. This has motivated us to propose a supervised discretization method for the context of hierarchical classification applied to gene and protein function prediction, which are important real world problems in bioinformatics.

In [Dougherty et al. 1995], the authors have shown that, for flat classification scenario, supervised discretization methods are usually better than unsupervised methods when considering the effect of discretization on the classifiers accuracy. In addition, they have reported that the accuracy of the flat Naive Bayes classifier significantly improved when attributes were discretized using a supervised discretization method. Thus, the hypothesis raised in this work is that supervised discretization methods, due to the fact that they take into account the class attribute at the time of discretization, could provide more accuracy improvement of a classifier, that is an extension of the Naive Bayes algorithm to handle hierarchical classification problems, than unsupervised discretization methods.

The proposal presented here corresponds to an adaptation made in the CAIM discretization method [Kurgan and Cios 2004] to make it consider the existing class hierarchy in hierarchical classification problems. In order to test our hypothesis, the proposed supervised method has been compared with two unsupervised discretization methods using 17 bioinformatics datasets related to gene and protein function prediction. The results show that, for most cases, the proposed method allowed a hierarchical Naive Bayes classifier to achieve predictive performance (in terms of hierarchical F-measure) superior to those obtained when the dataset was preprocessed by unsupervised methods.

The remainder of this article is organized as described below. Section 2 presents a brief review of the literature on hierarchical classification and data discretization. Then, the proposed method is detailed in Section 3 and the computational experiments with the results obtained are described in Section 4. Finally, Section 5 presents the conclusions of this work and points out future work.

2. THEORETICAL FRAMEWORK

2.1 Hierarchical Classification

In a hierarchical classification problem, the relationships between the classes are represented by a hierarchical structure which can be a tree or a direct acyclic graph (DAG). The main difference between these structures is that in a tree a node (class) is associated with at most one parent node, while in a DAG, a node can have more than one parent node.

According to [Freitas and de Carvalho 2007], hierarchical classification methods differ in a number of aspects. The first aspect refers to the type of structure the method is able to deal with. In the case of this work, the hierarchical structure of the classes corresponds to a tree.

The second aspect is related to the depth of the execution of the classification in the hierarchy. A method can perform predictions using only classes at leaf nodes in the class hierarchy (Mandatory Leaf-Node Prediction – MLNP) or classes referring to any node (internal or leaf) of the hierarchy (Non-Mandatory Leaf-Node Prediction – NMLNP). In this work, we consider the NMLNP scenario.

The third aspect is related to the number of different path labels in the hierarchy a method can assign an instance to. A method can be able to predict multiple classes for a particular instance (multi-label), thereby involving multiple paths of labels in the class hierarchy, or just a class (single-label), which will be linked to a single path of labels in the class hierarchy. The method proposed in this article deals with the single-label classification.

Finally, the fourth aspect is related to the type of approach that the classifier uses to explore the hierarchical structure. According to [Silla Jr and Freitas 2011] there are three types of approaches: (i) flat classification approach, in which the class hierarchy is ignored and the predictions are performed considering only the leaf nodes classes of the hierarchical structure; (ii) local approach, where several traditional flat classifiers are used, each with a local view of the hierarchical structure (e.g., [D'Alessio et al. 2000] and [Koller and Sahami 1997]); and (iii) global approach, where a single classification model is built taking the entire class hierarchy into account at once (e.g., [Labrou and Finin 1999], [Qiu et al. 2009] and [Silla Jr and Freitas 2009]). The discretization method proposed in this work aims to adapt the datasets to be used by global hierarchical classifiers, given that, for the local approach, supervised discretization methods designed for the flat classification scenario can be used.

2.2 Data Discretization

Discretization is a data reduction strategy widely used in the data preprocessing step [Garcia et al. 2013]. The discretization process transforms continuous attributes into discrete ones by dividing them into intervals and associating each of these intervals to a different discrete value.

According to [Garcia et al. 2013], discretization methods can be categorized as supervised or unsupervised. A method is called supervised when its execution takes into account the values of the class attribute. On the other hand, if the class attribute is not considered in the discretization process, the method is said to be unsupervised.

Different criteria can be used to evaluate discretization algorithms, such as the number of intervals generated, the level of inconsistency and the accuracy of classifiers. In this work, the discretization methods were evaluated from the global hierarchical classifier named Global Model Naive Bayes (GMNB), proposed in [Silla Jr and Freitas 2009].

In [Garcia et al. 2013] the authors evaluated 30 discretizers on 40 datasets using six flat classifiers. This evaluation showed that the CAIM was one of the most efficient discretization methods. Therefore, it was the method chosen in this work to be adapted to the hierarchical context. In addition, given the lack of supervised methods for the hierarchical context, the unsupervised methods Equal-Width and Equal-Frequency (adopted in [Merschmann and Freitas 2013] and [Silla Jr and Freitas 2009]) were used as baseline for comparison with the method proposed here. Next, more details of these unsupervised methods and CAIM discretization method, which has been adapted to the hierarchical classification context, will be presented.

2.2.1 *Unsupervised Discretization Methods.* Equal-Width and Equal-Frequency are unsupervised binning methods, as sorted attribute values are distributed into a number of bins, and then each bin value is replaced by a label.

Equal-Width method divides the attribute's range into k uniform sized bins. The width of each bin is:

$$w = (max\_value - min\_value)/k, \tag{1}$$

where min_value and max_value are the minimum and maximum values of the attribute to be discretized, respectively. In this way, the bin boundaries are $\{min\_value + w, min\_value + 2 * w, \ldots, min\_value + (k-1) * w\}$.

Equal-Frequency method divides the attribute's range into k bins, where each bin contains approximately the same number of values. Given an attribute with N values, each bin contains approximately N/k values. For both methods, k is a parameter specified by user.

2.2.2 *CAIM.* Class-Attribute Interdependency Maximization is a supervised discretization method that uses a metric to evaluate the interdependence between the class attribute and the discretized attribute [Kurgan and Cios 2004].

Consider a dataset composed of a set of instances M, characterized by a set of continuous attributes F and a class attribute S, where$|M|$, $|F|$ and $|S|$ are, respectively, the number of instances, number of continuous attributes and number of classes. In addition, each instance M_k is associated with a class S_i, where $k \in \{1, 2, ..., |M|\}$ and $i \in \{1, 2, \ldots, |S|\}$.

For each continuous attribute F_j, $j \in \{1, 2, \ldots, |F|\}$, the CAIM method sorts its values in ascending order and, after, divides them into n intervals as follows: $D = \{[d_0, d_1], (d_1, d_2], \ldots, (d_{n-1}, d_n]\}$, where d_0 e d_n, are respectively, the minimum and maximum values of the attribute F_j and $d_i < d_{i+1}$ for $i \in \{0, 1, \ldots, n-1\}$. Each pair of values $(d_i,\ d_{i+1})$ defines an interval of the attribute F_j, where the discretization result D, called discretization scheme of the attribute F_j, defines the following set of cut points $P = \{d_1, d_2, \ldots, d_{n-1}\}$, which are the midpoints of all adjacent pairs in the set D.

The interdependence between the class attribute S and a discretized attribute F_j, according to a discretization scheme D, is calculated using the CAIM metric (Equation 2), which makes use of a frequency matrix, called contingency matrix, shown in Figure 1. In this matrix, which represents a discretization scheme $D = \{[d_0, d_1], (d_1, d_2], \ldots, (d_{n-1}, d_n]\}$ of the discretized attribute F_j, q_{ir} is the number of instances belonging to the i-th class that are contained in the r-th interval, M_{i+} is the total number of instances belonging to the i-th class and M_{+r} is the total number of instances contained in the r-th interval.

$$CAIM(S, D|F_j) = \frac{\sum_{r=1}^{n} \frac{max_r^2}{M_{+r}}}{n}, \tag{2}$$

where n is the number of intervals and max_r is the maximum number of instances contained in the interval r belonging to the same class. This equation is used by the CAIM method to choose the best cut point to be inserted in a given discretization scheme. The higher the value returned by this metric, the greater the dependency between the F_j (discretized according to the D scheme) and the class attribute S.

CAIM algorithm can be divided into three stages: initialization, evaluation and verification. These steps are applied to each continuous attribute F_j. Next, we show each of them in detail.

Initialization: The first step of CAIM method is to identify the minimum (d_0) and maximum (d_n) values of the attribute F_j and create the initial discretization scheme containing only one interval limited by these two values ($D = \{[d_0,\ d_n]\}$), that is, the method starts with a single interval ($K = 1$) containing all values of the attribute F_j. The CAIM metric assigned to this interval is 0 ($GlobalCaim = 0$). Then, the method initializes the set of possible cut points B. This set is formed by calculating the midpoints between all the adjacent values of the set of distinct values of the attribute F_j arranged in ascending order. For example, if $U = \{1, 2, 3, 4, 5\}$ is the set of distinct values of F_j,

Classes	Intervals					Instances per Class
	$[d_0, d_1]$	...	$(d_{r-1}, d_r]$	...	$(d_{n-1}, d_n]$	
C_1	q_{11}	...	q_{1r}	...	q_{1n}	M_{1+}
...	...	...	...	...	...	...
C_i	q_{i1}	...	q_{ir}	...	q_{in}	M_{i+}
...	...	...	...	...	...	...
C_s	q_{s1}	...	q_{sr}	...	q_{sn}	M_{s+}
Instances per Interval	M_{+1}	...	M_{+r}	...	M_{+n}	M

Fig. 1. Contingency matrix for the attribute F_j with discretization scheme D

#	F_1	Class
01	1.2	A
02	1.2	A
03	1.8	A
04	1.8	A
05	1.8	A
06	3.2	B
07	3.2	B
08	3.2	B
09	3.8	B
10	3.8	C
11	5.2	C
12	5.2	C

Fig. 2. Sample Dataset

then the set of possible cut points is $B = \{1.5; 2.5; 3.5; 4.5\}$. After the definition of the set B, the method moves to the evaluation step.

Evaluation: This is an iterative step which consists in evaluating all cut points contained in B while the stop criterion is not satisfied. For each cut point p in the set B, the method creates a new discretization scheme D' by inserting the cut point p into the discretization scheme D. Then, the scheme D' is evaluated by the metric CAIM(S, $D'|F_j$). After evaluating all cut points contained in B, the method stores the cut point $p*$ that obtained the highest value for the CAIM evaluation criterion. This information is used in the verification step.

Verification: In this step, the method stop criterion is checked. The algorithm terminates its execution when the following two conditions are false: i) if the number of intervals generated so far (k) is less than the number of classes ($|S|$); ii) if the value of the CAIM metric for the cut point $p*$ is greater than the one obtained in the previous iteration ($GlobalCaim$). If the opposite is true, the algorithm removes the cut point $p*$ from the set B and adds it to the scheme D, then it increases the number of intervals created ($k = k + 1$), updates the value of the CAIM metric for the scheme D ($GlobalCaim = CAIM$), and finally returns to the evaluation stage, where the insertion of a new cut point will be evaluated.

In order to illustrate the application of the CAIM method, consider the dataset presented in Figure 2, where the column # contains the number of each instance, the column F_1 corresponds to the continuous attribute to be discretized and the column $Class$ contains the class of each instance. The dashed horizontal lines illustrate the possible cut points, which define the initial vector B as $B = \{1.5; 2.5; 3.5; 4.5\}$. Its worth noting that the dataset is already ordered according to the values of the continuous attribute F_1.

Notice that the dataset of Figure 2 contains: 12 instances ($|M| = 12$); 3 classes ($|S| = 3$), where $S = \{\text{A, B, C}\}$; and one continuous attribute ($|F| = 1$), where $F = \{F_1\}$. In addition, F_1 contains 5 distinct values ($q = 5$), forming the set of distinct values $U = \{1.2; 1.8; 3.2; 3.8; 5.2\}$.

Classes	Intervals		Instances per Class
	[1.2; 1.5]	(1.5; 5.2]	
A	2	3	5
B	0	4	4
C	0	3	3
Instances per Interval	2	10	12

Fig. 3. Contingency matrix for the scheme D'

p	Scheme D'	CAIM metric
2.5	{[1; 2.5], (2.5; 5]}	$(5^2/5 + 4^2/7)\ /\ 2 = 3.64$
3.5	{[1; 3.5], (3.5; 5]}	$(5^2/8 + 3^2/4)\ /\ 2 = 2.68$
4.5	{[1; 4.5], (4.5; 5]}	$(5^2/10 + 2^2/2)\ /\ 2 = 2.25$

Fig. 4. Evaluations of cut points in the first iteration of CAIM

The first step of the CAIM method is to identify the minimum (d_0) and maximum (d_n) values of the attribute F_1 and create the initial discretization scheme containing only one interval ($k = 1$) bounded by these two values ($D = \{[1.2; 5.2]\}$). Furthermore, this initial discretization scheme is assigned the value 0 for the CAIM metric ($GlobalCaim = 0$).

Afterwards, the method initializes the set of possible cut points B. Being $U = \{1.2; 1.8; 3.2; 3.8; 5.2\}$ the set of distinct values of attribute F_1, then the set of possible cut points will be $B = \{1.5; 2.5; 3.5; 4.5\}$. Thus, we have the following definitions necessary for the evaluation step: $D = \{[1.2; 5.2]\}$; $GlobalCaim = 0$; $B = \{1.5; 2.5; 3.5; 4.5\}$; $|S| = 3$ and $k = 1$.

On the first iteration, the CAIM method generates a scheme D' inserting in $D = \{[1.2; 5.2]\}$ the first cut point p contained in B ($p = 1.5$), resulting in $D' = \{[1.2; 1.5], (1.5; 5.2]\}$. Figure 3 presents the contingency matrix for the scheme D'.

From the contingency matrix the method evaluates this scheme D' using the CAIM metric (Equation 2). The calculation of the metric for the contingency matrix of Figure 3 is presented below.

$$CAIM = \frac{\frac{2^2}{2} + \frac{4^2}{10}}{2} = 1.8 \tag{3}$$

Similarly, the method evaluates all other possible schemes D' generated from the insertion (in $D = \{[1.2; 5.2]\}$) of the other existing cut points in B. Table 4 presents the evaluation of the other schemes D' evaluated in the first iteration of the CAIM.

After evaluating all the possible schemes D' generated from the existing cut points in B, the method checks whether the CAIM stopping criterion has been satisfied. In this case, since k is less than $|S|$ ($1 < 3$), the method continues its execution by choosing the cut point that generated the highest value for the CAIM metric, updating the value of $GlobalCaim$, removing this cut point of the set B, updating the scheme D with the chosen cut point, and finally, adjusting the value of k. In this iteration, the cut point $p = 2.5$ was the one that obtained the best evaluation (CAIM = 3.64) and, therefore, was used to update the scheme D. Thus, the following updates are performed before the method begins the second iteration: $D = \{[1; 2.5], (2.5; 5]\}$; $GlobalCaim = 3.64$; $B = \{1.5; 3.5; 4.5\}$; $k = 2$.

In the second iteration, the entire process of evaluating the possible discretization schemes D', generated from the insertion in D of each of the existing cut points in B, is done again. Table 5 presents the evaluation of each possible scheme D' in the second iteration of the method.

p	Scheme D'	CAIM metric
1.5	{[1; 1.5], (1.5; 2.5], (2.5; 5]}	$(2^2/2 + 3^2/3 + 4^2/7)$ / 3 = 2.42
3.5	{[1; 2.5], (2.5; 3.5], (3.5; 5]}	$(5^2/5 + 3^2/3 + 3^2/4)$ / 3 = 3.41
4.5	{[1; 2.5], (2.5; 4.5], (4.5; 5]}	$(5^2/5 + 4^2/5 + 2^2/2)$ / 3 = 3.40

Fig. 5. Evaluations of cut points in the second iteration of CAIM

p	Scheme D'	CAIM metric
1.5	{[1; 1.5], (1.5; 2.5], (2.5; 3.5], (3.5; 5]}	$(2^2/2 + 3^2/3 + 3^2/3 + 3^2/4)$ / 4 = 2.26
4.5	{[1; 2.5], (2.5; 3.5], (3.5; 4.5], (4.5; 5]}	$(5^2/5 + 3^2/3 + 1^2/2 + 2^2/2)$ / 4 = 2.62

Fig. 6. Evaluations of cut points in the third iteration of CAIM

#	AF_1	Class
01	1.2	A
02	1.2	A
03	1.8	A
04	1.8	A
05	1.8	A
06	3.2	B
07	3.2	B
08	3.2	B
09	3.8	B
10	3.8	C
11	5.2	C
12	5.2	C

(A)

#	F_1	Class
01	[1.2; 2.5]	A
02	[1.2; 2.5]	A
03	[1.2; 2.5]	A
04	[1.2; 2.5]	A
05	[1.2; 2.5]	A
06	(2.5; 3.5]	B
07	(2.5; 3.5]	B
08	(2.5; 3.5]	B
09	(3.5; 5.2]	B
10	(3.5; 5.2]	C
11	(3.5; 5.2]	C
12	(3.5; 5.2]	C

(B)

Fig. 7. **(A)**: Original dataset. **(B)**: Discretized dataset.

After evaluating the schemes D', the method checks again whether the CAIM stopping criterion has been reached. In this case, since k continues to be less than $|S|$ ($2 < 3$), the method follows by inserting in D the cut point ($p = 3.5$) that generated the best valuation ($CAIM = 3.41$) and updating the other variables. Therefore, at the end of the second iteration we have: $D = \{[1; 2.5], (2.5; 3.5], (3.5; 5]\}$; $GlobalCaim = 3.41$; $B = \{1.5; 4.5\}$; $k = 3$.

The third iteration of the method is done by adopting the same procedure described for the previous iterations. Table 6 resents the evaluation of all the possible schemes D' generated in this iteration.

Next, the method checks again whether the CAIM stopping criterion has been satisfied. As in this case k is not less than $|S|$ ($k = 3$ and $|S| = 3$) and the highest value of CAIM metric obtained in this iteration is not greater than $GlobalCaim$ ($2.62 < 3.41$), the method no longer updates the discretization scheme D and ends its execution. Therefore, the discretization of the attribute F_1 is finished. Figure 7 shows the result of the discretization of the attribute F_1 **(B)** of the original dataset **(A)**. The horizontal lines in the discretized dataset **(B)** represent the cut points used in the discretization of the attribute F_1.

3. PROPOSED METHOD

The main problem in using traditional supervised discretization methods (used in conjunction with flat classifiers) for databases related to the hierarchical classification context is in the fact that these discretizers are not able to consider the information of the relationships between the classes of the problem. In this work, it is assumed that this type of information, if considered along the discretization process, may contribute to the generation of a better quality discretized dataset for the classification task.

Therefore, the discretization method proposed here, called HCAIM (Hierarchical CAIM), considers the class hierarchy while performing the discretization process. The HCAIM corresponds to an adaptation of the CAIM discretization method for the hierarchical context, whose main difference is in the evaluation metric used by the method for the definition of cut points of a discretization scheme.

3.1 Evaluation Metric

To evaluate a discretization scheme $D = \{[d_0, d_1], (d_1, d_2], \ldots, (d_{n-1}, d_n]\}$ for an attribute F_j, the CAIM method checks how good the intervals are in this scheme. Through the metric also called CAIM (see Equation 2), each interval contained in D is evaluated by measuring the correlation between the values of the attribute F_j existing in that interval and the classes contained in it. This correlation is given by $(\frac{max_r^2}{M_{r+}})$, where max_r is the number of occurrences of the most frequent class in the interval r and M_{r+} is the number of instances contained in that same interval. This calculation allows the CAIM method to: (i) consider the degree of purity of the interval (the nearer max_r is to M_{r+}, the purer the interval) and (ii) prioritize intervals with higher number of instances.

However, this CAIM metric does not take into account the existing class hierarchy in a problem where classes are hierarchically organized. For example, given an interval r containing 9 instances ($M_{r+} = 9$), being 3 instances of the class *R.2* and 6 of the class *R.2.1*, the evaluation of that interval according to the CAIM metric is given by $6^2/9 = 4$, since the majority class $R.2.1$ is considered as completely distinct from $R.2$. However, in the hierarchical context, instances of the class *R.2.1* also belong to the class *R.2*, since $R.2.1$ is a child class of $R.2$.

Therefore, in this work, the CAIM metric was adapted to calculate the degree of purity of each interval considering the class hierarchy. In the proposed adaptation, called HCAIM (Hierarchical CAIM), the calculation of the purity of an interval is performed for each hierarchical level, its final value being a weighted average of the values calculated for each one of the levels. Thus, the dependence between the class attribute S and the discretization scheme D for a given attribute F_j taking into account the class hierarchy is given by:

$$HCAIM(S, D|F_j) = \frac{\sum_{r=1}^{n} \sum_{l=1}^{H_r} \frac{max_{r,l}^2}{M_{+r}} . W_{l,r}}{n}, \tag{4}$$

where n is the number of intervals, H_r is the depth of the class hierarchy referring to the interval r, $max_{r,l}$ is the number of occurrences of the most frequent class in the interval r considering the class hierarchy up to the level l, M_{+r} is the total of instances contained in the interval r e $W_{l,r}$ is weight associated with the level l of the class hierarchy for the interval r.

When calculating the HCAIM metric for a given interval r, in addition to the contingency matrices for each hierarchical level, it is necessary to calculate the weights $W_{l,r}$ that will be applied according to the hierarchical level l and the depth of the hierarchy H_r in that interval. The weight value for each of the hierarchical levels is given by:

$$W(l, r) = (H_r - l + 1) \frac{2}{H_r \times (H_r + 1)} \tag{5}$$

given that $\sum_{l=1}^{H_r} W_{l,r} = 1$.

Going back to the previous example, where we consider a single interval r containing 9 instances ($M_{r+} = 9$), being 3 instances of the class *R.2* and 6 of the class *R.2.1*, the evaluation of this interval according to the HCAIM metric is given by $9^2/9 \times W_{1,r} + 6^2/9 \times W_{2,r}$. The first portion ($9^2/9$) is due to the fact that we consider all the classes present in the interval r only up to the first level of the hierarchy, that is, all instances are associated with the class $R.2$. The second portion ($6^2/9$) is calculated by considering all classes up to the second hierarchical level, where we have 3 instances of the

class *R.2* and 6 of the class *R.2.1*. Considering that the weights associated with the hierarchical levels are $W_{1,r} = 2/3$ and $W_{2,r} = 1/3$, the final metric value for the example in question is $HCAIM = 7.33$.

3.2 HCAIM Method

In this work, the CAIM discretization method was adapted to the hierarchical classification context. The main alteration occurred in the evaluation metric used to define the cut points of the discretization scheme (see Section 2.2.2), so that in HCAIM this metric takes into account the class hierarchy of the problem. In addition to the metric adaptation, another change was made in the way the set of cut points B was initialized for the process of the discretization of a given attribute.

The pseudocode of the HCAIM method is shown in Figure 8. As input, the algorithm receives a dataset consisting of N instances, $|S|$ distinct classes, and continuous attributes F_i. Basically, for each attribute F_i to be discretized, the algorithm performs two steps: a) initialization of the set of possible cut points B and of the discretization scheme D; b) consecutive insertions of cut points in the discretization scheme D from their evaluations by the adapted metric HCAIM. The detail of each of these stages is described below.

The initialization step is performed for each attribute F_i (lines 3 to 7). In this step, the first initialization (line 3) is that of the set of cut points B. Considering that the attribute to be discretized F_i is ordered, the cut points inserted in the set B correspond to the average of the values of the attribute F_i for each pair of neighboring instances that are associated to distinct classes and have different values for the attribute in question. Then, the discretization scheme D is initialized (line 4) with a single interval [-∞,+∞]. Finally, between lines 5 and 7, the variables *globalHCaim* (stores the best value of HCAIM metrics throughout the discretization process), k (controls the number of cut points inserted in scheme D) and *stop* (controls the finalization of the discretization process of the attribute F_i) are also initialized.

Once the initializations described above have been achieved, while the stopping criterion is not reached, new cut points are consecutively inserted into the discretization scheme D (lines 8 to 27). The insertion of a new cut point in the discretization scheme is done by choosing, at each iteration, the cut point contained in B which is best evaluated by the HCAIM metric (lines 11 to 19). The *localHCaim* variable, initialized in the line 10, is responsible for storing the best value of the HCAIM metric along the process of choosing the best cut point contained in B. Every time a new cut point is inserted into the scheme D (line 22), it is also removed from the set of cut points B (line 23). The method stop criterion (line 20) establishes that new cut points must be entered in the discretization scheme D while the number of intervals in D is less than the number of distinct classes of the dataset or while the insertion of a new cut point improves the best value already obtained for the metric of evaluation (stored in *globalHCaim*). When the stopping criterion is reached, the discretization scheme D of the attribute F_i is stored in a list of schemes (line 28) and if there are other continuous attributes in the dataset, this whole discretization process is performed again.

Figure 9 presents the pseudocode of the algorithm used to calculate the HCAIM metric, which receives as input the dataset BD, the scheme T and the attribute being discretized F_i. For each interval I in the discretization scheme T, lists L_k (line 8) are built containing the frequency of the classes (considering them up to the level k of the class hierarchy) associated to the instances whose value of the attribute F_i is contained in the interval I. From these lists L_k, the metric value for each interval I is calculated (lines 11 and 12) using: a) the highest value contained in L_k, that is, the one associated with the most frequent class (line 10); b) the total number of instances whose value of the attribute F_i is contained in the interval I (line 6) and c) weight assigned to the level k of a hierarchical structure with depth H (line 9). The final value of the metric corresponds to the average of the HCAIM values calculated for each interval I (line 16).

```
Algorithm HCAIM (Dataset BD)
 1: Initialize SchemeList = ∅;
 2: for each continuous attribute F_i do
 3:    Initialize the set of cut points B of attribute F_i;
 4:    Create the initial discretization scheme D = {[−∞, +∞]};
 5:    Initialize globalHCaim = 0;
 6:    Initialize k = 1;
 7:    Initialize stop = FALSE;
 8:    while !stop do
 9:       stop = TRUE;
10:       Initialize localHCaim = 0;
11:       for each cut point c ∈ B do
12:          T = D;
13:          Insert the cut point c into the scheme T;
14:          hcaim = metricHCAIM(BD, T, F_i); //see Figure 9
15:          if (hcaim > localHCaim) then
16:             localHcaim = hcaim;
17:             p = c;
18:          end if
19:       end for
20:       if (localHCaim > globalHCaim or k < |S|) then
21:          globalHCaim = localHCaim;
22:          Insert the cut point p into the scheme D in ascending order;
23:          Remove the cut point p form the set B;
24:          stop = FALSE;
25:       end if
26:       k = k + 1;
27:    end while
28:    Insert the scheme D into SchemeList;
29: end for
end.
```

Fig. 8. HCAIM method's pseudocode.

```
Algorithm metricHCAIM(Dataset BD, Scheme T, Attribute F_i)
 1: Initialize hcaim = 0.0;
 2: Initialize numberOfIntervals = 0;
 3: for each interval I ∈ T do
 4:    S = Set of instances of the dataset BD whose value of the attribute F_i ⊂ I;
 5:    H = Depth of class hierarchy associated with the instances ∈ S;
 6:    M = Total number of instances contained in S
 7:    for k = 1 to H do
 8:       Create the list L_k containing the frequency of the classes associated to the instances ∈ S, considering
          them up to the level k of the class hierarchy;
 9:       W = calculateWeight(k, H);
10:       max = highest value contained in L_k;
11:       caimInterval = (max^2 / M) * W;
12:       hcaim = hcaim + caimInterval;
13:    end for
14:    numberOfIntervals = numberOfIntervals + 1;
15: end for
16: Return ( hcaim / numberOfIntervals);
end.
```

Fig. 9. Pseudocode of the algorithm used in the calculation of the HCAIM metric.

In the next section we present the computational experiments performed in order to evaluate the proposed method.

4. COMPUTATIONAL EXPERIMENTS

4.1 Datasets

All experiments were conducted from 17 datasets, 9 of which are related to classification of gene functions and 8 with classification of protein functions. As these datasets were obtained from different sources, they were organized into two groups.

Group A consists of 9 datasets of gene functions, in this case related to the yeast genome. On these datasets, predictor attributes include several types of bioinformatics data, such as sequence secondary structure, phenotype, homology, sequence statistics, and expression. These datasets, originally used in [Clare and King 2003], are multi label. As the focus of this work is the hierarchical single label classification, these datasets have been transformed into single label data by selecting, for each instance, the most frequent class in the original dataset.

Group B consists of 8 protein function datasets related to two different protein families: Enzymes and G-Protein-Coupled Receptors (GPCR). Enzymes are proteins that catalyze chemical reactions, while GPCRs are proteins that play central roles in biochemical and cellular processes, acting as molecular targets for various medical drugs. Four databases of enzymes (whose names begin with EC - Enzyme Commission) and four GPCR databases were used, where the predictive attributes correspond to protein properties and the classes to be predicted are hierarchical protein functions. Most of the predictive attributes are binary, indicating whether a protein signature (or motif) is present in a protein, and two attributes are continuous: amino acid sequence length and molecular weight. The names of the datasets are associated with the type of motif used: Interpro Entries, FingerPrints, Prosite Patterns and Pfam. These datasets have already been used in other hierarchical classification works, such as [Costa et al. 2007], [Holden and Freitas 2008] and [Silla Jr and Freitas 2009].

From the single label databases, an initial preprocessing was performed to replace the missing attribute values. By identifying a missing value for a given attribute F_j of an instance associated with the class C_i, we calculate the average of the observed values of the attribute F_j of all other instances of the dataset associated with the C_i class, then this mean is used to replace the missing value. If for class C_i no instance is found to have an observed value for the attribute F_j, the average of the observed values of the attribute F_j of all dataset instances associated with descendant classes of C_i in the hierarchy is calculated, then this average is used to replace the missing value. Ultimately, if class C_i does not have descendant classes or if for the descendant classes of C_i no instance has an observed value for the attribute F_j, then the missing value is replaced by the global average of attribute F_j.

In addition, prior to the execution of the classification algorithm, a second preprocessing was performed. In this preprocessing, each class with fewer than 10 instances was merged with its parent class. This process was repeated until every class in the hierarchy had at least 10 instances. If during this process the most specific class of an instance has become the Root class, then the instance has been removed from the database.

Table I shows the main characteristics of the datasets after these preprocessing steps. This table presents, for each dataset, the number of instances, the number of predictor attributes, the number of classes and their distribution by hierarchy levels ($1°|2°|3°|\ldots$).

4.2 Experimental Setup

Unsupervised discretization methods Equal-Frequency (EF) and Equal-Width (EW) were used as reference for comparison with the proposed method, HCAIM. These methods were chosen for the comparisons because they have already been adopted in hierarchical classification works, since there are no supervised discretization methods for this context.

Table I. Databases Characteristics

Databases	# Instances	# Attributes Continuous / Categorical	# Classes	# Classes by Level
Church	3755	26 / 1	190	7|37|72|47|25|2
Cellcycle	3757	77 / 0	190	7|37|73|46|25|2
Eisen	2424	79 / 0	143	4|26|55|34|22|2
Expr	3779	547 / 0	191	7|37|72|47|26|2
Gasch2	3779	52 / 0	191	7|37|73|46|26|2
Gasch1	3764	173 / 0	191	7|37|73|46|26|2
Derisi	3725	63 / 0	190	7|37|72|47|25|2
Spo	3703	77 / 3	191	7|37|73|46|26|2
Seq	3919	473 / 5	192	7|37|73|47|26|2
EC-Interpro	5192	2 / 1214	105	6|29|47|70
EC-Pfam	5192	2 / 706	105	6|29|47|70
EC-Prints	5192	2 / 380	105	6|29|47|70
EC-Prosite	5192	2 / 583	105	6|29|47|70
GPCR-Interpro	5156	2 / 448	149	7|42|74|49
GPCR-Pfam	5156	2 / 73	149	7|42|74|49
GPCR-Prints	5156	2 / 281	149	7|42|74|49
GPCR-Prosite	5156	2 / 127	149	7|42|74|49

EF and EW methods were executed from their implementations available in the *WEKA* tool [Hall et al. 2009]. As these methods have the parameter k, which defines the number of intervals (bins) to be created in the discretization process, in order to have a fair comparison with the proposed HCAIM, which is a method without parameters, we have adopted two comparison strategies. In the first batch of experiments HCAIM was compared to EF and EW using different values of k, namely, 5, 10, 15 and 20. After, in a second batch of experiments, EF and EW methods were compared to HCAIM using, for each attribute, the same number of intervals chose by HCAIM.

With the aim of evaluating the quality of the discretization performed by each of the methods considered here, the global hierarchical classifier Global Model Naive Bayes (GMNB) [Silla Jr and Freitas 2009] was used. In order to express the predictive performance of the GMNB the hierarchical metric *F-measure* (hF) proposed in [Kiritchenko et al. 2005] was adopted. In addition, the k-fold cross-validation method ($k = 10$) was used in the performance evaluation of the GMNB. Thus, the results presented in the next section correspond to averages of 10 executions. It is also worth mentioning that the data discretization occurred only after the partitioning of the dataset by the method 10-fold cross-validation, that is, for each dataset, it was applied considering each of the 10 training partitions.

4.3 Results

The results of comparative experiments are presented in Tables II and III. Table II contains the results of the first batch of experiments, where EF and EW using four values of parameter k (5, 10, 15 and 20) are compared to HCAIM. Next, Table III presents the results of the second batch of experiments, where the same number of intervals are considered for each attribute discretized by EW, EF and HCAIM.

Table II presents the average hierarchical Precision (hP), average hierarchical Recall (hR) and average hierarchical *F-measure* (hF) (with standard deviation of hF in parentheses) obtained by the GMNB classifier for each discretized dataset using the HCAIM and the other two methods used as reference, namely, Equal-Frequency (EF) and Equal-Width (EW). In the case of the discretization methods EF and EW, the column name is formed by the name of the method plus, in parentheses, the value of the adopted parameter k. For each dataset, in order to verify if there is difference with statistical significance between the predictive performances (hF) of the GMNB classifier when processing the dataset discretized by HCAIM and another reference method, we used the Wilcoxon statistical test with the Bonferroni correction due to the multiple comparisons between HCAIM and each reference

method [Japkowicz and Shah 2011]. This statistical test was performed with a confidence level of 95%. The values in bold indicate the best result obtained for each dataset. In addition, the symbol • shows that there is a difference with statistical significance between the reference method in question and HCAIM. Finally, the last line of this table summarizes the result of the statistical test, i.e. for each reference method, it is shown the number of times that the HCAIM outperformed it presenting a better predictive performance (hF) of the classifier GMNB.

The results presented in Table II show that the discretization method proposed in this work (HCAIM) has provided the largest predictive performance (hF) to GMNB for 11 of the 17 datasets used in the experiments (bold values). In addition, the statistical tests show that, for 8 datasets, HCAIM has outperformed all reference methods used in the comparative experiments. For only 5 datasets (Eisen, Seq, SPO, EC-Interpro and EC-Prints) the HCAIM obtained a statistically significant lower performance than some reference method.

Statistical tests also show that when comparing HCAIM with each of the other methods used in the experiments, it presents a statistically superior performance or at least the same as the other methods for most of databases evaluated. For example, when compared to Equal-Frequency method with $k = 5$ (EF (5)), the HCAIM is better for 12 datasets, equivalent in 3 and worse in only 2 datasets.

Finally, summing up the whole comparative evaluation, from Table II, we can observe that of the total of 136 comparisons, the HCAIM method was shown to be superior in 88 comparisons, equivalent in 31 and lower in only 17. Therefore, in the first batch of comparative experiments, the statistical tests confirm the superiority of HCAIM in relation to the other discretization methods used in the experiments.

Table III depicts the results of the second batch of comparative experiments, where Equal-Frequency (EF) and Equal-Width (EW) methods were ran using, for each attribute, the same number of intervals chose by HCAIM. This table shows average hierarchical Precision (hP), average hierarchical Recall (hR) and average hierarchical *F-measure* (hF) (with standard deviation of hF in parentheses) obtained by the GMNB classifier for each dataset discretized by HCAIM, EF and EW. For each dataset, in order to verify if there is difference with statistical significance between the predictive performances (hF) of the GMNB classifier when processing the dataset discretized by HCAIM and another reference method, we used the Wilcoxon statistical test with the Bonferroni correction due to the multiple comparisons between HCAIM and each reference method [Japkowicz and Shah 2011]. This statistical test was performed with a confidence level of 95%. The values in bold indicate the best result obtained for each dataset. In addition, the symbol • shows that there is a difference with statistical significance between the reference method in question and HCAIM. Finally, the last line of this table summarizes the result of the statistical test, i.e. for each reference method, it is shown the number of times that the HCAIM outperformed it presenting a better predictive performance (hF) of the classifier GMNB.

Similarly to the results obtained from the first batch of experiments, the statistical test results presented in Table III show that, for 8 datasets, HCAIM has outperformed all reference methods used in the comparative experiments. However, now, for only 3 datasets (Seq, EC-Prints, GPCR-Prosite) the HCAIM obtained a statistically significant lower performance than some reference method. In addition, for other 5 datasets all discretization methods are statistically equivalent.

In the comparison between HCAIM and each of the reference methods (EF and EW), HCAIM presents a performance (hF) statistically superior or equivalent to the other methods for most of datasets evaluated. When compared to EF method, the HCAIM is better for 8 datasets, equivalent in 8 and worse in only 1 dataset. In the comparison with EW, HCAIM outperformed it in 10 datasets, was equivalent in 5 and achieved lower performance in only 2 datasets.

Finally, summing up the whole comparative evaluation presented in Table III, we can observe that of the total of 34 comparisons, the HCAIM method was shown to be superior in 18 comparisons, equivalent in 13 and lower in only 3. Therefore, in the second batch of comparative experiments, the

Table II. Mean values of hP, hR and hF obtained by GMNB after discretization of the datasets.

Base	EF(5) hP/hR/ hF (SD)	EF(10) hP/hR/ hF (SD)	EF(15) hP/hR/ hF (SD)	EF(20) hP/hR/ hF (SD)	EW(5) hP/hR/ hF (SD)	EW(10) hP/hR/ hF (SD)	EW(15) hP/hR/ hF (SD)	EW(20) hP/hR/ hF (SD)	HCAIM hP/hR/ hF (SD)
Cellcycle	24,33/18,22 • /20,83 (1,39)	32,91/19,59 • /24,56 (2,69)	37,74/19,88 • /26,03 (2,21)	40,00/19,89 • /26,56 (2,13)	17,57/13,72 • /15,41 (1,75)	19,89/14,97 • /17,08 (1,37)	23,26/16,01 • /18,96 (1,63)	24,38/15,70 • /19,10 (2,03)	**55,14/22,40 /31,85 (1,69)**
Church	10,05/10,10 • /10,07 (1,25)	11,79/11,36 • /11,57 (1,31)	12,48/11,58 • /12,00 (1,63)	13,87/12,51 • /13,14 (1,45)	16,39/8,17 • /10,90 (1,01)	14,83/9,97 • /11,85 (1,51)	12,46/11,17 • /11,76 (1,41)	13,33/12,06 • /12,63 (1,58)	**20,06/17,41 /18,63 (1,13)**
Derisi	9,46/9,26 • /9,36 (1,00)	11,82/10,26 • /10,98 (1,20)	13,64/10,30 /11,73 (1,07)	14,23/9,68 /11,52 (1,07)	9,34/8,53 • /8,92 (0,75)	9,42/9,21 • /9,31 (1,28)	10,01/9,40 • /9,69 (1,30)	10,29/9,55 • /9,91 (1,14)	**18,15/9,45 /12,42 (0,82)**
Eisen	**26,84/19,47 /22,56 (1,33)**	29,95/18,06 /22,52 (2,10)	31,40/16,77 /21,86 (2,23)	32,85/16,30 /21,78 (1,70)	24,10/18,21 /20,74 (2,35)	26,56/19,03 /22,17 (1,42)	28,19/18,60 • /22,40 (1,27)	29,70/18,10 /22,49 (1,97)	37,68/14,83 /21,28 (1,43)
Expr	52,33/37,83 • /43,91 (1,39)	60,88/36,74 /45,82 (1,88)	64,36/35,40 /45,67 (2,35)	66,27/34,70 /45,54 (2,49)	30,49/23,94 • /26,82 (1,69)	35,48/25,42 • /29,62 (1,88)	40,70/27,19 • /32,60 (1,49)	44,34/28,19 • /34,46 (1,27)	**75,28/33,55 /46,41 (1,66)**
Gasch1	20,37/17,18 • /18,64 (1,97)	26,06/18,68 • /21,75 (2,33)	28,31/18,52 • /22,38 (1,51)	29,79/18,52 • /22,84 (1,75)	17,60/16,07 • /16,80 (1,47)	19,87/17,07 • /18,36 (1,61)	21,93/17,76 • /19,62 (2,27)	22,34/17,08 • /19,36 (2,06)	**46,01/18,96 /26,86 (1,16)**
Gasch2	18,34/14,96 • /16,48 (1,70)	21,20/15,03 • /17,59 (1,45)	24,87/15,86 • /19,37 (1,90)	26,30/15,74 • /19,69 (1,68)	16,50/13,34 • /14,75 (1,17)	17,68/14,79 • /16,10 (1,47)	17,81/14,16 • /15,77 (1,32)	19,25/14,60 • /16,61 (1,47)	**39,19/18,92 /25,51 (1,74)**
Seq	24,07/19,26 • /21,39 (0,87)	22,62/17,26 • /19,58 (1,03)	22,08/16,41 /18,83 (1,32)	22,34/16,16 /18,75 (1,15)	29,87/20,09 • /24,02 (1,67)	**31,78/20,49 • /24,91 (1,47)**	31,26/19,54 • /24,05 (1,11)	31,08/19,62 • /24,05 (1,26)	25,06/14,17 /18,10 (1,08)
SPO	14,84/12,59 /13,62 (1,41)	17,21/12,25 /14,31 (1,25)	**19,47/11,99 • /14,84 (1,37)**	19,79/11,20 /14,30 (0,78)	16,09/12,04 /13,77 (1,43)	14,56/12,02 /13,17 (1,19)	14,70/11,68 /13,02 (1,63)	16,01/11,86 /13,62 (0,85)	21,23/9,52 /13,14 (1,73)
EC-Interpro	96,51/87,83 • /91,96 (0,56)	97,18/87,59 • /92,14 (0,42)	96,97/88,91 /92,76 (0,53)	96,74/89,36 • /92,90 (0,31)	96,50/90,48 • /93,39 (0,61)	**96,39/91,08 • /93,66 (0,44)**	96,70/88,07 /92,18 (0,50)	96,57/88,64 /92,43 (0,67)	96,63/88,85 /92,58 (0,35)
EC-Pfam	96,38/84,23 /89,90 (0,56)	96,95/83,96 /89,98 (0,52)	96,43/83,33 • /89,40 (0,74)	96,58/83,16 • /89,37 (0,79)	96,92/82,52 • /89,13 (1,00)	96,55/82,44 • /88,93 (0,86)	**96,61/84,51 /90,16 (0,61)**	96,21/84,62 /90,04 (0,76)	96,53/84,54 /90,13 (0,61)
EC-Prints	95,96/85,84 • /90,62 (0,53)	**96,46/86,33 • /91,11 (0,67)**	96,35/85,83 • /90,78 (0,68)	96,07/85,26 • /90,34 (0,54)	96,42/86,12 • /90,98 (0,49)	96,06/85,59 • /90,52 (0,66)	96,28/84,16 /89,81 (0,46)	96,24/84,19 /89,81 (0,62)	95,85/84,45 /89,78 (0,70)
EC-Prosite	95,35/84,07 • /89,36 (0,54)	96,17/84,87 • /90,16 (0,84)	96,05/85,93 /90,71 (0,63)	96,18/85,57 • /90,56 (0,51)	96,29/84,28 • /89,88 (0,27)	95,43/85,43 • /90,15 (0,68)	95,54/84,46 • /89,66 (0,57)	95,50/84,38 • /89,59 (0,74)	**96,32/86,13 /90,94 (0,65)**
GPCR-Interpro	91,41/75,11 • /82,46 (0,69)	91,43/75,04 • /82,43 (0,73)	89,86/76,35 • /82,55 (0,94)	90,45/76,59 • /82,94 (0,64)	93,70/74,61 • /83,07 (0,71)	92,86/74,90 • /82,91 (0,65)	91,93/75,63 • /82,99 (0,72)	91,39/75,78 • /82,86 (0,73)	**91,89/76,90 /83,73 (0,84)**
GPCR-Pfam	75,63/46,32 • /57,45 (0,55)	71,16/49,66 • /58,49 (0,67)	74,94/51,06 • /60,72 (1,03)	70,32/52,38 • /60,03 (0,93)	91,82/31,78 • /47,21 (0,37)	87,08/32,94 • /47,80 (0,34)	86,10/34,89 • /49,65 (0,39)	86,30/33,94 • /48,70 (0,35)	**74,25/57,15 /64,58 (0,84)**
GPCR-Prints	91,82/71,13 • /80,16 (0,76)	90,70/71,41 • /79,90 (0,63)	88,50/73,21 • /80,13 (0,93)	89,31/73,02 • /80,35 (0,55)	93,78/71,18 • /80,93 (0,53)	93,66/71,45 • /81,06 (0,77)	92,00/72,06 • /80,82 (0,85)	90,14/71,90 • /79,99 (0,85)	**90,81/74,10 /81,61 (0,73)**
GPCR-Prosite	80,05/49,67 • /61,30 (0,48)	79,30/51,59 • /62,50 (0,45)	77,31/54,51 • /63,93 (0,73)	75,73/55,23 • /63,87 (0,80)	87,98/34,69 • /49,76 (0,30)	87,07/35,39 • /50,32 (0,40)	85,35/38,01 • /52,60 (0,37)	77,96/41,91 • /54,51 (0,53)	**76,37/59,07 /66,61 (1,18)**
# HCAIM Wins	12	11	9	10	12	12	11	11	

Table III. Mean values of hP, hR and hF obtained by GMNB after discretization of the datasets.

Base	EF hP/hR/ hF (SD)	EW hP/hR/ hF (SD)	HCAIM hP/hR/ hF (SD)
Cellcycle	50,89/19,69 • /28,39 (2,17)	37,86/15,41 • /21,90 (1,31)	**55,14/22,40 /31,85 (1,69)**
Church	17,80/15,14 • /16,35 (1,32)	16,95/14,82 • /15,79 (1,37)	**20,06/17,41 /18,63 (1,13)**
Derisi	21,38/8,38 /12,04 (0,76)	**19,10/9,96 /13,09 (0,74)**	18,15/9,45 /12,42 (0,82)
Eisen	36,32/13,84 • /20,04 (1,30)	31,47/13,23 • /18,63 (1,71)	**37,68/14,83 /21,28 (1,43)**
Expr	64,96/24,75 • /35,84 (2,12)	61,87/26,74 • /37,34 (1,54)	**75,28/33,55 /46,41 (1,66)**
Gasch1	40,34/15,22 • /22,10 (1,00)	32,74/15,50 • /21,04 (1,95)	**46,01/18,96 /26,86 (1,16)**
Gasch2	33,57/12,61 • /18,33 (0,80)	27,57/12,96 • /17,62 (1,60)	**39,19/18,92 /25,51 (1,74)**
Seq	25,76/13,42 /17,64 (1,48)	**26,16/15,57 • /19,51 (1,35)**	25,06/14,17 /18,10 (1,08)
SPO	24,15/8,84 /12,95 (0,97)	**21,80/10,21 /13,90 (1,32)**	21,23/9,52 /13,14 (1,73)
EC-Interpro	97,01/89,08 /92,87 (0,49)	**97,10/89,02 /92,88 (0,70)**	96,63/88,85 /92,58 (0,35)
EC-Pfam	**96,30/84,76 /90,16 (0,48)**	96,58/84,40 /90,08 (0,83)	96,53/84,54 /90,13 (0,61)
EC-Prints	95,91/84,68 /89,94 (0,60)	**96,25/85,46 • /90,53 (0,68)**	95,85/84,45 /89,78 (0,70)
EC-Prosite	96,30/86,43 /91,10 (0,47)	**96,34/86,04 /90,90 (0,75)**	96,32/86,13 /90,94 (0,65)
GPCR-Interpro	90,38/75,93 • /82,53 (0,99)	90,27/76,26 • /82,67 (0,56)	**91,89/76,90 /83,73 (0,84)**
GPCR-Pfam	**69,39/60,55 /64,67 (0,99)**	70,33/49,47 • /58,08 (0,69)	74,25/57,15 /64,58 (0,84)
GPCR-Prints	88,43/73,59 • /80,32 (0,99)	89,52/73,42 • /80,67 (0,65)	**90,81/74,10 /81,61 (0,73)**
GPCR-Prosite	**74,00/62,81 • /67,95 (0,66)**	75,05/52,18 • /61,55 (0,73)	76,37/59,07 /66,61 (1,18)
# HCAIM Wins	8	10	

statistical tests confirm the superiority of HCAIM in relation to the other discretization methods used in the experiments.

5. CONCLUSION

In spite of the importance of discretization methods to preprocess datasets used by classification techniques, to the best of our knowledge, there are no proposals in the literature of discretization methods tailored for hierarchical classification datasets that could be used in conjunction with global hierarchical classifiers.

Hierarchical classification is very important in bioinformatics, specially in gene and protein function prediction problems, where such functions are often hierarchically organized. Thus, previous work have already addressed these problems using hierarchical classification approaches. In addition, some of them have required a preprocessing stage in order to adjust the datasets to classification step. However, due to the lack of supervised discretization methods for the hierarchical context, works such as [Merschmann and Freitas 2013] and [Silla Jr and Freitas 2009] have employed unsupervised methods for data discretization.

Therefore, this work proposed a supervised discretization method for the hierarchical single label classification context applied to gene and protein function prediction, which are important real world problems in bioinformatics. The proposal presented here, called HCAIM, corresponds to an adaptation of the supervised discretization method named CAIM.

The computational experiments showed that the HCAIM method, for most of the datasets evaluated, allowed the hierarchical classifier GMNB to achieve predictive performance (hierarchical F-measure) superior to those reached when the datasets were preprocessed by the unsupervised methods Equal-Width and Equal-Frequency. This result clearly demonstrates the superiority of HCAIM over two unsupervised discretization methods and confirms the potential of applying the proposed method for performing the discretization of datasets used in hierarchical classification works.

As a future work we intend to extend and evaluate the method proposed here for the context of multi label hierarchical classification. In addition, evaluation of this proposal will be conducted using hierarchical datasets related to other domains and considering different global hierarchical classifiers.

REFERENCES

Clare, A. and King, R. D. Predicting gene function in saccharomyces cerevisiae. *Bioinformatics* 19 (suppl 2): ii42–ii49, 2003.

Costa, E. P., Lorena, A. C., Carvalho, A. C. P. L. F., Freitas, A. A., and Holden, N. Comparing several approaches for hierarchical classification of proteins with decision trees. In *Proceedings of the 2nd Brazilian Conference on Advances in Bioinformatics and Computational Biology, Lecture Notes in Bioinformatics 4643*. Angra dos Reis, Brazil, pp. 126–137, 2007.

D'Alessio, S., Murray, K., Schiaffino, R., and Kershenbaum, A. The effect of using hierarchical classifiers in text categorization. In *Content-Based Multimedia Information Access - Volume 1*. Paris, France, pp. 302–313, 2000.

Dougherty, J., Kohavi, R., Sahami, M., et al. Supervised and unsupervised discretization of continuous features. In *Machine Learning: Proceedings of the Twelfth International Conference*. Vol. 12. pp. 194–202, 1995.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. From data mining to knowledge discovery in databases. *AI Magazine* 17 (3): 37–54, 1996.

Freitas, A. and de Carvalho, A. C. VII. In D. Taniar (Ed.), *A Tutorial on Hierarchical Classification with Applications in Bioinformatics*. Vol. Research and Trends in Data Mining Technologies and Applications. Idea Group, pp. 182–196, 2007.

Galvão, L. and Merschmann, L. H. C. HSIM: A supervised imputation method for hierarchical classification scenario. In *Proceedings of the International Conference on Discovery Science*. Lecture Notes in Computer Science, vol. 9956. Bari, Italy, pp. 134–148, 2016.

Garcia, S., Luengo, J., Sáez, J. A., López, V., and Herrera, F. A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. *IEEE Transactions on Knowledge and Data Engineering* 25 (4): 734–750, 2013.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. The weka data mining software: An update. *Special Interest Group on Knowledge Discovery and Data Mining Explorations* 11 (1), 2009.

Holden, N. and Freitas, A. A. Improving the performance of hierarchical classification with swarm intelligence. In *Proceedings of the 6th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*. pp. 48–60, 2008.

Japkowicz, N. and Shah, M. *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, New York, NY, USA, 2011.

Kamal, N. A. M., Bakar, A. A., and Zainudin, S. Filter-wrapper approach to feature selection of gpcr protein. In *Proceedings of the International Conference on Electrical Engineering and Informatics*. Denpasar, Indonesia, pp. 693–698, 2015.

Kiritchenko, S., Matwin, S., and Famili, A. F. Functional annotation of genes using hierarchical text categorization. In *Proceedings of the ACL Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, 2005.

Koller, D. and Sahami, M. Hierarchically classifying documents using very few words. In *Proceedings of the Fourteenth International Conference on Machine Learning*. San Francisco, CA, USA, pp. 170–178, 1997.

Kurgan, L. A. and Cios, K. J. Caim discretization algorithm. *IEEE Transactions on Knowledge and Data Engineering* 16 (2): 145–153, 2004.

Labrou, Y. and Finin, T. Yahoo! as an ontology: using yahoo! categories to describe documents. In *Proceedings of the Eighth International Conference on Information and Knowledge Management*. Kansas City, USA, pp. 180–187, 1999.

Merschmann, L. H. C. and Freitas, A. A. An extended local hierarchical classifier for prediction of protein and gene functions. In *Proceedings of the 15th International Conference on Data Warehousing and Knowledge Discovery*. Prague, Czech Republic, pp. 159–171, 2013.

Naik, A. and Rangwala, H. Embedding feature selection for large-scale hierarchical classification. In *Proceedings of the IEEE International Conference on Big Data*. Washington DC, USA, pp. 1212–1221, 2016.

Paes, B. C., Plastino, A., and Freitas, A. A. Exploring attribute selection in hierarchical classification. *Journal of Information and Data Management* 5 (1): 124–133, 2014.

Qiu, X., Gao, W., and Huang, X. Hierarchical multi-class text categorization with global margin maximization. In *Proceedings of the ACL International Joint Conference on Natural Language Processing Conference Short Papers*. Association for Computational Linguistics, pp. 165–168, 2009.

Silla Jr, C. N. and Freitas, A. A. A global-model naive bayes approach to the hierarchical prediction of protein functions. In *Proceedings of the IEEE International Conference on Data Mining*. Miami, USA, pp. 992–997, 2009.

Silla Jr, C. N. and Freitas, A. A. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery* 22 (1-2): 31–72, 2011.