An Iterative Decision Tree Threshold Filter

Oscar Picchi Netto[1], **José Augusto Baranauskas**[1]

[1]Departamento de Computação e Matemática - Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto- Universidade de São Paulo (USP)

Abstract. *In this paper we propose and analyze a new filter for feature subset selection using an iterative decision tree threshold method. Using several biomedical or bioinformatics datasets, the filter has been evaluated on its data compression ability and AUC (Area Under Curve) performance within three scenarios. On average, the filter compressed almost 50% of the data. Additionally, AUC values using all versus selected filter features have not produced performance degradation in five different machine learning algorithms.*

Resumo. *Neste trabalho é proposto e analisado um novo filtro para seleção de atributos utilizando um método iterativo com árvores de decisão. Utilizando diversas bases biomédicas, o filtro foi avaliado em capacidade de compressão e valor AUC (*Area Under Curve*) para três cenários. Em média, o filtro foi capaz de compactar 50% dos dados. As análises dos valores AUC comparando todos os atributos contra aqueles atributos selecionados não produziu perda de desempenho significativa nos cinco algoritmos de aprendizado de máquina testados.*

1. Introduction

Data Mining (DM) is an interdisciplinary field bringing together techniques from Machine Learning, statistics, pattern recognition, databases and visualization to address the issue of extracting high-level knowledge from low-level data in large databases. When using Machine Learning (ML) techniques for DM, where the number of records (instances) is very large, usually several representative samples from the database are taken and presented to a ML algorithm. Afterwards the knowledge extracted from those samples by ML algorithms is combined in some way [Fayyad et al. 1996].

For instance, the exponential growth of the amount of biological data available raises two problems: on one hand, efficient information storage and management and, on the other hand, the extraction of useful information from this data [Larrañaga et al. 2006]. One important issue to be considered when using ML in DM is database records dimensionality reduction by reducing the number of records attributes (*i.e.* deleting columns in tables in the database literature or features/attributes in the Machine Learning literature). The data subset resulting from these deletions maintains the same number of instances but only a subset of features with predictive performance comparable to the full set of features. This is known as the Feature Subset Selection (FSS) problem, where one of the central issues is the selection of relevant features and/or the elimination of irrelevant ones.

Using ML and DM algorithms is one way to more effectively extract that information. But since the amount of data is huge, the use of an efficient FSS algorithm sometimes is essential, not only to speed up algorithms, but also to reduce data that can be benchmark

tested. For this reason FSS has passed from an illustrative example to real prerequisite for model building [Saeys et al. 2007]. Especially in microarray analysis or even in text mining there are huges amount of data and a FSS algorithm can help reducing that.

There are several reasons for doing Feature Subset Selection. One of them is that it, in general, improves accuracy since many ML algorithms degrade in performance when given too many features. Another reason is that FSS may improve comprehensibility, which is the ability for humans to understand the data and the classification rules induced by symbolic ML algorithms. Finally, FSS can reduce measurement cost since in some domains measuring features may be expensive. In this study we present an approach for feature subset selection using decision trees within a filter [Foithong et al. 2011].

This work is organized as follows: Section 2 presents the basic concepts of the feature subset selection problem; Section 3 has a brief description of the datasets used on this work; Section 4 describes the methodology proposed; Section 5 shows the experimental setup used to evaluate the proposed methodology; in Section 6 are shown the experiments and their results; Section 7 presents the conclusion of this study and describes how this study can be continued in the future.

2. Feature Subset Selection

Supervised learning is the process of automatically creating a classification model from a set of instances (records or examples) called the *training set* which belong to a set of classes. There are two aspects to be considered in this process: which features to use in describing the concept and how to combine those features. Once a model is created, it can be used to automatically predict the class of other unclassified records.

In other words, in supervised learning, an inducer is given a set of N training examples. Each example x is an element of the set $F_1 \times F_2 \times \ldots \times F_m$, where F_j is the domain of the jth feature. Training examples are tuples (x, y) where y is the label, output or class. The y values are typically drawn from a discrete set of classes $\{1, \ldots, K\}$ in the case of *classification* or from the real values in the case of *regression*. In this work we will refer to classification. Given a set of training examples, the learning algorithm (*inducer*) outputs a *classifier* such that, given a new instance, it accurately predicts the label y.

One of the central problems in supervised learning is the selection of useful features. Although most learning methods attempt to either select features or assign them degrees of importance, both theoretical analysis and experimental studies indicate that many algorithms scale poorly to domains with large numbers of irrelevant features. For example, the number of training cases needed for simple nearest neighbor to reach a given level of accuracy appears to grow exponentially with the number of irrelevant features, independent of the target concept. Even methods for inducing univariate decision trees, which explicitly select some attributes in favor of others, exhibit this behavior for some target concepts. Some techniques, like the Naive Bayes classifier, are robust with respect to irrelevant features but can be very sensitive to domains with correlated features, even if the features are relevant. This can be explained by the assumption of this sort of techniques related to independence among features. This suggests the need for additional methods to select a useful subset of features when many are available [Han et al. 2011].

Approaches for feature selection that have been developed can be grouped into three classes: those that *embed* the selection within the basic induction algorithm, those

that use feature selection to *filter* features during a pre-processing step ignoring the induction algorithm, and those that treat feature selection as a *wrapper* around the induction process, using the induction algorithm as a black-box [Blum and Langley 1997]. Another possible approach is to use a hybrid (filter and wrapper) method, trying to optimize the efficiency of feature selection [Min and Fangfang 2010, Lan et al. 2011, Estévez et al. 2009].

In the FSS filter model, which is of special interest within this work, the features are filtered independent of the induction algorithm. In this model the FSS is done as a preprocessing step, totally ignoring the effects of the selected features subset on the performance of the induction algorithm. For example, a simple decision tree algorithm can be used as a FSS filter to select features in large feature space for other inducers that take more time to search their solution space. The set of features selected by the tree are the output of the filter FSS process and the tree itself is discarded. The remaining unused features are then deleted from the training set, reducing its dimension, and this training set can be used by any other inducer. Still, features that are good for decision trees are not necessarily useful for other family of algorithms that may have an entirely different inductive bias.

The main disadvantage of the filter approach is that it totally ignores the effects of the selected feature subset on the performance of the induction algorithm. However, an interesting feature about filters is that once a dataset is filtered it can be used and evaluate by several inducers and/or paradigms. In the next section the filter approach proposed in this study is described.

3. Datasets

The experiments reported here used 30 datasets, all of them representing real medical data, such as gene expressions, surveys, diagnostics, etc. The medical domain often imposes difficult obstacles to learning algorithms: high dimensionality, huge or very small amounts of instances, several possible class values, unbalanced classes, etc. This sort of data are indicated for filters, due its large dimension, and the fact filters have a computational efficiency over wrappers [Kantardzic 2011]. Table 1 shows a summary of the datasets, none of which having missing values for the class attribute.

Since the number of attributes and instances on each dataset can influence results, we have used the density metric proposed by [**?**] partitioning datasets into 6 low-density (Density ≤ 1) and 24 high-density (Density > 1) datasets. The density is computed as Density $\triangleq \log_a n$, where n represents the number of instances, and a is the number of attributes.

Next we provide a brief description of each dataset. *Breast Cancer*, *Lung Cancer*, *CNS* (Central Nervous System Tumour Outcome), *Colon*, *Lymphoma*, *Leukemia*, *Leukemia nom.*, *WBC* (Wisconsin Breast Cancer), *WDBC* (Wisconsin Diagnostic Breast Cancer), *Lymphography* and *H. Survival* (*H.* stands for *Haberman's*) are all related to cancer and their attributes consist of clinical, laboratory and gene expression data. *Leukemia* and *Leukemia nom.* represent the same data, but the second one had its attributes discretized [Netto et al. 2010]. *C. Arrhythmia* (*C.* stands for *Cardiac*), *Heart Statlog*, *HD Cleveland*, *HD Hungarian* and *HD Switz.* (*Switz.* stands for *Switzerland*) are related to heart diseases and their attributes represent clinical and laboratory data. *Allhyper*, *Allhypo*, *ANN Thyroid*, *Hypothyroid*, *Sick* and *Thyroid 0387* are a series of datasets related

Table 1. Summary of the datasets used in the experiments. ATTR, $a_{\#}$ and a_a stand for the total number of attributes and for the number of numerical and nominal attributes, respectively; MISS represents the percentage of attributes with missing values, not considering the class attribute. Datasets are in ascending order of density.

#	Dataset	N	c	ATTR	MISS	Density
1	CNS	60	2	7129	0.00%	0.46
2	Leukemia	72	2	7129	0.00%	0.48
3	Leukemia nom.	72	2	7129	0.00%	0.48
4	Colon	62	2	2000	0.00%	0.54
5	Lymphoma	96	9	4026	5.09%	0.55
6	Lung Cancer	32	3	56	0.28%	0.86
7	C. Arrhythmia	452	16	279	0.32%	1.09
8	Hepatitis	155	2	19	5.67%	1.71
9	WDBC	569	2	30	0.00%	1.87
10	Dermatology	366	6	34	0.06%	1.67
11	Lymphography	148	4	18	0.00%	1.73
12	Splice Junction	3190	3	60	0.00%	1.97
13	Heart Statlog	270	2	13	0.00%	2.18
14	HD Switz.	123	5	13	17.07%	1.88
15	Sick	3772	2	29	5.54%	2.45
16	P. Patient	90	3	8	0.42%	2.16
17	Hypothyroid	3163	2	25	6.74%	2.50
18	HD Hungarian	294	5	13	20.46%	2.22
19	HD Cleveland	303	5	13	0.18%	2.23
20	Allhypo	3772	4	29	5.54%	2.45
21	Breast Cancer	286	2	9	0.35%	2.57
22	Allhyper	3772	5	29	5.54%	2.41
23	ANN Thyroid	7200	3	21	0.00%	2.92
24	WBC	699	2	9	0.25%	2.98
25	Pima Diabetes	768	2	8	0.00%	3.19
26	Liver Disorders	345	2	6	0.00%	3.26
27	Thyroid 0387	9172	32	29	5.50%	2.71
28	C. Method	1473	3	9	0.00%	3.32
29	Ecoli	482	13	280	1.07%	2.99
30	H. Survival	306	2	3	0.00%	5.21

to thyroid conditions. *Hepatitis* and *Liver Disorders* are related to liver diseases, whereas *C. Method* (*C.* stands for *Contraceptive*), *Dermatology*, *Pima Diabetes* (Pima Indians Diabetes) and *P. Patient* (*P.* stands for *Postoperative*) are other datasets related to human conditions. *Splice Junction* is related to the task of predicting boundaries between exons and introns. *E.Coli* is related to protein localization sites. Datasets were obtained from the UCI Repository [Frank and Asuncion 2010], *Leukemia* and *Leukemia nom.* were obtained from [Institute 2010].

4. An Iterative Decision Tree Threshold Filter

Our approach is based in a previous experiment [Netto et al. 2010] with a high dimensional gene expression dataset. In that study, ten decision trees were grown iteratively, each time removing from the training set attributes appearing on the previous tree. It was possible to realize a better result for intermediate trees than the first ones. The best tree found also outperformed several approaches using only 1-2 attributes (genes).

In general, filter algorithms evaluate each attribute individually for some degree of relevance related to the target concept class. Sometimes two or more attributes can be considered at a time but at a high computational cost [Gao et al. 2010]. Our approach differs from those in the sense a decision tree may be able to capture relationships among several attributes w.r.t. class at a time. Besides that, inducing a decision tree is fast, which allows performing this process on high dimensional datasets commonly found in gene expression profiles or massive medical databases.

Therefore, the filter approach proposed in this study can be seen as an upgrading of [Netto et al. 2010]. It iteratively builds a decision tree, selects attributes appearing on

that tree (based on a threshold from the first tree performance), and removes them from the training set. These steps are repeated until there is no more attributes left on the training set or the decision tree induced is a leaf (which means no attributes were found to be able to separate class concepts). At the end, the filter outputs the selected attributes.

Algorithm 1 shows the high level code of the attribute selection approach proposed here, where N represents the number of instances in the training set, x_i and y_i, $i = 1, \ldots, N$ represent a vector containing the attribute values and the class label for instance i, respectively.

First, a bootstrap sample from all instances is taken, creating the training set (Line 2). Instances that do not appear in the training set (Bag) are set apart as the test set, also known as the out-of-bag set (Line 3). The first decision tree is induced (Line 5) and its AUC value is used. In fact, the first AUC value and Θ, an algorithm parameter, are combined to define the threshold θ (Line 6). Next, attributes are selected in the following way. At every iteration, the AUC obtained by the decision tree is evaluated by the threshold θ, which selects or not attributes appearing on that tree. All attributes on tree are now removed from the training (Line 12) and test sets (Line 13), and a new tree is grown (Line 14). This process is repeated until a leaf is induced (Line 15) or all attributes have been used. Finally, all the selected attributes are returned (Line 16).

The threshold θ is as a percentage Θ of the AUC from the first tree. If $\Theta = 0$ all attributes, despite de AUC value, will be selected.

Algorithm 1 An Iterative Decision Tree Threshold Filter — IDTTF

Require: Instances: a set of N labelled instances $\{(x_i, y_i), i = 1, 2, \ldots, N\}$
Θ: a parameter for selecting attributes, where $0 < \Theta \leq 1$
Ensure: Selected : a subset of attributes
1: **procedure** idttf(Instances, Θ)
2: Bag $\leftarrow$ BootstrapSample(Instances)
3: OutOfBag $\leftarrow$ Instances $\setminus$ Bag
4: Selected $\leftarrow \emptyset$
5: $C \leftarrow$ build_decision_tree(Bag)
6: $\theta \leftarrow \Theta \times$AUC($C$,OutOfBag)
7: **repeat**
8: AttrOnClassifier $\leftarrow$ all attributes appearing on C
9: **if** AUC(C,OutOfBag) $\geq \theta$ **then**
10: Selected $\leftarrow$ Selected $\cup$ AttrOnClassifier
11: **end if**
12: Bag $\leftarrow$ Bag $\setminus$ AttrOnClassifier
13: OutOfBag $\leftarrow$ OutOfBag $\setminus$ AttrOnClassifier
14: $C \leftarrow$ build_decision_tree(Bag)
15: **until** C is a leaf **or** there is no attributes left in Bag
16: **return** Selected

Table 2 shows a running example of Algorithm 1 using $\Theta = 100\%$. Consider a dataset containing ten attributes $a_1, a_2, \ldots, a_{10}$ and a class attribute c. Assume a decision tree is induced containing the attributes a_1, a_5 and a_9 and AUC $= 90\%$. All trees now induced with an AUC larger or equal than $\theta = 90\%$ will have attributes selected by the algorithm. The first iteration starts by analyzing the tree already built, since it has an AUC $= 90\%$ it will have its attributes selected. Still in the first iteration (as well as on the subsequent iterations), the attributes that appeared on the first tree are removed and the

Table 2. A running example of Algorithm 1 for $\Theta = 100\%$

Iteration	Tree	Attributes on Tree	AUC	θ	Selected
	T_1	$\{a_1, a_5, a_9\}$	90%	90%	
I_1	T_2	$\{a_4, a_2, a_{10}, a_8\}$	75%	90%	$\{a_1, a_5, a_9\}$
I_2	T_3	$\{a_6, a_7, a_3\}$	95%	90%	$\{a_1, a_5, a_9\}$
I_3	T_4	$\emptyset$	End		$\{a_1, a_5, a_9, a_6, a_7, a_3\}$

second tree is grown, and assume now this second tree contains attributes a_4, a_2, a_{10} and a_8.

The second iteration begins analyzing the second tree, which has an AUC $= 75\%$, which is lower then $\theta = 90\%$. Therefore, attributes appearing on this second tree will not be selected by our filter. However, as before, these attributes are removed from the dataset. The third tree is then induced and assumes this time the attributes a_6, a_7 and a_3 are within this tree. The third iteration starts and tests if the tree has an AUC larger then $\theta = 90\%$. Assuming the third tree has an AUC $= 95\%$ the attributes will be selected and removed from the dataset.At the end of the third iteration the fourth tree is induced, but all attributes were already removed from the dataset; for that reason the tree built is a leaf and the stop criterion is achieved. The attributes selected a_1, a_5, a_9, a_6, a_7 and a_3 are now returned, in this order, as the filter output.

5. Experimental Methodology

In this section an evaluation of the proposed filter is performed. We have used different machine learning paradigms and several datasets. The paradigms used were rules represented by the PART algorithm, decision-tree represented by J48, statistic learning using Naive Bayes (NB), support vector machines with Sequential Minimal Optimization (SMO) and lazy learning using the IBk algorithm. All of them have been used in their default settings, except IBk where $k = 3$, referred from now on as IBk-3. We have used the Weka[1] [Witten and Frank 2005] plataform for running the experiments. The proposed filter was implemented as a novel Weka class.

As shown in Algorithm 1, the threshold is set relatively to the first tree AUC. Three settings were used in the experiments, changing the value of Θ, 100%, 95% and 75% referred from now on as m100, m095 and m075, respectively.We have evaluated two filter aspects in the experiments: the compression capacity and the AUC values (area under ROC curve).The compression capacity can be defined as how compact can a dataset be by the filter, it means how many attributes the filter can remove from the original dataset, hopefully without taking away significant information. For instance a dataset with 1000 and 75% compression capacity means only 250 attributes were selected by the filter.

The baseline for comparisons is the AUC value obtained by the classifier induced with all attributes through ten fold cross-validation. For the filter, ten fold cross-validation has been used also, but each test fold was never seen by the filter. In other words, the filter only sees 9 folds as the full training set and finds an attribute subset. This subset is

[1] `www.cs.waikato.ac.nz/~ml/weka`

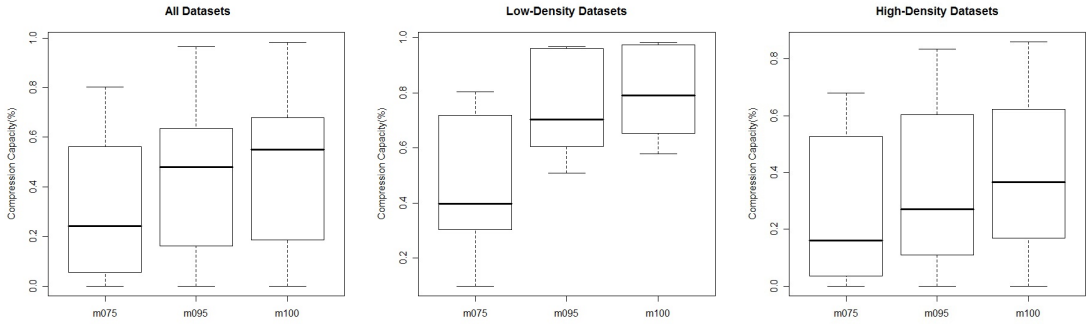

Figure 1. IDTTF Compression for each Setting

used to select attributes from both the 9 training folds as well as the remaining test fold. The 9 filtered training folds are then fed to one of the inducers mentioned earlier and its accuracy is evaluated on the filtered test fold. This process was repeated ten times and results averaged.

6. Results

Figure 1 shows compression capacity for all (top), 6 low-density (center) and 24 high-density (bottom) datasets and the average values in the experiments. It is possible to observe the m075 had the worst compression capacity. This can be explained by the fact that since it is a lower threshold more attributes will be selected. For high-density datasets the mean compression capacity of the filter is less than 40% for all three settings. Considering the low-density datasets the first two settings, m100 and m095 were able to compress the data 75% while m075 could not reach 50%, on average.

AUC values for all dataset and filter settings are shown on Table 3. To analyze the performance of the filter on each dataset was used the Friedman test [Friedman 1940] considering a significance level of 5%; the null hypothesis $H0$ assumes that all classifiers have equal performance. If the null hypothesis is rejected, a Benjamini-Hochberg *post-hoc* test [Benjamini and Hochberg 1995] is used to detect any significant difference among classifiers.

Analyzing the Average Rank on Table 3 its possible to observe m075 setting, for all paradigms and all datasets, had the worst performance. Only SMO had a better performance than all of the filter settings. On the low-density datasets J48 also had a better performance than all the three settings. On the high-density datasets with all inducers, except SMO, the m100 and m095 settings were the best, except J48 where m095 had a slightly better performance than m100. Since any of them had the same performance, the null hypothesis is rejected and a *post-hoc* test were made.

The result of *post-hoc* test is shown in Table 4 considering the three settings. On this table the symbol $\triangle$ ($\blacktriangle$) means that the classifier on the row using all attributes is (significantly) better than the same classifier using attributes selected by the filter setting at the column; the symbol $\triangledown$ ($\blacktriangledown$) means that classifier on the row using all attributes is (significantly) worst than the same classifier using attributes selected by the filter setting at the column.

Table 3. AUC values, mean, median and average rank obtained in the experiments.

		J48				IBk-3				NB				PART				SMO		
Datasets	J48	m100	m095	m075	IBk-3	m100	m095	m075	NB	m100	m095	m075	PART	m100	m095	m075	SMO	m100	m095	m075
CNS	0.50	0.49	0.49	0.46	0.56	0.56	0.56	0.53	0.58	0.59	0.59	0.53	0.00	0.59	0.59	0.53	0.63	0.63	0.63	0.56
Leukemia	0.75	0.76	0.76	0.70	0.85	0.88	0.88	0.78	0.98	1.00	1.00	0.90	0.75	0.76	0.76	0.70	0.98	0.98	0.98	0.90
Leukemia nom.	0.92	0.92	0.92	0.83	0.99	0.99	0.99	0.89	0.96	0.99	0.99	0.90	0.88	0.86	0.86	0.78	0.97	0.97	0.97	0.89
Colon	0.80	0.73	0.81	0.73	0.83	0.87	0.87	0.77	0.61	0.63	0.63	0.54	0.79	0.83	0.83	0.75	0.84	0.82	0.82	0.74
Lymphoma	0.90	0.90	0.90	0.80	0.92	0.92	0.92	0.83	0.76	0.77	0.77	0.68	0.81	0.82	0.82	0.73	0.98	0.98	0.98	0.88
Lung Cancer	0.68	0.63	0.64	0.63	0.68	0.78	0.78	0.68	0.71	0.71	0.71	0.61	0.69	0.69	0.69	0.61	0.63	0.63	0.63	0.55
C. Arrhythmia	0.72	0.74	0.73	0.65	0.66	0.67	0.67	0.59	0.81	0.81	0.81	0.72	0.74	0.75	0.75	0.67	0.79	0.79	0.79	0.70
Hepatitis	0.70	0.71	0.72	0.63	0.79	0.76	0.76	0.68	0.86	0.86	0.86	0.79	0.83	0.80	0.80	0.71	0.75	0.70	0.70	0.62
WDBC	0.93	0.94	0.94	0.84	0.98	0.98	0.98	0.88	0.98	0.98	0.98	0.88	0.94	0.94	0.94	0.84	0.97	0.97	0.97	0.88
Dermatology	0.97	0.98	0.98	0.88	0.99	0.99	0.99	0.89	1.00	1.00	1.00	0.90	0.98	0.98	0.98	0.88	0.98	0.98	0.98	0.89
Lymphography	0.79	0.79	0.79	0.72	0.89	0.88	0.88	0.80	0.91	0.91	0.91	0.82	0.79	0.79	0.79	0.73	0.87	0.85	0.85	0.78
Splice Junction	0.96	0.96	0.96	0.87	0.94	0.95	0.95	0.85	0.99	0.99	0.99	0.89	0.96	0.96	0.96	0.87	0.96	0.96	0.96	0.87
Heart Statlog	0.76	0.74	0.75	0.69	0.83	0.83	0.83	0.75	0.90	0.90	0.90	0.82	0.74	0.75	0.75	0.66	0.84	0.84	0.84	0.75
HD Switz.	0.55	0.55	0.55	0.49	0.49	0.54	0.54	0.49	0.53	0.54	0.54	0.47	0.54	0.56	0.56	0.50	0.57	0.56	0.56	0.50
Sick	0.95	0.95	0.95	0.86	0.88	0.92	0.92	0.82	0.93	0.93	0.93	0.83	0.96	0.97	0.97	0.87	0.50	0.50	0.50	0.45
P. Patient	0.49	0.49	0.49	0.44	0.31	0.50	0.50	0.44	0.39	0.46	0.46	0.43	0.46	0.46	0.46	0.41	0.47	0.49	0.49	0.44
Hypothyroid	0.95	0.95	0.95	0.86	0.85	0.87	0.87	0.79	0.97	0.98	0.98	0.89	0.96	0.95	0.95	0.85	0.75	0.74	0.74	0.67
HD Hungarian	0.77	0.73	0.73	0.66	0.87	0.86	0.86	0.77	0.90	0.89	0.89	0.80	0.87	0.84	0.84	0.75	0.80	0.78	0.78	0.70
HD Cleveland	0.80	0.82	0.80	0.73	0.85	0.84	0.84	0.77	0.90	0.90	0.90	0.82	0.81	0.84	0.84	0.76	0.84	0.84	0.84	0.75
Allhypo	1.00	1.00	1.00	0.90	0.74	0.76	0.76	0.69	0.93	0.94	0.94	0.84	1.00	1.00	1.00	0.90	0.59	0.59	0.59	0.54
Breast Cancer	0.63	0.62	0.63	0.56	0.66	0.64	0.64	0.58	0.72	0.71	0.71	0.63	0.58	0.58	0.58	0.51	0.59	0.57	0.57	0.51
Allhyper	0.90	0.89	0.89	0.81	0.85	0.90	0.90	0.81	0.97	0.97	0.97	0.88	0.94	0.95	0.95	0.87	0.66	0.63	0.63	0.57
ANN Thyroid	0.99	0.99	0.99	0.89	0.74	0.78	0.78	0.70	0.93	0.93	0.93	0.84	0.99	1.00	1.00	0.90	0.59	0.59	0.59	0.53
WBC	0.95	0.96	0.95	0.86	0.99	0.99	0.99	0.89	0.99	0.99	0.99	0.89	0.97	0.97	0.97	0.88	0.97	0.97	0.97	0.87
Pima Diabetes	0.75	0.75	0.75	0.67	0.74	0.74	0.74	0.67	0.82	0.82	0.82	0.73	0.79	0.79	0.79	0.71	0.72	0.72	0.72	0.65
Liver Disorders	0.67	0.67	0.67	0.60	0.64	0.64	0.64	0.58	0.65	0.65	0.65	0.58	0.67	0.67	0.67	0.60	0.50	0.50	0.50	0.45
Thyroid 0387	0.98	0.98	0.98	0.89	0.84	0.85	0.85	0.76	0.93	0.93	0.93	0.83	0.98	0.98	0.98	0.88	0.98	0.98	0.98	0.88
C. Method	0.66	0.66	0.66	0.60	0.62	0.62	0.62	0.56	0.69	0.69	0.69	0.63	0.64	0.65	0.65	0.59	0.63	0.63	0.63	0.57
Ecoli	0.92	0.93	0.93	0.83	0.94	0.94	0.94	0.84	0.97	0.97	0.97	0.87	0.92	0.92	0.92	0.83	0.95	0.95	0.95	0.85
H. Survival	0.58	0.58	0.58	0.51	0.63	0.63	0.63	0.58	0.67	0.67	0.67	0.60	0.55	0.55	0.55	0.49	0.51	0.50	0.50	0.45
All																				
Mean	0.80	0.79	0.80	0.72	0.78	0.80	0.80	0.72	0.83	0.84	0.84	0.75	0.78	0.81	0.81	0.73	0.76	0.76	0.76	0.68
Median	0.79	0.78	0.79	0.72	0.84	0.84	0.84	0.76	0.90	0.90	0.90	0.82	0.81	0.82	0.82	0.74	0.77	0.76	0.76	0.68
Average Rank	2.13	2.07	1.85	3.95	2.33	1.88	1.88	3.90	2.40	1.82	1.82	3.97	2.43	1.80	1.80	3.97	1.73	2.13	2.13	4.00
Low-density																				
Mean	0.76	0.74	0.75	0.69	0.81	0.83	0.83	0.75	0.77	0.78	0.78	0.69	0.65	0.76	0.76	0.68	0.84	0.84	0.84	0.76
Median	0.77	0.75	0.79	0.71	0.84	0.87	0.87	0.77	0.74	0.74	0.74	0.65	0.77	0.79	0.79	0.72	0.91	0.89	0.89	0.81
Average Rank	1.66	2.66	1.91	3.75	2.83	1.67	1.67	3.83	2.83	1.58	1.58	4.00	2.67	1.75	1.75	3.83	1.83	2.08	2.08	4.00
High-density																				
Mean	0.81	0.81	0.81	0.73	0.78	0.80	0.80	0.72	0.85	0.85	0.85	0.77	0.82	0.82	0.82	0.74	0.74	0.74	0.74	0.66
Median	0.79	0.80	0.79	0.72	0.84	0.83	0.83	0.75	0.91	0.91	0.91	0.82	0.85	0.84	0.84	0.76	0.75	0.73	0.73	0.66
Average Rank	2.25	1.92	1.83	4.00	2.21	1.94	1.94	3.92	2.29	1.88	1.88	3.96	2.38	1.81	1.81	4.00	1.71	2.15	2.15	4.00

Observing Table 4 it is possible to confirm the results already mentioned about Table 3. The m075 setting is always worst than the original dataset; for SMO and J48 it is always significantly worst. The SMO is always better than the filter proposed for all datasets. On IBk-3, PART and NB the filter obtained a better result, but not significant, for all, low and high-density datasets. On the low-density datasets J48 obtained a better result than all three settings.

Table 4. Benjamini-Hochberg *post-hoc* Test - IDTTF+inducer versus inducer

	M100 (ALL/HIGH/LOW)	M095 (ALL/HIGH/LOW)	M075 (ALL/HIGH/LOW)
J48	▽/▽/△	▽/▽/△	▲/▲/▲
IBK-3	▽/▽/▽	▽/▽/▽	▲/▲/△
NB	▽/▽/▽	▽/▽/▽	▲/▲/△
PART	▽/▽/▽	▽/▽/▽	▲/▲/△
SMO	△/△/△	△/△/△	▲/▲/▲

7. Conclusion

In this paper we proposed a iterative decision tree threshold filter for feature subset selection. Although the proposed filter can use any classifier with embedded feature selection and any metric to determinate if an attribute should be selected, we have fixed the the classifier used here as J48 and AUC as the metric. Using several biomedical datasets we have evaluated our filter on compression ability and performance in five machine learning paradigms.

The data compression on three methodology settings showed a high threshold, such as m100 and m095 can compress almost 50% of the data, but a lower threshold, m075, only can compress around 25%. Analyzing low-density datasets these percentages changed to over 75% for higher thresholds and 45% for lower thresholds. A statistical test showed the first two settings (m100 and m095) had a performance slightly better, but not significantly, against the full dataset. For threshold m075, results were always worst, sometimes significantly, than the full dataset. For five paradigms evaluated on this study, the filter proposed had achieved a better rank performance in four of them.

Future work can be done to improve the filter proposed here. We are working on a stop criteria, trying to reduce the time the filter spends trying to get the final subset. Other possible upgrade would be allowing to change the inducer inside the filter, such as induction of rules, as well as to compare our approach with others FSS algorithms. If the classifier used on the methodology generates results where some attributes can have more information than others, such as decision trees, another improvement can be made using the level that the attribute appears on the tree to give it a better score. Other settings can be implemented to try to improve the methodology, such as changing AUC to other metrics.

Acknowledgments.

This work was partially funded by a joint grant between the National Research Council of Brazil (CNPq), and the Amazon State Research Foundation (FAPEAM) through the Program National Institutes of Science and Technology, INCT ADAPTA Project (Centre for Studies of Adaptations of Aquatic Biota of the Amazon).

References

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society Series B*, 57:289–300.

Blum, A. L. and Langley, P. (1997). Selection of relevant features and examples in machine learning. *AI*, 97(1–2):245–271.

Estévez, P., Tesmer, M., Perez, C., and Zurada, J. (2009). Normalized mutual information feature selection. *Neural Networks, IEEE Transactions on*, 20(2):189–201.

Fayyad, U. M., Piatetsky-Shapiro, G., and Smyth, P. (1996). *From Data Mining to Knowledge Discovery: An Overview*, pages 1–30.

Foithong, S., Pinngern, O., and Attachoo, B. (2011). Feature subset selection wrapper based on mutual information and rough sets. *Expert Systems with Applications*.

Frank, A. and Asuncion, A. (2010). Uci machine learning repository.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):86–92.

Gao, K., Khoshgoftaar, T., and Van Hulse, J. (2010). An evaluation of sampling on filter-based feature selection methods. In *Proceedings of the 23rd International Florida Artificial Intelligence Research Society Conference*, pages 416–421.

Han, J., Kamber, M., and Pei, J. (2011). *Data mining: concepts and techniques*. Morgan Kaufmann.

Institute, B. (2010). Cancer program data sets.

Kantardzic, M. (2011). *Data mining: concepts, models, methods, and algorithms*. Wiley-IEEE Press.

Lan, Y., Ren, H., Zhang, Y., Yu, H., and Zhao, X. (2011). A hybrid feature selection method using both filter and wrapper in mammography cad. In *Image Analysis and Signal Processing (IASP), 2011 International Conference on*, pages 378–382. IEEE.

Larrañaga, P., Calvo, B., Santana, R., Bielza, C., Galdiano, J., Inza, I., Lozano, J., Armañanzas, R., Santafé, G., Pérez, A., et al. (2006). Machine learning in bioinformatics. *Briefings in bioinformatics*, 7(1):86–112.

Min, H. and Fangfang, W. (2010). Filter-wrapper hybrid method on feature selection. In *Intelligent Systems (GCIS), 2010 Second WRI Global Congress on*, volume 3, pages 98–101. IEEE.

Netto, O., Nozawa, S., Mitrowsky, R., Macedo, A., Baranauskas, J., and Lins, C. (2010). Applying decision trees to gene expression data from dna microarrays: A leukemia

case study. In *XXX Congress of the Brazilian Computer Society, X Workshop on Medical Informatics*, page 10.

Oshiro, T. M., Perez, P. S., and Baranauskas, J. A. (2012). How many trees in a random forest? *In Proceedings of the 8th International Conference on Machine Learning and Data Mining*. Submitted.

Saeys, Y., Inza, I., and Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition*. Morgan Kaufmann.