Knowledge-Based Systems 25 (2012) 35–44

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Fast wrapper feature subset selection in high-dimensional datasets by means of filter re-ranking

Pablo Bermejo *, Luis de la Ossa, José A. Gámez, José M. Puerta

Department of Computing Systems, Intelligent Systems and Data Mining Laboratory (I3A), University of Castilla-La Mancha, Albacete 02071, Spain

ARTICLE INFO

Article history:
Available online 5 February 2011

Keywords:
Feature subset selection
High-dimensional datasets
Wrapper algorithms
Filter measures
Complexity
Rank-based algorithms
Re-ranking

ABSTRACT

This paper deals with the problem of supervised wrapper-based feature subset selection in datasets with a very large number of attributes. Recently the literature has contained numerous references to the use of hybrid selection algorithms: based on a filter ranking, they perform an incremental wrapper selection over that ranking. Though working fine, these methods still have their problems: (1) depending on the complexity of the wrapper search method, the number of wrapper evaluations can still be too large; and (2) they rely on a univariate ranking that does not take into account interaction between the variables already included in the selected subset and the remaining ones.

Here we propose a new approach whose main goal is to drastically reduce the number of wrapper evaluations while maintaining good performance (e.g. accuracy and size of the obtained subset). To do this we propose an algorithm that iteratively alternates between filter ranking construction and wrapper feature subset selection (FSS). Thus, the FSS only uses the first block of ranked attributes and the ranking method uses the current selected subset in order to build a new ranking where this knowledge is considered. The algorithm terminates when no new attribute is selected in the last call to the FSS algorithm. The main advantage of this approach is that only a few *blocks* of variables are analyzed, and so the number of wrapper evaluations decreases drastically.

The proposed method is tested over eleven high-dimensional datasets (2400–46,000 variables) using different classifiers. The results show an impressive reduction in the number of wrapper evaluations without degrading the quality of the obtained subset.

1. Introduction

Feature (or variable, or attribute) subset selection (FSS) is the process of identifying the input variables which are relevant to a particular learning (or data mining) problem [1,2], and is a key process in supervised classification. FSS helps to improve classification performance (accuracy, AUC, etc.) and also to obtain more interpretable classifiers or to detect outliers [3]. In the case of high-dimensional datasets, e.g., datasets with thousands of variables, FSS is even more important because otherwise the number of instances needed to obtain reliable models will be enormous (impracticable for many real applications such as microarray domains).

Most algorithms for supervised FSS can be classified as *filter* or *wrapper* approaches. In the filter approach an attribute (or attribute subset) is evaluated by only using intrinsic properties of the data (e.g. statistical or information-based measures). Filter techniques have the advantage of being fast and general, in the sense that the subset obtained is not biased in favor of a specific classifier. On the other hand wrapper algorithms are those that use a classifier (usually the one to be used later) in order to assess the quality of a given attribute subset [4]. Wrapper algorithms have the advantage of achieving greater accuracy than filters but with the disadvantage of being (far) more time-consuming and obtaining an attribute subset that is biased towards the classifier used. Over the last decade wrapper-based FSS has been an active area of research. Different search algorithms (greedy sequential [5], floating [6], best-first search, branch and bound [7], evolutionary algorithms [8–10], etc.) have been used to guide the search process while some classifier (e.g. Naive Bayes, KNN, etc.) is used as a surrogate in order to evaluate the goodness of the subset proposed by the search algorithm. There is no doubt that the results provided by wrapper methods are better than those obtained by using filter algorithms, but the main problem is that they do not scale well. Thus, while datasets of up to 100 or 500 variables were the norm in the last decade of the 20th century, at the start of the 21st century new datasets which involve thousands of variables appeared (e.g. genetics or information-retrieval-based datasets), and the result is that the use of *pure* wrapper algorithms is intractable in

* Corresponding author.
E-mail address: pablo.bermejo@uclm.es (P. Bermejo).

0950-7051/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2011.01.015

many cases [11]. Because of this, hybrid filter-wrapper algorithms have become the focus of attention in the last few years. The idea is to use a filter algorithm whose output guides the wrapper algorithm. In this way the advantages of the wrapper approach are retained whilst the number of wrapper evaluations is (considerably) reduced. Examples of these algorithms are [11,12], which incrementally explore the attributes by following the ranking obtained by a filter measure; [13], which applies a wrapper sequential forward search but only over the first k (e.g. 100) attributes in the filter ranking; and [14,15], which use the filter-based ranking for a better organization of the search process.

Our idea in this paper is to improve the efficiency of these so-called hybrid filter-wrapper FSS algorithms. To do this, our aim is to *drastically* reduce the number of wrapper evaluations by increasing the number of the filter evaluations carried out. Our proposal is based on working incrementally not only at the attribute level, but also at the *block* or *set* of attributes level, taking into account the selected subset ($\mathcal{S}$) in the previous blocks. Thus, we start by using a filter measure to rank the attributes, then an incremental filter-wrapper algorithm $\mathcal{A}$ is applied but only over the first *block*, that is, over the first B ranked attributes. Let $\mathcal{S}$ be the subset of attributes selected from this first block. Then, a new ranking is computed over the remaining attributes but taking into account the already selected ones ($\mathcal{S}$). Then, algorithm $\mathcal{A}$ is run again over the first block in this new ranking but initializing the selected subset to $\mathcal{S}$ instead of $\emptyset$. This process is iterated until no modification in the selected subset is obtained. As we show, in our experiments, the number of *re-ranks* carried out is very small, and so only a small percentage of attributes is explored, which leads to a great reduction in wrapper evaluations (and so in CPU time) but without decreasing the accuracy of the output obtained. Even the size of the selected subset is reduced.

Besides this introduction, this paper is organized as follows: the following section presents a set of well-known incremental selection algorithms; in Section 3 we introduce the motivation for reconstructing the ranking in search-time; in Section 4 we introduce our contribution/improvement based on re-ranking; then, Section 5 contains the experimental evaluation carried out, and finally in Section 6 we provide a summary of the main conclusions and future work.

2. Previous work on wrapper FSS for high-dimensional data

In this section we briefly review previous works in the literature for speeding-up wrapper subset selection when working with high-dimensional datasets. We focus on methods based on the use of a filter ranking, so we start with the construction of the ranking and then we briefly describe some algorithms that make use of it.

2.1. Filter step: creating the ranking

As we are in a supervised problem, in order to create the ranking a measure $m(A_i;C)$ is computed for each predictive attribute A_i with respect to the class feature C. Therefore, this stage requires $\mathcal{O}(n)$ filter evaluations.[1] It is very common to use correlation and information-based metrics as filter measure $m(A_i;C)$. In our case, we follow [11,12,14,16] and symmetrical uncertainty (SU) [17] is used to evaluate the *individual* merit with respect to the class for each attribute. SU is a nonlinear information-theory-based measure that can be interpreted as a sort of mutual information normalized to interval $[0,1]$:

$$SU(A_i,C) = 2\left(\frac{H(C) - H(C|A_i)}{H(C) + H(A_i)}\right),$$

C being the class and $H()$ being the Shannon entropy. Attributes are ranked in decreasing SU order; that is, more informative attributes are placed first.

2.2. Rank-based FSS algorithms

In the following algorithms we assume that the ranking r has already been computed.

2.2.1. Rank search

The *Rank Search* algorithm [18] evaluates exactly n subsets, containing the first ranked variable, the first two ranked variables, the first three ranked variables,... Therefore, it is linear in the number of wrapper evaluations, that is, $O(n)$, however its main drawback is that it usually chooses relatively large subsets.

2.2.2. Incremental selection

This approach starts with $\mathcal{S} = \emptyset$ and runs over the ranking by iteratively testing $\mathcal{S} \cup A_{r_i}$ in a wrapper way. Then, if the wrapper evaluation obtained is better than the current one (corresponding to $\mathcal{S}$), A_{r_i} is added to $\mathcal{S}$, otherwise it is discarded. Obviously, this approach is also linear in the number of variables with the extra advantage over Rank Search that the evaluated models in practice have (far) fewer variables, $|\mathcal{S}|$ vs $\frac{n+1}{2}$ on average. Note that Rank Search evaluates exactly n subsets with cardinality $1,2,3,\ldots,n$, and therefore the average cardinality is $\frac{1}{n}\frac{n\cdot(n+1)}{2} = \frac{n+1}{2}$. However, the cardinality of the subsets evaluated by IWSS is bounded by $|\mathcal{S}|$. Thus, in the worst case, if all the n variables are selected, IWSS evaluates exactly the same subsets as Rank Search. However, from our experiments (see Table 2) $|\mathcal{S}| \ll n$, and so the complexity of the wrapper evaluations is clearly favorable for IWSS. Different proposals follow this idea with some modifications. In [12,19] a look-ahead parameter l is used to allow the search to finish when l consecutive attributes have been explored and discarded. BIRS [11] uses a relevance criterion based on the use of t-tests over the output of an inner 5-folds cross validation. Later, in IWSS [20] alternative relevance criteria to the use of a t-test are studied. In the experiments described in this paper we follow the suggestion of [20] and a variable is considered to be relevant (and so added to $\mathcal{S}$) if besides having a better mean in the 5-fold cross validation, it is also better in at least 2 out of the 5 folds.

2.2.3. Incremental selection with replacement

In [14] a more sophisticated incremental wrapper algorithm is presented: IWSSr. Now, when an attribute ranked in position i is analyzed, then not only its inclusion is studied but also its interchange with any of the variables already included in $\mathcal{S}$. Thus, the algorithm can retract from some of its previous decisions, that is, a previously selected variable can become useless after adding some others. As shown in [14] this new algorithm behaves in a similar way to the simpler (IWSS) incremental approach with respect to accuracy but obtains more compact subsets. Of course,

Table 1
Number of attributes, instances and class cardinality in the used datasets.

Dataset	#Feats.	#Inst.	$\lvert C\rvert$	Dataset	#Feats.	#Inst.	$\lvert C\rvert$
warpPIE10P	2421	210	10	warpAR10P	2400	130	10
pixraw10P	10,000	100	10	orlraws10P	10,304	100	10
TOX-171	5749	171	4	SMK-CAN-187	19,993	187	2
GLI-85	22,283	85	2	GLA-BRA-180	46,151	180	4
CLL-SUB-111	11,340	111	3	pcmac	3289	1943	2
basehock	4862	1993	2				

[1] Notice that this ranking can also be created by using the wrapper approach, but in this case it needs more CPU time which is biased to the classifier used.