A Simple Classifier based on a Single Attribute

Lei Du
Department of Computer Science and Technology
Xi’an Jiaotong University
Xi’an, China
dulei.323@stu.xjtu.edu.cn

Qinbao Song
Department of Computer Science and Technology
Xi’an Jiaotong University
Xi’an, China
qbsong@mail.xjtu.edu.cn

Abstract—Seeking a simple but effective classifier is exciting and meaningful in both machine learning and data mining. As usual, simplicity and high performance are two sides of a same coin. Our aim is to explore an easy-to-use classifier without losing its effectiveness. On this account, a single attribute based classification (SAC) algorithm is proposed. SAC first splits the original data set into multi one-dimensional data sets. After that, it creates a base classifier, e.g. C4.5, for each one-dimensional data, and then selects all classifiers having the highest accuracy. At last, SAC uses these selected classifiers to make prediction and the most frequent label is assigned to the new instance. Results of classification accuracy on 16 data sets from UCI machine learning repository indicate that the proposed method performs better in comparison with classical OneR algorithm. Experiments on high-dimensional data are also conducted to evaluate the proposed method, which demonstrates its scalability.

I. INTRODUCTION

Classification, as one of the key task of data mining and machine learning, has aroused wide attentions from researchers and communities. To date, a plethora of classification algorithms of different methodologies, e.g. decision tree based [1], [2], associative rule based [3], [4], [5] and so forth, have been seen in literatures. Even though, a simple but effective classifier or classification algorithm is yet preferred according to the “simplicity first” methodology, which means that a learner should search in a relatively small space containing only simple hypotheses [6].

OneR [6], a simple and easy classification algorithm, has been used widely in machine learning. Generally, it cannot be the best choice for a classification task since some other algorithms, e.g. C4.5 [2], tend to perform more accurate. The algorithm OneR obtains rules from every individual attribute of the training set, and then selects the rule with the highest accuracy (or lowest error rate) as its decision rule. This demonstrates that one single attribute sometimes can be effective. Thus there has reason to believe that every single attribute may contain information which is helpful to classification. However, OneR only randomly chooses one of the best rules, which means it make insufficient use of information lying in attributes when more than one rules are equally the best. Therefore, the accuracy of OneR cannot be very high.

For a multi-dimensional data set, it is believed that there are irreducible interactions between attributes [7]. These interactions can be helpful to classification, whereas they can be harmful to classification too. Take the weather data set as an example, on attribute humidity, we train a Bayes classifier and obtain correctly classification rate of 71.4%. However, when combining attribute humidity and attribute temperature together, the accuracy only reaches 57.1%. This is because the accuracy produced by temperature is only 42.9%, which can deteriorate the classifier. Well then, how about use all the attributes together? The accuracy is 64.3%, which is still lower than that merely on attribute humidity. This inspires us that a single attribute may obtain good performance.

How about OneR’s performance on data set weather? Then, its accuracy is only 42.9%, which is even worse than that of random guess. This again encourages us to explore new single attribute based method, which can make better use of the information provided by attributes.

To address the problem discussed above and to explore a simple and easy-to-use method, we treat each attribute independently by assuming they are mutually exclusive. That is, the original data set is first divided into multi one-dimensional data sets. For example, a data set of two attributes $D = \{A_1, A_2, y\}$ is divided into two subsets $d_1 = \{A_1, y\}$ and $d_2 = \{A_2, y\}$, where $y$ is their class label. Based on this, we proposed a novel classification algorithm SAC, acronym for Single Attribute based Classification, which is simple and yet effective. In the training phase, SAC builds one classical classifier on each one-dimensional subset and chooses the highest one as prediction model. Unlike OneR arbitrarily selecting one of the best rule, SAC takes into account all the best rules if there is more than one. In the classification phase, SAC uses the prediction model to classify an unknown instance. In the experimental study, we evaluate SAC, and compare it with oneR as well as several classical classification algorithms. The results show that the proposed SAC outperforms OneR and thus its accuracy is considerable.

This is a classical example data set in machine learning, and it is distributed with WEKA [8].
The remainder of this paper is organized as follows. In Section II, we introduce the background of our work. In Section III, we present the details of the proposed method. In Section IV, we evaluate our method with OneR and several benchmark algorithms on 16 public available data sets. The conclusion of this paper is contained in Section VI.

II. BACKGROUND

As early as 1993, Holte has indicated that very simple rules can perform well upon several commonly used data sets, and OneR algorithm is also introduced thereinto [6]. As we know, OneR cannot perform as well as some state-of-the-art classifiers, especially the mature classifiers such as C4.5. However, it is quite easy-to-use for its simplicity, and it is commonly used as a reference method.

In the training phase, OneR builds rules on each attribute of the training data, and randomly selects one of the rules having the highest accuracy (lowest error rate) as the decision rule. In the prediction phase, OneR classifies an unknown instance according to the previously obtained decision rule. The followings Algorithm 1 presents the algorithmic description of OneR.

Algorithm 1: OneR

inputs: Training set.
output: Decision rule.

1. for each attribute \( A_i \) do
2.   for each value \( v_i \) of \( A_i \) do
3.     count how often each class appears;
4.     find the most frequent class \( y_f \);
5.     make a rule = \{ \( A_i = v_i \land y = y_f \) \};
6.   end
7. calculate the accuracy of the rules;
8. end
9. return the rule with the highest accuracy.

As discussed earlier, OneR is simple at the expense of the accuracy. For this reason, there are several drawbacks within this algorithm, and the most critical one is its arbitrarily selecting a rule when more than one rules simultaneously have the best accuracy. This in fact can lose lots of information provided by those rest best rules. Thus this random selection is surely not the optimal choice.

Motived by these drawbacks, as well as the “simplicity first” methodology, we explore a method which makes sufficient use of information provided by all best attributes in this work. In the next section, a more detailed description will be presented.

III. SINGLE ATTRIBUTE BASED CLASSIFIER

A. General View of SAC

As aforementioned, SAC treats the original data set as combination of multi one-dimensional subsets, and it builds classifier upon each separately. Since some attributes may do harm to the classification, the classifier built on corresponding one-dimensional data set can be less effective. Therefore, SAC discards them and only chooses the best classifiers as the prediction model.

The framework of SAC is portrayed in Fig. 1. As it is shown, SAC consists of two parts, viz:

1) Training phase. SAC divides the original data set into multi one-dimensional ones. After that, it builds a base classifier for each one-dimensional data set. SAC selects all the best classifiers to establish the prediction model (see III-B for details).

2) Predicting phase. SAC uses the best classifiers on corresponding attributes, and produces the final prediction (see III-C for details).

B. Training Phase of SAC

Since some attributes may be harmful to classification, which means they could decrease the accuracy, SAC just selects the best attribute to make prediction. Previous of how to choose the best attribute, we first entail the following definition.

Definition 1: For a data set \( D \) of \( n \) attributes, we built \( n \) base classifiers, e.g. C4.5, on each attribute separately. Then we evaluate every classifier on corresponding attribute. We define the attribute on which the classifier having the highest accuracy to be the best attribute.

SAC trains on each subset, which consists of one attribute and the class label, and produces one classifier for each. Generally, the training phase of SAC works as follows.

Step 1: SAC divides the original data set into multi one-dimensional data sets which have the form of one attribute connected with the class label. For example, a data set of \( n \) attributes (the class attribute is excluded) \( D = \{A_i, y\} \) can produce \( n \) sub data sets \( d_i = \{A_i, y\}, 1 \leq i \leq n \).

Step 2: SAC builds \( n \) different base classifiers upon every one-dimensional data sets \( d_i \) depending on their respective characters.

Step 3: After the \( n \) classifiers have been built, SAC evaluates each classifier. Then the ones having the highest accuracy are selected as the final prediction model.
Note that the best attribute can be more than one. In order to make more accurate result, the proposed method uses all best attributes, rather than randomly choosing one as OneR does. SAC is also distinct from feature selection, which attempts to select all relevant features for building robust learning models [9], [10]. The goal of feature selection is to seek a subset of interacting features and combine them to train classifier. In this work, though all the best features are selected, they are treated separately. Now we present the algorithmic description of training procedure of SAC (Algorithm 2).

### Algorithm 2: SAC training

**inputs:** Training set: $T$  
**output:** Classifiers: $C$,  

Best attribute index: $Index$.  

1. initialize $maxAcc$, $C$ and $Index$; /* $maxAcc$ is the highest accuracy of classifier in $C$ */  
2. for each attribute $A_i$ do  
   3. contract data set $d_i = \{A_i, y\}$; // $y$ is class label  
   4. train a base classifier $C_i$ on $d_i$;  
   5. evaluate $C_i$ on $d_i$ to obtain $Acc_i$;  
   6. if $Acc_i > maxAcc$ then  
      7. replace classifiers in $C$ with $C_i$;  
      8. $Index$ is replaced by $i$;  
   else  
      9. if $Acc_i == maxAcc$ then  
         10. add $C_i$ to $Index$; // $Index$ becomes array  
      11. add $i$ to $Index$; // $Index$ becomes array  
   end  
12. end  
13. end  
14. return $C$, $Index$.  

In the above pseudo code, the first line is used for initialization. Lines 2~15, the key part of SAC, consists of three parts according to the three steps previously stated. The first part (line 3) is to divide the training set into multi one-dimensional data sets $d_i$. The second part (line 4) is used to create a base classifier on $d_i$. The last but not least part (lines 5~15) employs the same data set $d_i$ as the test set to evaluate the accuracy of just trained classifier. After that, if the accuracy $Acc_i$ is better than history max accuracy $maxAcc$, this attribute is called the best attribute thus far. It is then chosen as the unique classifier and other history classifiers are dropped. The present attribute index replaces the $Index$. (lines 7~8). However, if $Acc_i$ equals to $maxAcc$, which demonstrates that this attribute and history attribute are both the best attribute, SAC adds this classifier into history classifier pool $C$ and adds present attribute index to index pool $Index$ (lines 11~12). At last, when all attributes have been processed, SAC returns $C$ as its prediction model as well as $Index$ (line 16).

### C. Predicting Phase of SAC

After the training process, SAC obtains the prediction model which is trained on all the best attributes separately. For an unknown instance, SAC only selects the best attribute and uses the corresponding classifier to make classification. If there are more than one best attribute, the class label which is most frequently is assigned to this instance.

The predicting of SAC is quite simple: (1) it selects the best attribute according to the index pool $Index$ obtained from training phase; (2) it uses all the corresponding classifiers $C_i$ to predict an instance. Algorithm 3 contains the algorithmic description of this procedure.

As it is shown, the predicting phase of SAC can be divided into two parts according to the number of classifiers in $C$. The first part (lines 2~3) is triggered when there is only one best attribute. The index of this single attribute and the corresponding classifier are used to predict. The second part (lines 5~9) works when there are more than one best attributes. For each of them, SAC chooses the relative classifier to make prediction. After that, the most frequently appeared class label is assigned as the final result (line 10).

At last, SAC returns this specific class label $y$ (line 12).

### Algorithm 3: SAC predicting

**inputs:** Test set: $T$  
Classifiers: $C$  

Best attribute index: $Index$.  
**output:** class label: $y$.  

1. if only one classifiers $C_i$ in $C$ then  
   2. choose attribute $A_{Index}$ of $T$;  
   3. use $C_i$ to predict $y$;  
   4. else  
   5. for each classifier $C_i$ in $C$ do  
      6. choose attribute $A_{Index}$ of $T$;  
      7. use $C_i$ to predict $y$;  
      8. add $y_i$ to $Y = \{y_i\}$;  
   9. end  
10. assign the most frequent $y_i$ in $Y$ to $y$;  
11. end  
12. return $y$.  

### IV. Experiment and Results

In this section, a series of experiments are conducted to evaluate the performance of the proposed method SAC. We first depict the benchmark data sets commonly used by machine learning communities, and then discuss the experimental results in terms of comparison with that of some classical classification algorithms. We also use five public available high-dimensional data sets to evaluate SAC’s scalability in the latter of this section.

#### A. Data Source

To enable the experimental results reproducible, all data sets we use are publicly available at UCI machine learning repository [11].
Table I

<table>
<thead>
<tr>
<th>Data sets</th>
<th>Instances</th>
<th>Attributes</th>
<th>Class</th>
<th>Missing Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>BC (breast-cancer)</td>
<td>286</td>
<td>9</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>CH (kr-vs-kp)</td>
<td>3196</td>
<td>36</td>
<td>2</td>
<td>no</td>
</tr>
<tr>
<td>GL (glass)</td>
<td>214</td>
<td>9</td>
<td>6</td>
<td>no</td>
</tr>
<tr>
<td>GL2 (modified of GL)</td>
<td>163</td>
<td>9</td>
<td>2</td>
<td>no</td>
</tr>
<tr>
<td>HD (heart-disease)</td>
<td>303</td>
<td>13</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>HE (hepatitis)</td>
<td>155</td>
<td>19</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>HO (horse-colic)</td>
<td>368</td>
<td>22</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>HY (thyroid-disease)</td>
<td>3163</td>
<td>25</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>IR (iris)</td>
<td>150</td>
<td>4</td>
<td>3</td>
<td>no</td>
</tr>
<tr>
<td>LA (labor)</td>
<td>57</td>
<td>16</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>LY (lymphography)</td>
<td>148</td>
<td>18</td>
<td>4</td>
<td>no</td>
</tr>
<tr>
<td>MU (mushroom)</td>
<td>8124</td>
<td>22</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>SE (sick-euthyroid)</td>
<td>3163</td>
<td>25</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>SO (soybean-small)</td>
<td>47</td>
<td>35</td>
<td>4</td>
<td>no</td>
</tr>
<tr>
<td>VO (house-votes-84)</td>
<td>435</td>
<td>16</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>V1 (modified of VO)</td>
<td>435</td>
<td>15</td>
<td>2</td>
<td>yes</td>
</tr>
</tbody>
</table>

Table II

<table>
<thead>
<tr>
<th>Data, SAC_C4.5, SAC_NB, SAC_1NN, SAC_SVM, SAC_kNN, SAC_NB, OneR</th>
</tr>
</thead>
<tbody>
<tr>
<td>BC 67.6 67.3 67.3 67.4 67.6 67.3 67.4</td>
</tr>
<tr>
<td>GL2 79.1 73.0 73.0 75.5 73.0 73.0 76.1</td>
</tr>
<tr>
<td>HO 100 100 100 100 100 100 100</td>
</tr>
<tr>
<td>LA 80.4 79.3 69.8 80.4 77.5 73.7 72.3</td>
</tr>
<tr>
<td>SE 94.2 90.7 94.7 90.7 94.7 94.7 94.7</td>
</tr>
<tr>
<td>V1 87.4 87.4 87.4 87.4 87.4 87.4 87.4</td>
</tr>
<tr>
<td>higher than (%) 4.34 3.47 0.37 3.10 0.99 0.74 0.74 -</td>
</tr>
</tbody>
</table>

There are 16 data sets used in this study, and they are abbreviated to two-letter for ease of description. Some of these data sets are modified (Please see [6] for details). Other data sets are exactly as they are distributed. These data sets are selected based on the following considerations: number of attributes, number of instances, number of classes and whether or not there existed missing values. For ease of fair comparison, all data sets we selected are the same to that have been used in [6]. We present the characteristics of these data sets in Table I.

B. Experiment Setup

As a classification algorithm similar to OneR, we reasonably select OneR as the benchmark in the experiment. And in order to thoroughly access the performance of SAC, we also choose some other classical algorithms, including a tree based classifier C4.5 [2], a kernel based classifier SVM [12] and an instance based classifier 1-NN (k-NN) [13] and NB (Naive Bayes) [14] classifier.

When compared with all benchmarks, we use 5×10 folds cross-validation methodology to make the results reliable and dependable. The algorithms of OneR, C4.5, SVM, k-NN and NB are employed from WEKA [8], a well-known machine learning software tool. All the parameters here are set to their respective defaults.

As said in section III-B, for different attributes SAC can uses different base classifiers. Therefore, in the experiment, to clearly evaluate SAC’s performance, we use different base classifier in SAC. For example, SAC_C4.5 employs C4.5 as base classifier for training.

All experiments are conducted on a dual core (1.86 GHz) PC with 2GB main memory, running Windows XP.

C. Analysis of Results

Classification accuracy indicates the ability of a classifier correctly classifying the unlabeled instances. Thus we use it as the evaluation criterion here.

1) Comparison with OneR: Table II contains the comparison results in terms of accuracy of OneR and SAC wrapped...
with different base classifiers. From the table we observe that when adopting different classifiers, SAC obtains different accuracies, and we further find that SAC-C4.5 wins out for most times, i.e. 10 out of 16 data sets.

To see the general performance, the average accuracy obtained from 16 data sets is shown (Row Avg.), and the accuracy improved by SACs over OneR is also presented (Row higher than). For each data set, the highest accuracy is reported in boldface.

From the table, we observe that all SACs perform better than OneR in average accuracy. The accuracy of SAC-C4.5 (84.1%) is the best one followed by that of SAC-NB (83.4%). Specifically, the average accuracy of SAC-C4.5 is 4.34% higher than that of oneR, and the average accuracy of SAC-NB, SAC-kNN, SAC-SVM, SAC-kC4.5 (SAC with k-NN on continuous attributes and C4.5 on nominal attributes) and SAC-kNB (SAC with k-NN on continuous attributes and Naive Bayes on nominal attributes) are higher than that of OneR by 3.47%, 0.37%, 3.10%, 0.99% and 0.74% respectively.

2) Comparison with other benchmarks: To thoroughly evaluate the classification ability of the proposed method, we also show in Table III about the accuracy results of C4.5, NB, SVM, k-NN, OneR and SACs. Their respective average accuracy of 16 data sets are also shown (Row Avg.). For each data set, the highest accuracy is reported in boldface.

From the table, we observe that though SACs are not the best classifier, they perform well on most of these data sets, especially SAC-C4.5 and SAC-NB. We can also find that on data sets GL2, HO and SE, SAC-C4.5 outperforms all of the experimental classifiers.

According to all the comparison results above, we observe that SAC is statistically better than the classical OneR. Moreover, from Table III, we also find that SAC outperforms Naive Bayes in the average accuracy. On 3 data sets (GL2, HO and SO), SAC-C4.5 acts a winner among all the algorithms.

V. SCALABILITY EXPERIMENT ON HIGH-DIMENSIONAL DATA SETS

According to the previous discussion, SAC divides a training data into several one-dimensional data sets and then treats each as an independent classification task. Therefore, it transfers the classification problem in terms of multi-dimensional data set into multi one-dimensional data sets. This can be useful and positive, since its time consumption only linearly depends on the dimension of data set.

In accessing the performance of SAC on high-dimensional data set, we only employ OneR as benchmark here since other classical methods, e.g. C4.5, are not directly used due to their large time consumption on high-dimension data set. We also use SAC-C4 and SAC-NB in this experiment for convenience.

We use five publicly available high-dimensional data sets, i.e. HIV (Hiva_agnostic_valid), AR (Arrhythmia), EM (Embryonal_C), LE1(Leukemia_train) and LE2 (Leukemia_valid), and their separate details are presented as follows.

<table>
<thead>
<tr>
<th>Data</th>
<th>#Instances</th>
<th>#Attributes</th>
<th>#Class</th>
<th>Missing Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR</td>
<td>452</td>
<td>279</td>
<td>13</td>
<td>no</td>
</tr>
<tr>
<td>EM</td>
<td>60</td>
<td>7129</td>
<td>2</td>
<td>no</td>
</tr>
<tr>
<td>HIV</td>
<td>384</td>
<td>1617</td>
<td>2</td>
<td>no</td>
</tr>
<tr>
<td>LE1</td>
<td>38</td>
<td>7129</td>
<td>2</td>
<td>no</td>
</tr>
<tr>
<td>LE2</td>
<td>34</td>
<td>7129</td>
<td>2</td>
<td>no</td>
</tr>
</tbody>
</table>

AR is collected from UCI repository [11] and it has been used by Guvenir et.al [15], which aim is to discriminate between the presence and absence of cardiac arrhythmia. EM is used to predict the central nervous system embryonal tumour based on Gene (attribute) Expression [16]. HIV is drug discovery data set, having been used to predict which compounds are active against the AIDS HIV infection [17]. LE1 and LE2 are data sets of molecular classification of cancer [18]. Thereinto LE1 is the training set and LE2 is...
the test set. All five data sets are high-dimensional and their respective character details are contained in Table IV.

Table V shows the results of the accuracy comparison of SAC_C4 and SAC_NB with OneR, and their average accuracies are also shown (Row Avg.). Besides, the accuracy improved by SAC_C4 and SAC_NB over OneR are also presented. For each data set, we report the highest accuracy in boldface.

From the table, we observe that: (1) SAC_NB obtains more accurate results than OneR for all five data sets. Further, its average accuracy is higher than that of OneR by 4.51%; (2) SAC_C4 outperforms 3 times (AR, HIV and LE1) and equals 1 time, and its average accuracy is higher than OneR by 1.33%. This reveals that both SAC_NB and SAC_C4 perform better than OneR on high-dimensional data sets.

VI. CONCLUSION

In this paper, we have proposed a novel simple classification algorithm based on a single attribute of the original data set. It first divides the training data into multi one-dimensional data sets, and then builds one classifier on each of them. In the prediction, the method selects all the best classifiers and combines them for the final prediction.

The empirical results have revealed that SACs performed better than the classical OneR algorithm. Thereinto, SAC_C4 outperforms all benchmarks on three data sets (GL2, HO and SO). Furthermore, it is convenient to use SAC on high-dimensional data set which cannot significantly increase the time consumption, and the scalability experiments have demonstrated it performs better than OneR.

ACKNOWLEDGMENT

The authors would like to thank Guangtao Wang for his valuable advice. This work is supported by the National Natural Science Foundation of China under grant 61070006.

REFERENCES


