

Active Testing Strategy to Predict the Best Classification Algorithm via Sampling and Metalearning

Rui Leite¹ and Pavel Brazdil²

Abstract. Currently many classification algorithms exist and there is no algorithm that would outperform all the others in all tasks. Therefore it is of interest to determine which classification algorithm is the best one for a given task. Although direct comparisons can be made for any given problem using a cross-validation evaluation, it is desirable to avoid this, as the computational costs are significant. We describe a method which relies on relatively fast pairwise comparisons involving two algorithms. This method exploits sampling landmarks, that is information about learning curves besides classical data characteristics. One key feature of this method is an iterative procedure for extending the series of experiments used to gather new information in the form of sampling landmarks. Metalearning plays also a vital role. The comparisons between various pairs of algorithm are repeated and the result is represented in the form of a partially ordered ranking. Evaluation is done by comparing the partial order of algorithm that has been predicted to the partial order representing the supposedly correct result. The results of our analysis show that the method has good performance and could be of help in practical applications.

1 Introduction

Currently many classification algorithms exist and there is no algorithm that would outperform all the others in all tasks. Therefore it is of interest to determine which classification algorithm is the best one for a given task. One possible way of solving the problem would be by evaluating all possible candidates using a well-known cross-validation (CV) procedure. We could then order all candidates using the given evaluation measure, such as mean success rate and pick the first item in the ordering. However, this procedure is slow, particularly if it involves many algorithms.

In this paper we describe a method which relies on relatively fast pairwise comparisons involving two algorithms. This method is based on a previous one [8, 9] that exploits performance information of so called *sampling landmarks* representing accuracy estimates on simplified versions of the data [16, 6]. A series of sampling landmarks represents in effect a partial learning curve.

Earlier approaches have also explored data characteristics, captured in the form of various statistical and information-theoretic measures [14, 3]. Although methods based on sampling landmarks usually perform better than those that rely on data characteristics [8, 9], it is reasonable to suspect that using both types of information may represent the best solution. Some data characteristics could give

us clues concerning the best algorithm when considering a specific pair of classification algorithms.

One key feature of this method is an iterative procedure for building up the characterization in the form of sampling landmarks. The characterization involves conducting experiments on a new dataset. The plan of these experiments is built up gradually, by taking into account the results of all previous experiments – both on previously used datasets and on the new dataset. Metalearning plays a vital role in determining what should be done and which algorithm is recommended.

The comparisons between various pairs of algorithm are repeated and the result is represented in the form of a partially ordered ranking. If two or more alternatives have a comparable performance, no ordering exists between them in the ranking.

In this paper we also discuss how results of this form can be evaluated. The objects of study here are the predicted partial order of items which is compared to similar partial order representing the supposedly correct result. The results of this evaluation show that the method has good performance and could be of help in practical applications.

This paper is organized as follows. In Section 2 we describe the problem of determining the best of N classification algorithms in more detail. In Section 3 we review the method for determining which of the two given algorithms is better. This method exploits sampling landmarks and data characteristics. Section 4 discusses the problem of evaluation and presents the results.

2 Determining the Partial Order of Classification Algorithms

The issue of identifying the best classification algorithm(s) can be resolved in various ways. One simple possibility involves the following steps:

1. Pick a pair of classification algorithms and determine which one of the two is better using method T .
2. Repeat this for all pairs of algorithms.
3. Process the partial order obtained in order to identify the best element(s).

Let us refer to this method as $TestAll_T$, where the suffix T represents the test method used. It is a parameter of the method that needs to be supplied by the user. Let us analyze some of the steps in more detail, starting with step 1.

Step 1

Suppose we have chosen algorithms A_i and A_j . One simple method which is often used involves running a cross-validation (CV) pro-

¹ LIAAD-INESC Porto L.A./Faculty of Economics, University of Porto, Portugal, email: rleite@liaad.up.pt

² LIAAD-INESC Porto L.A./Faculty of Economics, University of Porto, Portugal, email: pbrazdil@liaad.up.pt

cedure and applying a statistical significance test. Let us refer to this method here as CV_{ST} (comparison using CV accompanied by a significance test). Later on in this paper we describe an alternative method (SAM, Selection of Algorithms using Metalearning) which does this faster and produces also good results. Here we will use the term T to represent all the methods of this kind, including CV_{ST} and SAM.

Method T (standing for CV_{ST} , SAM) can be seen as function applied to algorithms A_i and A_j which returns result $+1(-1)$, if A_i is significantly better (worse) than A_j . The value 0 is returned when it is not possible to determine whether A_i is better than A_j , or vice versa.

Step 2

involves applying T to all pairs of algorithms. The results for N algorithms can be represented using a graph where algorithms are represented by nodes and the relation is *significantly better than* by links.

Link $A_i \rightarrow A_j$ is drawn if and only if $T(A_i, A_j) = +1$. The graph results in a partial order of items. The function T may return 0, in which case no link is drawn and consequently the order of these items is undetermined. Some authors use also the term *quasi-linear ranking*, instead of partial order of items [4].

Fig. 1 shows an example of a partial order of 6 algorithms representing the result for Abalone dataset, obtained by the method CV_{ST} discussed earlier. More details about the classification algorithms used here can be found later (Section 4).

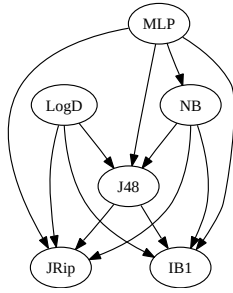


Figure 1. An example of partial order of 6 classification algorithms obtained by $TestAll_{CV_{ST}}$ for Abalone dataset

Step 3

The aim is to analyze the partially ordered set of items and identify the topmost element(s).

Basically, we can consider the following strategies:

- L, considering losses (incoming arrows),
- W, considering wins (outgoing arrows),
- LW, considering both losses and wins.

It is clear that if we adopt strategy L in the example shown in Fig. 1, the item MLP and LogD have the least number of losses (zero losses) and hence they should both share the topmost level (level 1). As item NB has suffered one loss (represented by an incoming arrow if we use a graph), it should be placed on level 2.

Strategy W gives a somewhat different result. MLP is placed on level 1, as it has scored 4 wins, while LogD and NB appear on level

2 (3 wins). Strategy WL gives yet another result. Suppose wins are represented by $+1$, losses by -1 and draws by 0. If we use this scheme, MLP scores $+4$, LogD gets 3 points, while NB gets 2 points.

Here we have adopted strategy L, although we had no particular reason for preferring this strategy to the others.

Method for reordering elements into levels

The general method for reordering elements into levels using strategy L is quite straightforward. Let us assume that the graph is represented in the usual way, that is using a set of nodes, N and a set of oriented links, L . In the first step the aim is to identify nodes with the least number of incoming arrows (i.e., the least number of losses).

Let us consider the example of 6 partially ordered algorithms shown earlier in Fig. 1. As MLP has no incoming arrows, it is placed at the topmost level (level 1). Then all nodes identified in this step are marked and so are all the links originating from these nodes. In our example this includes node MLP and the links $MLP \rightarrow JRip$, $MLP \rightarrow J48$, $MLP \rightarrow NB$ and $MLP \rightarrow IB1$. The whole process is then repeated. As $LogD$ and NB have no incoming links they are placed on level 2. The process continues like this until termination.

3 Predicting the Relative Performance of a Pair of Algorithms

The aim in this section is to describe the method SAM which uses both data characteristics and performance on samples to predict which of the two given algorithm (A_p and A_q) should be used for a new dataset (i.e. dataset for which we seek a recommendation). The approach described follows the general strategy of metalearning [2]. The method relies on two kinds of information:

- Information about various datasets used in the past. This information is of two kinds: (1) data characteristics, and (2) information about accuracies of the two classifications algorithms on samples of data from these datasets. Following [2] this information is referred to as *metaknowledge*.
- Information about the new dataset. This is again of two kinds, similar to above, but the information about accuracies of the two classifications algorithms is only *partial*. It includes accuracies relative to *some data samples* only.

The method is based on an intuition that performance of the two algorithms on a set of samples sheds some light on the final outcome of the learning. Let us analyze Fig. 2 that illustrates this. It shows the measured accuracy on three samples of data s_1, s_2 and s_3 . Here $s_1 \dots s_m$ are used to represent random samples of increasing size. The two curves have been extended on the basis of other similar curves retrieved from the metadatabase. It enables to determine that algorithm A_q will achieve higher accuracy at the end, that is, when the whole dataset is used. The details regards this method are described in the following.

Metaknowledge about Past Datasets

As has been mentioned before, the method described here relies on the existence metaknowledge about various datasets dealt with in the past. Let us refer to these datasets as $d_1 \dots d_n$. This information is of two kinds: data characteristics and information about how various classifications algorithms performed on samples of data from these datasets.

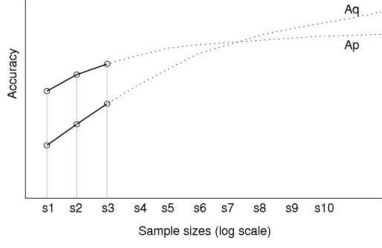


Figure 2. Example of two extended learning curves determining which algorithm is better

Regards data characteristics here we consider the set presented in Table 1. We have chosen this set as it was used in other studies [15] and it was shown that this set achieves similar results in metalearning tasks as a much larger set suggested in previous work [14].

Table 1. Data Characteristics Measures Used

Id	Measure
DC_1	Number of examples
DC_2	Proportion of symbolic attributes
DC_3	Proportion of missing values
DC_4	Proportion of numerical attributes with outliers
DC_5	Class entropy
DC_6	Average mutual information between attributes and the class
DC_7	First canonical correlation

Besides this, we need information about how various classifications algorithms performed on an exhaustive set of samples of data from these datasets. In other words, we need information about full learning curves.

Metaknowledge about the New Dataset

This metaknowledge is again of two kinds. First, it includes data characteristics, such as those shown earlier 1.

Besides, the method requires some information about how the chosen two classifications algorithms performed on *some* samples of data from the new dataset. Normally only some of these meta-attributes are sufficient for the prediction. In other words, only a part of each learning curve is sufficient for making quite good prediction.

In the next section we describe method SAM_F that uses a fixed set of samples. Further on (in Section 3.2) we describe a more general approach (SAM), that uses an automatic mechanism to determine which samples should be used.

3.1 Using Data Characteristics and Performance Characteristics based on a Fixed Set of Samples

This method described here, referred to as SAM_F , is based on previous work [9] which exploits only partial learning curves to predict the best classifier. As we shall see in the next section it forms a part of the more general method SAM . The method requires as input a fixed set of samples for each algorithm. Let S_p (S_q) represent the samples required to characterize algorithm A_p (A_q). So, for instance, the samples passed as input can be $S_p = \langle s_1, s_2, s_3 \rangle$ and $S_q = \langle s_1, s_2 \rangle$. The method encompasses the following steps:

1. Compute the data characteristics for the new dataset d .

2. Characterize the new dataset d by conducting experiments with algorithm A_p on S_p samples and measuring accuracies. Repeat this for A_q on S_q samples. In other words build two partial learning curves.
3. Compute the distances between the information relative to dataset d and stored information relative to all other datasets $d_1 \dots d_n$ (details in the next subsection).
4. Identify the subset of k nearest datasets.
5. For each of the k nearest datasets identified and retrieved in the previous step, *adapt* each pair of learning curves to the new partial learning curves build for dataset d . Adaptation is done by rescaling each retrieved learning curve in order to minimize the square distance from this curve to the respective partial learning curve for dataset d .
6. For each pair of adapted curves decide which algorithm achieves higher performance on the adapted and extended learning curves.
7. Identify the algorithm that is better on the new dataset d , by considering the results on k pairs of nearest datasets.

More details on the distance function used in Step 3

The distance function used consists of two parts. One that calculates the distance using data characteristic measures and another that uses the estimates of accuracies of the algorithms on samples. It is defined by the following equation:

$$d(d_i, d_j) = w_{DC} * d_{DC}(d_i, d_j) + w_S * d_S(d_i, d_j) \quad (1)$$

where w_{DC} and w_S are weights (normally set to 0.5 each). The distance relative to dataset characteristics is calculated using the equation shown below. Each measure is first rescaled to fit the interval [0,1].

$$d_{DC}(d_i, d_j) = \sum_{k \in DC} \frac{|DC_t(i) - DC_t(j)|}{\max(DC_k) - \min(DC_k)} \quad (2)$$

where DC contains the indices of classical data characteristics. The distance relative to accuracies on samples is calculated using the equation:

$$d_S(d_i, d_j) = \underbrace{\sum_{s \in S_p} |A_{p,i,s} - A_{q,j,s}|}_{A_p} + \underbrace{\sum_{s \in S_q} |A_{q,i,s} - A_{p,j,s}|}_{A_q} \quad (3)$$

where S_p (S_q) contains the indices of meta-features used to characterize the new case (dataset) using accuracies of algorithm A_p (A_q) on specific samples. The symbol $A_{p,i,j}$ represents the meta-feature corresponding to the estimate of accuracy of algorithm A_p on a sample of size s_j extracted from dataset d_i .

The disadvantage of the method just described is that the user has to decide how many samples should be used and what their sizes should be. This is not easy to determine.

If we use the performance on more samples, or larger samples, it is reasonable to expect that better decisions will be made, as the shape of the learning curves will be better defined. However the computational costs involved in computing the meta-attributes (i.e. performance of the algorithms on these samples) will rise too. If we use fewer samples, or smaller samples, the quality of the decision will decrease. The advantage is that the computational costs will also decrease. A good decision involves striking a good compromise. In the next section we describe a method that resolves this issue.

3.2 Method that Determines which Samples to Use

Method SAM described here seeks a good compromise between trying to achieve a good performance and keep the computational costs to a minimum. In this respect it differs from an ordinary feature selection method (e.g. forward selection) that would only try to improve the accuracy of the method. The desired sequence of samples is identified using a hill climbing approach. At each stage the existing set of characteristics is considered and a plan of possible extensions is built up. These are ordered by costs (i.e., time needed to compute them). The aim is to improve the capacity of the method to predict which of the two algorithms is better for a given task. The plan formulated earlier is followed to determine which of the extensions defined earlier can improve the meta-accuracy of the method by a certain pre-defined amount Δ . As the alternatives are ordered by increasing costs, the first alternative that is expected to provide this improvement is accepted. The method continues like this until no further improvement can be made. An overview of the method is presented in the following:

1. Compute the data characteristics for the new dataset d .
2. Initialize the meta-characteristics of dataset d based on samples of data (e.g., with $S_{p,1} = \langle s_1 \rangle$ and $S_{q,1} = \langle s_1 \rangle$). Call SAM_F to determine which algorithm (A_p or A_q) is better and store the prediction on a stack.
3. Generate a plan concerning how to extend the existing meta-characteristics. Consider both classical characteristics and characteristics based on samples of data. Order the alternatives by increasing costs (i.e., time needed to compute them). More details on this are given later.
4. For each alternative defined in the plan conduct virtual experiments with SAM_F to estimate the corresponding meta-accuracy. Select the first alternative with the least cost that is likely to improve the previous meta-accuracy by a given amount Δ . If no alternative has been identified, go to step 6 to terminate.
5. Call SAM_F with the alternative identified to determine which algorithm (A_p or A_q) is better. Store the prediction on a stack and go back to step 3.
6. Retrieve the last prediction from the stack regards which algorithm (A_p or A_q) is better and output it.

More details on step 4

Virtual experiments are conducted by exploiting solely the information stored in the meta-database. Therefore these experiments are rather fast to execute. Method leave-one-out is followed. In each cycle one dataset is left out, and method SAM_F is called with the proposed set of meta-features. The aim is to obtain a prediction for the case in question and make comparisons with correct value. After all cycles of leave-one-out, it is possible to determine the estimate of meta-accuracy.

4 Evaluation of Method SAM used for Pairwise Comparisons

In this section we describe the evaluation methodology and also the experiments whose aim was to evaluate the method SAM used in pairwise comparisons.

Evaluation Methodology

To evaluate the method proposed we need to compare, for each case (dataset) and a given pair of algorithms, the predicted class with the true classification determined by usual cross-validation evaluation procedure. A statistical test (t-test) is used to compute the statistical significance. Instead of using the usual accuracy measure, a different measure is used here that is more suited for our classification task with 3 possible outcomes. The errors are called *penalties* and are calculated as follows:

- If the classification is +1 (or -1), while the true class is -1 (or +1), this counts as a complete failure, and the penalty is 1.
- If the method classifies the dataset as 0, while the true class is +1 (or -1) then we consider that the method partially failed. The penalty is 0.5.
- If a particular method (e.g. SAM) classifies some case as +1 (or -1), while the true class is 0 (the given algorithms are not significantly different) then, from a practical point of view the method did not fail, because any decision is a good one. Therefore the penalty is 0.

The corresponding accuracy, referred to as *meta-accuracy*, is computed using the formula $1 - \sum_{d \in D} \frac{\text{penalty}(d)}{|D|}$, where D is the collection of datasets.

Evaluation Set-up

In this empirical study we have used the following 6 base-level classification algorithms, all implemented within Weka [7] machine learning tools: J48 (a C4.5 tree implementation), JRip (rule set learner), LogD (logistic discriminant), MLP (multi-layer perceptron), IB1 (instance-based learner) and NB (naive bayes).

Using this setting we get 15 classification problems, one for each pair of algorithms. In total 40 datasets were used in the evaluation. Some come from UCI [1], others from the project METAL [13]. The datasets were characterized using 6 data characteristic measures presented in Table 1.

Regards meta-attributes based on performance of classification algorithms on samples, samples $s_1 \dots s_{10}$ were used. The calculation of the size of s_i was determined using the following formula $2^{6+0.5 \times i}$. For s_1 , the size of the first sample, this gives $2^{6.5}$, i.e. 91 cases after rounding. For s_2 this gives 2^7 , i.e. 128 cases etc. Each sample included randomly chosen cases subject to the restriction that s_i be a subset of s_{i+1} for all i . This restriction helps to restrict the variance of the resulting classifier to a minimum.

For each decision problem the method was evaluated using a leave-one-out methodology. In each case we measure meta-accuracy and computational cost. Computational costs are expressed as a ratio of times. The time required by the particular method (e.g. SAM) is normalized, by dividing it by the time required to obtain the decision by cross-validation. Cross-validation represents a slower, but in principle it is a more reliable method.

In the experiments the following setting were used $\Delta = 0.07$ and $\Delta_{DC} = 0.001$ (the threshold for data characteristics). Since the data characteristic measures were previously shown to be less predictive in comparison with performance on samples meta-attributes the value of Δ_{DC} was set to a much smaller value than the value of Δ .

Regards the number of neighbours, here we have used $k = 23$ which was shown to be a good setting in previous experiments [10].

Results

Table 2 shows the results including the meta-accuracies. The decision problem in question is presented in column 1. Default meta-accuracy, determined as usual by the most frequent class, is shown in column 2³. Meta-accuracy is shown in column 3.

³ The default meta-accuracy provides an estimate of how difficult the meta-classification problem is. For instance if an algorithm A is better than B on 80% of the datasets, the best decision is to use A . This problem is easier than the one where A and B have about the same chance of achieving a better result on a new dataset.

As can be seen it is rather high on average (92.35) indicating that in a high proportion of cases method *SAM* recommended the right decision. The method is much more precise than the method that uses classical dataset characteristics (its average meta-accuracy was 81.15).

Table 2. Meta-accuracy of the method (average)

Algorithm Pair	Default Accuracy	Meta-Accuracy (SAM)	Meta-Accuracy (SAM without DC)	Meta - Accuracy (SAM just DC)
IB1-J48	80.00	92.50	92.50	95.00
IB1-JRip	72.50	92.50	92.50	87.50
IB1-LogD	66.67	97.22	94.11	80.56
IB1-MLP	84.21	92.11	92.11	85.53
IB1-NB	67.50	92.50	92.50	70.00
J48-JRip	75.00	90.00	90.00	76.25
J48-LogD	77.78	97.22	97.22	70.83
J48-MLP	63.16	86.84	86.84	75.00
J48-NB	85.00	90.00	90.00	76.25
JRip-LogD	75.00	86.11	86.11	72.22
JRip-MLP	68.42	89.47	89.47	81.58
JRip-NB	82.50	92.50	92.50	78.75
LogD-MLP	80.00	97.14	97.14	78.57
LogD-NB	94.44	94.44	94.44	94.44
MLP-NB	94.74	94.74	94.74	94.74
Mean	77.79	92.35	92.17	81.15

Regarding the costs of the method, measured by the time spent on computing required metafeatures, the experiment results have shown that the method is about 7 times(0.139 of CV) faster than cross-validation. The version that did not use the *data characterization* spent just 0.137 compared to CV. The method that relies on classical data characteristics, *SAM_{DC}*, was faster (0.007 of CV), but its accuracy was much lower.

It is interesting to analyse which sequences of samples were generated by SAM. Here we present some typical patterns: $S_{IB1} = \langle s_1, s_2, s_4 \rangle$ and $S_{NB} = \langle s_1, s_2, s_5 \rangle$ (abalone), $S_{IB1} = \langle s_1, s_2, s_6 \rangle$ and $S_{NB} = \langle s_1, s_2, s_5 \rangle$ (acetylation), $S_{J48} = \langle s_1, s_8 \rangle$ and $S_{JRip} = \langle s_1, s_8 \rangle$ (abalone), $S_{J48} = \langle s_1, s_8 \rangle$ and $S_{JRip} = \langle s_1, s_8 \rangle$ (byzantine).

We note the sequences generated are typically quite short. The ones shown above consist of at most three samples. Furthermore we note that the sequences may include gaps and nonconsecutive samples. For instance, in the first example shown that involves the algorithms *IB1* and *NB* on *abalone* dataset, the samples s_1 and s_2 are not followed by s_3 , but rather by s_4 . The estimate of accuracy on s_3 is not sufficiently informative and does not allow to required increase the meta-accuracy. Therefore a larger sample (s_4) is used. Finally, we note also different algorithms are characterized using different samples. This is yet another interesting feature of the method. A similar analysis can be made regards classical data characteristics. We have observed that most of the times these measures were not selected by *SAM* and when they were selected the decision concerning the best algorithm remained basically unaffected. The measures seemed to have been useful for a relatively small proportion of datasets. For some problems (e.g. IB1-MLP) the method did not select any data characterization measure. Some measures were selected more often than others. For instance in problem J48-MLP, DC6 (av. mutual information) was selected in 18.4% of the datasets. The figures were in the range between 0% and 18.4%. More details about this study can be found in [10].

5 Evaluation of the Method for Determining the Best of N Algorithms

Evaluation involves comparing a prediction to some baseline. In a general case both are partially ordered rankings. The prediction is represented by a partially ordered ranking returned by the variant of the method *TestAll_{SAM}* which calls *SAM*. The baseline is represented by the partially ordered ranking returned by the variant of method *TestAll_{CV_{ST}}* which calls *CV_{ST}*.

It should be stressed that the method *TestAll_{CV_{ST}}* is not entirely reliable, as Type I and Type II errors may occur. However, the aim here is to use this method as a baseline for comparison purposes. The question is how well the method *TestAll_{SAM}* performs when compared to the baseline. The problem that we wish to discuss next is what metric should be used that would permit to evaluate the quality of predicted ranking with respect to the supposedly correct ranking.

Here we will focus on just the first level in both rankings. This makes sense, as normally we are interested to identify those algorithms that are considered to be the best ones (in general, there may be more than one). We want to have a way of evaluating such situations.

Evaluation that focuses on the top level in both rankings

The aim of this section is to describe an evaluation method that is oriented towards just the first level in both rankings. Before describing our proposal, let us analyze a few situations that may occur (see Table 3).

Cases 1 and 2 are straightforward. As the predictions agree with the correct value, the measure of success (MS) is 1. Case 3 is again straightforward. The prediction involves only one item (A_1) and as this disagrees with the baseline prediction (A_2), the measure of success is 0.

In case 4 the prediction and the correct value are not identical. Our strategy here is to adopt a "benevolent position" and still consider that the measure of success is 1 in this case. This can be justified by the fact that if A_1 and A_2 are equivalent, it is good enough for the method to propose one of them (A_1 in this case).

Case 5 involves a prediction of two items (A_1, A_2), while only A_1 is correct. Here the predictions that include more than one item are considered as *uncertain predictions*. These should be accompanied by probabilities. If we have no specific information available, the best is to assume that the predictions are equi-probable. (i.e. here both 1/2).

Table 3. Examples of some possible situations involving the top level

Case	Prediction <i>TestAll_{SAM}</i>	Correct value <i>TestAll_{CV_{ST}}</i>	Measure MS
1	A_1	A_1	1
2	A_1, A_2	A_1, A_2	1
3	A_2	A_1	0
4	A_1	A_1, A_2	1
5	A_1, A_2	A_1	0.5

The method can be formalized as follows. First, let us introduce some notation. Let $\hat{Y}_k$ represent the set of elements predicted for case k. Let Y_k represent the set of elements that are correct for case k. So, for instance, if we consider the data shown in the table above, $\hat{Y}_5$ returns A_1, A_2 and Y_5 returns A_1 . The method for calculating MS relative to case k can be expressed as follows:

$$MS_k = \sum_{i,j} I(\hat{Y}_k[i], Y_k[j]) * p(\hat{Y}_k[i])$$

where $I(x, y)$ returns 1 if $x = y$ and 0 otherwise and $p(\hat{Y}_k[i])$ represents the probability accompanying the prediction i for case k .

Evaluation results for 6 algorithms

The evaluation results for 6 algorithms and some of the datasets used in our tests are shown in Table 4 (here we present only the first 7 cases for illustration). As can be seen, our method does the right recommendation in most cases as is indicated by the measure of success, $MS = 1$. The average value of the measure of success, MS , on all datasets considered was 0.9.

Table 4. Evaluation results for 6 algorithms on different datasets

	Prediction	Correct Value	Measure MS
krkopt	MLP	MLP	1
krvskp	J48	J48 JRip MLP	1
led24	J48	J48 LogD NB	1
letter	MLP	IB1	0
mfeat	IB1	IB1 MLP	1
musk	MLP	MLP	1
nursery	MLP	MLP	1
...	...	...	...

6 Discussion

Other approaches for evaluation of rankings

In section 5 we have presented an evaluation method that has focused on the top level in both rankings. However, sometimes we may want to consider more levels in the rankings. We may want to distinguish situations when the prediction is not far from the ideal from others. One possibility is to adopt a measure of correlation (e.g. Spearman correlation) or perhaps better, *weighted rank measure* that attributes more weight to items in top positions [5]. Another possibility is to assume that the evaluation is carried out with respect to the top level in the correct ranking and the aim is to determine how far down in the predicted ranking are the correct items. A measure of this kind, *estimated rank of best algorithm*, was used in [11]. Alternatively we could assume that the evaluation should be carried out with respect to the top level in the predicted ranking and the aim is to determine how far down in the correct ranking are the predicted elements. A measure of this type, referred to as *average rank of predicted algorithm*, was used in [12].

7 Conclusions

In this paper we have described method SAM that exploits both classical data characteristics and estimates of accuracy on data samples to determine which of two given classification algorithms is better on a new dataset. The metafeatures (meta-attributes) used by the method are either data characteristics of the dataset in question, or performance estimates of the given classification algorithms on samples of specified sizes. The method automatically establishes how many samples are needed and their sizes. Besides the method also determines which data characteristic measures are useful for the decision concerning which is the best algorithm. Experimental evaluation has shown that method SAM used to determine the best algorithm in pairwise comparisons achieves good performance when compared to the other methods. The average meta-accuracy of this method was 92.35%.

Besides we have described also method $TestAll_{SAM}$ that can be used to predict the best classification algorithm for a given set of

algorithms. It achieved very good results. In our experiments with 6 algorithms the method described did the right recommendation in most cases. The average value of the measure of success, MS , on all datasets considered was 90%. The methods described represent thus a significant improvement over previous methods for dealing with the problem of predicting the relative performance of learning algorithms in a systematic manner.

REFERENCES

- [1] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.
- [2] P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta, *Metalearning: Applications to Data Mining*, Springer, 2009.
- [3] P. Brazdil, C. Soares, and J. Costa, 'Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results', *Machine Learning*, **50**, 251–277, (2003).
- [4] W.D. Cook, M. Kress, and L.W. Seiford, 'A general framework for distance-based consensus in ordinal ranking models', *European Journal of Operational Research*, **96**(2), 392–397, (1996).
- [5] Joaquim P. Costa and Carlos Soares, 'A weighted rank measure of correlation', *Australian and New Zealand Journal of Statistics*, **47**(4), 515–529, (2005).
- [6] Johannes Fürnkranz and Johann Petrak, 'An evaluation of landmarking variants', in *Proceedings of the ECML/PKDD Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning (IDDM-2001)*, pp. 57–68. Springer, (2001).
- [7] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten, 'The weka data mining software: an update', *SIGKDD Explorations*, **11**(1), 10–18, (2009).
- [8] Rui Leite and Pavel Brazdil, 'Predicting relative performance of classifiers from samples', in *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pp. 497–503, New York, NY, USA, (2005). ACM Press.
- [9] Rui Leite and Pavel Brazdil, 'An iterative process for building learning curves and predicting relative performance of classifiers', in *Progress In Artificial Intelligence, Proceedings of the 13th Portuguese Conference on Artificial Intelligence Workshops (EPIA 2007)*, eds., J. Neves, M.F. Santos, and J. Machado, volume 4874 of *Lecture Notes in Computer Science*, pp. 87–98, Guimarães, Portugal, (December 2007). Springer.
- [10] Rui Leite and Pavel Brazdil, 'Selecting classifiers using metalearning with sampling landmarks and data characterization', in *Proceedings of the Planning to Learn Workshop (PlanLearn 2008)*, held at *ICML/COLT/UAI 2008*, eds., P. Brazdil, A. Bernstein, and L. Hunter, pp. 35–41, Helsinki, Finland, (July 2008).
- [11] D. Ler, I. Koprinska, and S. Chawla, 'A new landmarker generation based on correlativity', in *Proceedings of the IEEE International Conference on Machine Learning and Applications*, pp. 178–185. IEEE Press, Louisville, USA, (2004).
- [12] D. Ler, I. Koprinska, and S. Chawla, 'Utilizing regression-based landmarks within a meta-learning framework for algorithm selection', in *Proceedings of the Workshop on Meta-Learning, associated with 22nd International Conference on Machine Learning*, pp. 44–51, Bonn, Germany, (2005).
- [13] Metal project site. <http://www.metal-kdd.org/>, 1999.
- [14] D. Michie, D.J. Spiegelhalter, and C.C. Taylor, *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, 1994.
- [15] C. Soares, *Learning Rankings of Learning Algorithms*, Ph.D. dissertation, Department of Computer Science, Faculty of Sciences, University of Porto, 2004.
- [16] Carlos Soares, Johann Petrak, and Pavel Brazdil, 'Sampling-based relative landmarks: Systematically test-driving algorithms before choosing', in *Proceedings of the 10th Portuguese Conference on Artificial Intelligence (EPIA 2001)*, pp. 88–94. Springer, (2001).