Reduction Stumps for Multi-class Classification

Felix Mohr, Marcel Wever(✉), and Eyke Hüllermeier

Heinz Nixdorf Institute, Department of Computer Science, Paderborn University, Paderborn, Germany
marcel.wever@uni-paderborn.de

Abstract. Multi-class classification problems are often solved via *reduction*, i.e., by breaking the original problem into a set of presumably simpler subproblems (and aggregating the solutions of these problems later on). Typical examples of this approach include decomposition schemes such as one-vs-rest, all-pairs, and nested dichotomies. While all these techniques produce reductions to purely binary subproblems, which is reasonable when only binary classifiers ought to be used, we argue that reductions to other multi-class problems can be interesting, too. In this paper, we examine a new type of (meta-)classifier called *reduction stump*. A reduction stump creates a binary split among the given classes, thereby creating two subproblems, each of which is solved by a multi-class classifier in turn. On top, the two groups of classes are separated by a binary (or multi-class) classifier. In addition to simple reduction stumps, we consider ensembles of such models. Empirically, we show that this kind of reduction, in spite of its simplicity, can often lead to significant performance gains.

Keywords: Multi-class classification · Reduction · Ensembles
Automated machine learning

1 Introduction

Reduction of a multi-class classification problem means breaking down the original problem into other presumably simpler subproblems. Typical examples include one-vs-rest and all-pairs decomposition [6], as well as nested dichotomies [5]. One-vs-rest creates one binary problem for each class, in which the class is separated from the rest, whereas all-pairs creates a binary problem for each pair of classes. Nested dichotomies reduce the given problem by recursively splitting the set of classes, which yields a binary tree structure where each leaf has one class and each inner node is meant to separate the classes occurring under the left child from the classes occurring under the right child. In general, the subproblems created by reduction are all binary. Thus, reduction makes multi-class problems amenable to binary classifiers, which can be seen as their main merit.

W. Duivesteijn et al. (Eds.): IDA 2018, LNCS 11191, pp. 225–237, 2018.
https://doi.org/10.1007/978-3-030-01768-2_19

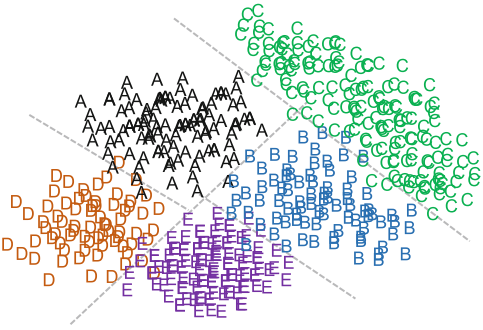

Fig. 1. A multi-class problem with five classes.

While a complete reduction to binary problems is necessary when only binary classifiers can be used, reductions to other multi-class problems might be interesting as well. A priori, one cannot exclude that modifying an underlying multi-class problem by reducing it to other multi-class problems (on less but possibly more than two classes) is beneficial for a learner, even if the latter is principally able to solve multi-class problems right away. For example, one may suspect that a multi-class learner like a random forest or a neural network could benefit from an explicit reduction of the problem shown in Fig. 1.

An interesting area of research where such reductions can be important is automated machine learning (AutoML). AutoML aims at automatically finding a machine learning "pipeline" (including methods for data preprocessing, model induction, etc.) that optimizes a performance measure of interest for a given learning task (typically specified by a dataset). A couple of approaches to AutoML have been proposed [4,10–12] in the recent past. Currently, however, reduction is hardly considered by these approaches, although its potential benefit is completely unclear.

In this paper, we examine a new classifier that we call *reduction stumps*. Reduction stumps split the given set of classes into two subsets and then solve the resulting problems with a multi-class classifier. Separating instances of the two subsets is achieved by a third (and possibly binary) classifier. The motivation for looking at this type of classifier is that it offers a middle-ground solution between native multi-class classifiers and a complete reduction to binary problems as in nested dichotomies. Therefore, reduction stumps achieve a reasonable compromise between simplification and complexity.

Due to this property, reduction stumps offer an interesting means to achieve performance gains in AutoML. However, the effort to determine such reductions needs to be moderate, as AutoML tools must consider many different machine learning pipelines and cannot spend too much time on a single decision, such as whether or not reduction should be used. Besides, each of the reduced problems gives rise to a new AutoML problem itself, so again for reasons of complexity, one should avoid creating too many of them.

Empirically, we show that (ensembles of) reduction stumps indeed lead to significant performance gains over *many* other classification algorithms in a significant number of cases. More precisely, we compare reduction stumps against several standard classification algorithms, including multi-class logistic regression, neural networks, nearest neighbors, support vector machines via all-pairs reduction, decision tress, and random forests. We find that reduction stumps or ensembles of them are better than *any* of the other algorithms in more than half of the 21 examined datasets from the UCI repository [1]. Even though the improvement over the best non-reduction-based classifier is often only small, we thus provide a strong justification for the consideration of reduction stumps when solving multi-class classification problems.

2 Background: Nested Dichotomies

Nested dichotomies (NDs) reduce a multi-class classification problem to a set of (presumably easier to solve) binary problems. To this end, the original set of classes (as long as comprising more than one element) is recursively split into two nonempty subsets. Thus, an ND defines a binary tree, in which every node is labeled with a set of classes, such that every leaf is labeled with a distinct class, and every inner node with the union of labels of its children. Figure 2 shows two example dichotomies for the case of four classes.

To each inner node of an ND, a classifier is attached and fitted for the task of discriminating the two sets of classes (meta-classes) assigned to its children. These classifiers are trained using a base learner, which is typically supposed to produce probabilistic predictions. For a given instance, the class with the highest probability is predicted, where the probability for a leaf is obtained by multiplying the probabilities predicted by the base learners along the path from the root to the respective leaf node.

Of course, the performance of an ND critically depends on the structure of the binary tree, as it determines the complexity of the induced binary classification problems, as well as the type of classifier attached to the inner nodes. Usually, the type of classifier is fixed, and one is interested in finding the most beneficial structure [9]. The criterion that is commonly optimized is the overall predictive accuracy (percentage of correct classifications), which depends on the quality of the binary classifiers, and therefore on the structure of the ND. Approaches to finding suitable dichotomies are based on random sampling [5,9], greedy error minimization [8], clustering [3], and evolutionary optimization [13].

A *partial* nested dichotomy is an ND in which the leaf nodes may be labeled with more than only one class. Therefore, partial NDs need to be equipped with multi-class classifiers in their leaf nodes. Of course, other reduction techniques such as one-vs-rest or all-pairs [6] could be used here as well. To our knowledge, partial dichotomies have only occurred during the construction process of complete nested dichotomies, but have never been used as classifiers themselves.

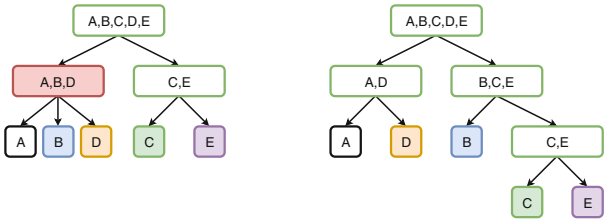

Fig. 2. A partial (left) and a complete (right) nested dichotomy for five classes.

3 Reduction Stumps and Reduction Stump Ensembles

In this paper, we focus on partial NDs of depth 1, which we refer to as *reduction stumps*. Reduction stump have exactly one inner node, which is the root, and two child nodes with an arbitrary number of classes. Of course, just like NDs, reduction stumps are multi-class classifiers.

3.1 Motivation and Overview

The main goal of classical reduction techniques is to make multi-class problems amenable to binary classification. Therefore, techniques such as one-vs-rest, all-pairs [6], ECOC [7], and nested dichotomies [2,5] reduce the original problem in such a way that *all* induced subproblems are binary. As opposed to this, the distinguishing feature of reduction stumps is their partiality. In fact, reduction stumps have a different use case and aim at *distributing* smaller and presumably easier subproblems among two multi-class classifiers. For instance, this can be advantageous because the multi-class classifier, despite being principally able to treat an arbitrary number of classes, does not scale well for larger numbers of classes. Furthermore, it might be interesting to consider different types of base learners at the inner nodes, as one classifier might be able to separate one group of classes well, while another one may prove beneficial for separating the remaining classes.

In contrast to previous approaches, we allow different classifiers to be used as base learners in reduction stumps. The flexibility of adopting different classifiers for (different parts of) the same problem is precisely one of the supposed strengths of reduction stumps. We call reduction stumps with different classifiers *heterogeneous*. Analogously, reduction stumps with the same classifier for all subproblems are called *homogeneous*. One reason for considering homogeneous reduction stumps may be that only one classification algorithm is available; for example, in medical data analysis, often only decision trees are accepted for reasons of interpretability.

In addition to single reduction stumps, we also consider ensembles. It has been observed that ensembling often improves predictive accuracy, provided there is a reasonable variety among the classifiers within the ensemble. For reduction stumps, there are two sources of variety, which makes them especially

suitable for being used in an ensemble. First, there is randomness in the choice of the splits of a reduction stump, which can be averaged out using ensembles; for the same reason, it is common to consider ensembles of nested dichotomies instead of only single dichotomies [5,8]. Second, the topology of a reduction stump strongly depends on the choice of the base learner. Hence, the ensemble effect might even be amplified when heterogeneous reduction stumps are used.

3.2 Training a Reduction Stump and Obtaining Predictions

Training a reduction stump requires two decisions. First, one must choose a split of the set of classes, i.e., decide which classes belong to the left and which to the right child of the root node. Second, one must choose the classifiers for (i) the binary split and (ii) the rest problems associated with the left and the right child, respectively. Obviously, there is an interaction between these two choices.

Even though the number of reduction stumps is *much* smaller than the number of nested dichotomies, it is still infeasible to enumerate all possible reduction stumps. Given a problem with n classes, there are $2^{n-1} - 1$ different splits. We *may* enumerate these candidates for small n, but we cannot reasonably build a general algorithm on this basis. However, there is reason to believe that near-optimal splits can be found without extensive search. First, a number of heuristics for finding good splits has been proposed [8]. Second, it has been shown that even randomly sampling a relatively small number of splits and taking the best of them often yields at least as good results [9].

Moreover, it is possible to conduct a grid-search over the possible classifiers for the binary problem and the two child nodes, i.e., to enumerate all options for the second choice. A full nested dichotomy has $n-1$ inner nodes, where n is the number of classes. If m classification algorithms are eligible, this leads to m^{n-1} possible combinations of classifiers over the inner nodes, which is generally not feasible. For the reduction stump, however, we have at most m^3 many combinations[1]. For example, we shall consider $m = 23$ classifiers, which leads to over 10^{12} combinations in a full nested dichotomy for a 10-class problem, whereas we only have 12,167 possible combinations for a reduction stump.

As a consequence, we design the reduction stump as a meta-classifier that can be parametrized with a *set* of base classifiers. Given such a set C of classifiers, it iterates over all active classifier combinations $R \subseteq C^3$, where $R = C^3$ in the heterogeneous case and $R = \{(c, c, c) \mid c \in C\}$ in the homogeneous case. For each classifier combination $r \in R$, depending on the split technique, one or more splits are identified and evaluated. The algorithm then associates the classifier combination r with the best determined split $s(r)$ and the respective validation score $t(r)$. The stump eventually selects $\operatorname{argmin}_{r \in R} t(r)$ as the classifier combination and uses $s(r)$ as the split.

The split is computed by drawing splits (uniformly) at random and evaluating them against a validation set. To this end, the set of classes is organized in a

[1] If the split separates a single class from the rest, then we even have only m^2 many combinations.

shuffled list, and a position within that list is drawn uniformly at random. We draw k such splits and, for each of them, we also split the given training data into a reduced training set and a validation set. The reduced training set is used to train the reduction stump, and the validation set is used to validate it. We do neither conduct cross-validation nor a holdout method here, but only use single evaluations. First, conducting a sophisticated evaluation would be costly and further reduce the data available for training the stump. Second, since the sampling routine itself considers different splits of both data and classes, an averaging effect over different data is still achieved.

We also experimented with another interesting split technique called *random pairs*. The random-pair selection heuristic (RPND) was proposed by Leathart et al. [8] for the construction of nested dichotomies and suggests to build the two subsets by randomly choosing two classes as "seeds", training a classifier to separate them, and adding each of the other classes to the seed to which most of its instances are assigned by the trained classifier. Our experiments showed that RPND does not perform significantly better than the above best-of-k random sampling, so we do not include it in the experiments for space reasons; this also conforms to the observations of [9]. However, the results are available with the implementation.

The inference for reduction stumps is straight forward. Given a new instance, the root classifier decides whether the instance should belong to the first or the second subset of classes. Based on this decision, the respective classifier for the subproblem is used to make a final decision for the class of the instance. For probabilistic predictions, each of the three classifiers computes a probability for the instance to belong to its covered classes. The class distribution for an instance is computed by multiplying the class probabilities produced by the base learners at the child nodes with the probability for the respective child as obtained from the root node's base learner.

3.3 Ensembles of Reduction Stumps

As already said, in addition to single reduction stumps, we are interested in *ensembles* of such models. Yet, in this paper, we only consider ensembles of homogeneous reduction stumps. Although we conjecture that heterogeneous ensembles can be much more powerful, a training procedure needs to make more decisions and would have to be more sophisticated than the straight forward approach as described below. To reduce the computational cost, an effective heuristic would be required. Since this work is more concerned with the fundamental question of whether a reduction is beneficial at all, designing such a heuristic is left for future work.

For ensembles of homogeneous reduction stumps, the training method is quite straight forward. Given a set C of available base classifiers, the algorithm iterates over all $c \in C$, using c as the base learner at each inner node. The algorithm constructs an ensemble of a given size by creating the structure of the reduction stumps at random. Furthermore, instead of performing a best-of-k selection for each ensemble member, we apply the best-of-k heuristic to the entire ensemble.

That is, we build k ensembles of reduction stumps and select the ensemble with the best score. For computing the score, the training data is again split into a reduced training set and a validation set, and the ensemble is trained on the former and evaluated on the latter. This evaluation procedure is repeated several times to obtain a stable estimate for the ensemble (which is not changed over the iterations), resulting in a holdout validation of the ensembles. After having selected all $|C|$ ensembles, the algorithm chooses the one with the best score to be used for future predictions and trains it on the entire training data.

The prediction routine for ensembles is implemented as a majority vote. Each reduction stump votes for a class, and the prediction of the ensemble is the class that collects the highest number of votes.

4 Experimental Evaluation

In our experimental evaluation, we compare the proposed (ensembles of) reduction stumps to (ensembles of) single classifiers to analyze the potential benefit of reducing the original problem to a set of simpler problems (with fewer classes). Recalling our motivation of reduction as a possible means to improve automated machine learning, note that, as part of an AutoML toolbox, reduction stumps would serve as an *option* rather than a *default choice*. Correspondingly, including them in the toolbox seems warranted if they provide the best choice in a *sufficient* portion—but not necessarily the *majority*—of the cases.

We subdivide our analysis into two main aspects. First, we carry out a detailed analysis of (ensembles of) homogeneous reduction stumps, comparing them to the single classifier and a bagged ensemble of the latter. Second, we additionally consider heterogeneous reduction stumps, comparing the best models using any classifier as a base learner.

4.1 Experimental Setup

In total, we evaluate the four methods on 21 datasets (as shown in Table 1) from different domains, including image recognition, biology, and audio. To estimate the predictive accuracy of each method, we used a 20-holdout (also known as Monte-Carlo cross-validation), splitting the data into 70% training data and 30% test data. As learning algorithms, which were also used as base learners for the reduction stumps, we considered BayesNet (BN), NaiveBayes (NB), NaiveBayesMultinomial (NBM), Logistic (L), MultilayerPerceptron (MP), SimpleLogistic (SL), SMO (SMO), IBk (IB), KStar (KS), JRip (JR), PART (PART), DecisionStump (DS), J48 (J48), LMT (LMT), RandomForest (RF), and RandomTree (RT).

To build ensembles of reduction stumps, we used the Best-of-k strategy. To this end, we used another internal stratified split of 70% data for building the reduction stumps and 30% validation data for selection, and set $k = 10$.

We have implemented both reduction stumps and ensembles as WEKA classifiers. The code, the data used to conduct the experiments, and the database

Table 1. Datasets used in the evaluation

Dataset	#instances	#attributes	#classes
audiology	226	69	24
autoUnivau6750	750	40	8
car	1728	6	4
cnae9	1080	856	9
fbis.wc	2463	2000	17
kropt	28056	6	18
letter	20000	16	26
mfeat-factors	2000	216	10
mfeat-fourier	2000	76	10
mfeat-karhunen	2000	64	10
mfeat-pixel	2000	240	10
optdigits	5620	64	10
pendigits	10992	16	10
page-blocks	5473	10	5
segment	2310	19	7
semeion	1593	256	10
vowel	990	13	11
waveform	5000	40	3
winequality	4898	11	11
yeast	1484	8	10
zoo	101	17	7

with the presented results are publicly available[2]. The computations were executed on (up to) 150 Linux machines in parallel, each of which with a resource limitation of 2 cores (Intel Xeon E5-2670, 2.6 Ghz) and 16 GB memory. The total run-time was over 30 k CPU hours (more than 3 years).

4.2 Analysis of Homogeneous Reduction Stumps

Table 2 shows the error rate averaged over 20 train/test splits of single classifiers (SC), homogeneous reduction stumps (RS), bagged ensembles of classifiers (BA), and majority vote ensembles of homogeneous reduction stumps (EN) for different classifiers and datasets. In the last column of the table, a statistic of wins/ties/losses (W/T/L) is provided comparing RS to SC and EN to BA. Missing values indicate that the respective algorithm was either not applicable to the problem or that it did not finish in a given timeout of 1h.

[2] https://github.com/fmohr/ML-Plan/tree/ida2018.

Considering the results for RS, we can indeed see that the performance of *every* classifier can sometimes be increased when wrapped into a reduction stump. Although RS is not an overall dominating strategy, except for SL and LMT, there are at least 3 datasets for each classifier where a reduction stump performs better than the single classifier. This indicates that reduction in principle can be beneficial in terms of performance improvement.

Comparing BA to EN, for some datasets, the overall picture is quite similar to the comparison of RS and SC. Neglecting the classifiers BN and NBM, there are at least 8 datasets for which EN yields a better performance. In particular, EN seems to yield improved results for DS, LMT, RT, and RF rather frequently. For 13 of 16 classifiers, EN wins more often than it loses against BA. We conclude that reduction stumps might be more suitable for being used in ensembles.

4.3 Analysis of Heterogeneous Reduction Stumps

In the second part of our evaluation, we consider heterogeneous reduction stumps. Since we cannot compare heterogeneous stumps in the context of a single base classifier, we now consider the overall best performance achieved with *any* classifier of the respective class. From the perspective of AutoML, this is the most interesting part of the evaluation, because it answers the questions whether (ensembles of) reduction stumps can be superior to *any* other classification algorithm (either by itself or used within a (bagging) ensemble).

The results are summarized in Table 3, where we now distinguish between RS-hom for homogeneous and RS-het for heterogeneous reduction stumps. Note that with EN we still only refer to ensembles of homogeneous reduction stumps. Significant differences are determined using a t-test with $p = 0.05$. While significant improvements of reduction stumps over the baseline (single classifier respectively bagged ensemble) are indicated by $\bullet$, significant degradations are indicated by $\circ$. Best performances within one row are highlighted in bold. Results that are not significantly worse than the best result are underlined.

Regarding RS-hom, in this table, it becomes even clearer that homogeneous reduction stumps do not perform that strong and in this context never achieve the best performance for any of the datasets. Nevertheless, if used in an ensemble, the homogeneous reduction stumps yield the best result in 5 cases. Furthermore, compared to BA, EN achieves 9 significant improvements. A significant degradation, in turn, is observed only once.

However, the most remarkable observations are made for heterogeneous reduction stumps that clearly outperform the other approaches. RS-het yields 6 significant improvements over SC while being significantly worse in only two cases. Furthermore, it turns out to achieve the best performance in 15 of 21 cases. From these results, we conclude that reduction stumps represent an interesting approach for decomposing multi-class classification problems to a set of simpler subproblems.

The results also motivate the investigation of ensembles of heterogeneous reduction stumps. On one hand, it would be interesting to design a heuristic for building such ensembles as a standalone classifier. On the other hand, instead

Table 2. Mean error rate of base learners on 21 UCI datasets. (see Footnote 2)

		audiology	autoUnivau6750	car	cnae9	kropt	letter	mfeat-factors	mfeat-fourier	mfeat-karhunen	mfeat-pixel	optdigits	page-blocks	pendigits	segment	semeion	vowel	waveform	winequality	yeast	zoo	fbis.wc	
BN	SC	**27.2**	77.1	13.5	**11.7**	63.7	26.0	7.6	23.5	8.5	6.5	8.1	6.6	**12.1**	**8.6**	14.4	42.3	20.0	52.4	42.1	5.7	29.7	
	RS	30.3	**75.8**	**12.0**	12.8	**63.4**	27.8	-	-	-	-	-	6.3	13.8	9.2	-	41.4	**18.9**	**49.7**	43.2	7.1	-	7/0/7
	BA	29.3	77.0	14.0	12.1	63.8	**25.8**	7.4	23.2	7.4	6.3	8.0	**5.9**	12.3	8.8	14.0	**37.1**	19.5	50.9	**40.9**	6.1	28.6	
	EN	28.8	77.4	13.8	12.1	63.9	26.1	-	-	-	-	-	6.3	12.6	8.7	-	38.0	22.9	52.0	41.2	**4.5**	-	4/1/9
NB	SC	**28.8**	80.1	13.7	8.4	63.7	35.9	8.1	**23.5**	6.5	6.8	9.1	10.7	**14.2**	19.6	13.8	37.6	19.9	56.3	**42.1**	4.8	37.7	
	RS	31.4	**79.5**	**12.4**	9.8	**63.7**	38.1	9.9	24.5	8.2	8.7	10.5	**6.4**	15.8	**18.6**	15.4	40.4	**17.6**	**51.2**	42.8	6.3	38.9	6/1/14
	BA	33.1	80.7	14.4	7.9	63.9	**35.8**	7.7	23.7	**6.4**	**6.5**	9.2	10.2	14.3	19.2	**13.6**	**36.6**	19.8	55.2	42.9	5.7	**32.7**	
	EN	30.7	79.7	13.9	**7.6**	64.0	36.2	**7.6**	24.1	6.8	7.2	**9.0**	10.8	14.3	19.8	14.0	38.0	22.2	55.7	42.3	**3.9**	37.5	8/1/12
NBM	SC	-	-	-	5.9	-	**44.7**	15.4	25.3	-	-	**9.4**	8.7	**19.4**	-	**15.4**	-	-	60.1	66.3	-	23.3	
	RS	-	-	-	6.7	-	45.8	21.3	**24.7**	-	-	12.0	**8.0**	22.3	-	17.3	-	-	**56.2**	66.4	-	25.0	3/0/8
	BA	-	-	-	6.8	-	44.8	**15.1**	26.0	-	-	9.4	8.2	19.5	-	15.5	-	-	59.4	**66.2**	-	**22.2**	
	EN	-	-	-	**5.6**	-	45.3	15.4	25.3	-	-	9.5	8.5	19.6	-	15.9	-	-	60.2	66.5	-	23.1	2/0/9
L	SC	22.6	81.6	6.3	8.3	-	22.6	2.8	29.0	6.9	-	6.7	3.4	**4.5**	4.9	29.8	20.0	**13.1**	46.1	40.7	10.9	-	
	RS	22.4	-	-	7.7	-	24.7	3.3	25.8	8.4	-	6.4	3.2	5.1	-	17.6	23.4	16.5	**45.9**	-	-	-	7/0/6
	BA	25.7	84.1	6.4	7.6	59.7	22.7	2.2	24.8	**5.2**	-	**3.6**	**3.0**	4.5	4.6	**9.7**	21.9	13.1	46.3	41.3	12.2	-	
	EN	**19.7**	80.7	6.1	**6.6**	-	-	**2.1**	**21.9**	5.3	5.8	3.7	3.4	4.7	4.5	12.3	**15.3**	16.1	46.1	40.9	5.6	-	11/0/6
MP	SC	**18.1**	83.9	1.1	-	37.3	17.7	2.5	17.6	4.1	-	1.8	3.7	5.6	3.8	7.8	9.1	16.6	44.0	40.9	4.8	-	
	RS	22.4	-	-	-	-	17.9	-	18.3	5.2	-	1.9	3.3	**1.6**	-	-	5.7	16.9	44.5	-	-	-	3/0/7
	BA	22.8	82.1	1.8	29.3	33.8	14.3	2.5	17.8	4.1	0.0	1.9	**3.2**	4.9	3.7	7.3	9.1	**13.8**	**42.6**	39.4	6.1	0.0	
	EN	20.1	81.7	0.4	-	-	-	-	**16.7**	**3.5**	-	**1.4**	3.3	1.8	3.0	-	**3.6**	16.5	44.1	39.7	4.2	-	10/0/4
SMO	SC	**21.2**	80.3	**6.6**	7.2	56.3	18.4	2.4	**15.9**	3.8	2.6	1.7	7.2	**2.2**	7.6	6.7	32.8	**13.6**	47.4	43.1	4.4	22.1	
	RS	22.4	**79.3**	6.8	8.6	57.3	23.0	2.6	15.9	4.1	3.1	2.1	**6.6**	2.6	7.1	7.3	34.7	16.0	**47.4**	44.0	4.3	23.0	4/2/15
	BA	28.1	81.4	7.0	7.3	**56.1**	19.0	**2.3**	16.9	3.6	2.6	**1.7**	6.8	2.3	8.3	**6.7**	38.9	13.8	47.6	44.4	6.1	23.4	
	EN	21.3	79.6	7.0	**6.7**	56.9	-	2.4	16.7	**3.5**	**2.5**	2.1	6.8	4.3	**7.1**	7.3	**28.6**	16.1	47.5	**43.0**	**3.8**	**20.5**	12/2/6
SL	SC	16.4	77.2	6.7	8.0	**59.4**	22.7	2.7	18.0	4.6	4.4	3.1	3.4	4.7	5.1	9.3	19.8	13.0	**46.2**	40.6	7.0	18.2	
	RS	21.3	-	-	9.2	60.1	24.7	2.9	18.8	5.1	4.7	3.2	**3.3**	6.3	-	9.5	20.9	16.1	46.3	-	-	18.2	1/1/14
	BA	19.3	81.7	7.0	8.1	59.7	22.8	**2.2**	18.1	3.9	3.9	3.0	3.4	**4.6**	5.2	**7.7**	19.5	**13.0**	46.4	41.2	9.1	**15.2**	
	EN	-	78.8	6.8	**7.2**	59.6	-	2.2	**17.3**	**3.7**	**3.6**	**2.8**	3.5	5.1	4.9	8.4	**16.1**	16.0	46.4	40.9	6.2	15.6	12/2/5
IB	SC	22.2	84.3	8.3	16.6	**29.7**	4.6	4.4	20.8	3.8	3.9	1.5	4.1	0.8	**3.4**	10.0	1.8	26.4	**37.6**	48.5	3.9	50.3	
	RS	**20.7**	84.1	**7.2**	17.7	29.8	4.8	4.5	20.4	**3.7**	3.8	1.4	**3.8**	0.8	3.5	9.5	**1.7**	26.6	37.8	49.5	4.6	51.0	10/1/10
	BA	25.0	84.4	8.0	**15.8**	32.5	4.8	**3.8**	**19.8**	4.0	**3.8**	1.5	3.8	0.8	4.1	9.8	4.7	**24.7**	39.6	**47.0**	5.2	49.1	
	EN	23.4	**83.6**	7.3	16.2	32.3	-	4.2	20.7	4.1	4.2	**1.4**	4.0	**0.7**	3.4	**9.2**	2.0	26.8	37.9	48.0	**3.7**	**49.1**	11/1/8
KS	SC	**19.1**	84.7	12.4	11.4	41.3	4.8	84.1	25.2	6.4	3.9	2.4	3.3	**0.9**	3.1	9.7	2.8	27.3	37.3	47.2	3.5	-	
	RS	19.3	-	-	12.5	**41.1**	-	-	23.9	6.5	**3.8**	2.5	**2.9**	1.0	-	9.0	2.6	27.6	**37.3**	-	-	-	6/1/6
	BA	21.6	85.8	13.1	**11.1**	42.0	4.8	83.9	24.9	**6.3**	3.8	2.3	3.2	1.0	3.2	9.4	4.7	24.4	39.1	46.7	4.3	62.4	
	EN	21.3	84.8	12.5	11.2	41.2	-	-	-	6.4	4.1	**2.2**	3.1	**0.9**	3.3	**8.7**	**2.5**	-	37.6	46.9	4.0	-	11/0/5
JR	SC	31.0	73.4	15.1	15.3	61.5	15.8	13.9	27.6	20.5	15.5	10.0	3.0	4.3	5.7	29.0	33.1	21.0	46.7	42.5	12.2	25.6	
	RS	31.0	74.0	**11.6**	14.9	48.7	15.5	12.4	28.8	20.4	15.2	9.0	2.8	4.3	5.6	25.7	28.1	20.9	46.2	42.9	10.9	25.7	15/2/4
	BA	27.8	78.1	13.2	**10.1**	**46.8**	8.4	5.7	21.6	9.4	6.5	3.9	**2.5**	1.8	**3.2**	14.5	20.0	**16.0**	**42.0**	41.1	**9.6**	20.6	
	EN	-	**72.7**	12.4	11.6	58.7	-	**4.6**	**19.5**	**7.6**	**4.9**	**3.4**	2.8	**1.4**	3.5	**12.6**	**15.5**	19.7	45.2	**40.7**	10.8	**20.3**	12/0/7
PART	SC	24.1	82.2	4.3	13.1	48.8	12.4	11.4	26.4	16.2	17.3	8.0	3.1	3.4	4.1	22.7	23.8	22.6	43.1	46.0	5.7	26.1	
	RS	24.1	83.6	**3.8**	13.8	43.8	13.1	11.8	26.6	18.3	14.9	8.0	2.7	3.6	4.5	21.7	26.0	22.3	42.6	45.5	9.0	25.8	9/2/10
	BA	22.2	80.8	4.0	**10.9**	41.7	7.8	6.4	20.6	10.1	10.9	4.2	**2.4**	1.9	3.5	14.6	17.6	**16.3**	**37.5**	**40.0**	7.0	**20.0**	
	EN	**21.6**	**79.6**	4.2	11.6	**38.1**	-	**5.9**	**19.4**	**8.2**	**6.7**	**3.1**	2.7	**1.5**	**3.0**	**11.4**	**13.7**	21.9	38.9	41.5	**5.5**	20.3	13/0/7
DS	SC	50.0	78.0	**29.4**	79.6	81.1	92.9	80.2	80.4	80.5	80.8	80.2	6.6	79.6	71.5	80.2	82.4	43.1	55.0	59.0	30.4	63.4	
	RS	**44.8**	**76.2**	29.4	70.5	**76.4**	88.2	67.8	65.8	68.1	66.4	68.5	**5.3**	63.1	46.6	69.7	75.5	**34.5**	**49.4**	**54.3**	**17.9**	**51.2**	20/1/0
	BA	50.0	77.4	29.4	65.2	78.5	90.4	63.9	80.5	79.9	79.0	76.2	6.6	79.5	43.6	79.8	72.1	42.0	55.0	59.3	25.7	63.4	
	EN	46.2	77.5	**29.4**	**50.3**	78.5	**82.1**	**48.8**	**60.5**	**61.5**	**56.7**	**55.9**	5.8	**55.4**	**34.9**	**64.0**	**66.3**	36.6	55.0	57.0	19.7	63.5	16/3/2
J48	SC	22.9	83.3	9.5	11.8	46.5	13.3	12.0	25.5	19.0	23.0	10.2	3.0	3.9	3.5	27.0	22.2	24.8	43.3	45.8	6.1	26.9	
	RS	22.4	82.2	**8.2**	12.4	45.1	13.7	12.3	27.9	21.3	20.2	9.3	2.8	4.1	4.1	25.1	25.4	24.5	43.2	44.7	8.2	27.4	11/0/10
	BA	**21.2**	80.5	8.3	**10.9**	43.9	9.0	7.1	21.1	10.4	17.0	5.1	**2.6**	2.4	3.5	17.7	16.4	**17.6**	**38.2**	**40.0**	7.0	**21.4**	
	EN	21.5	**78.2**	8.4	11.2	**43.6**	**6.5**	**6.3**	**20.3**	**10.0**	**12.7**	**3.7**	2.9	**1.8**	**3.2**	**13.2**	**15.1**	24.6	39.9	42.1	**5.7**	23.6	13/0/8
LMT	SC	16.4	77.1	3.5	8.0	-	9.1	2.7	18.2	4.6	4.4	2.8	3.0	1.6	3.6	9.3	7.9	**13.0**	43.6	40.6	7.0	-	
	RS	20.7	-	-	9.2	-	-	3.2	17.9	5.4	5.4	2.9	2.7	1.7	-	10.1	12.4	16.0	44.9	-	-	-	2/0/11
	BA	19.3	82.5	3.9	8.1	32.2	6.1	2.2	17.4	3.9	3.9	2.9	**2.6**	1.2	3.1	**7.7**	8.1	14.3	**37.8**	41.3	9.1	15.1	
	EN	-	78.7	2.9	**7.1**	29.4	3.9	**2.2**	**16.5**	**3.6**	**3.6**	**1.9**	2.8	**0.9**	2.8	7.9	**5.2**	16.0	39.2	40.5	6.3	-	14/1/4
RF	SC	22.8	74.6	6.2	6.6	32.8	4.2	3.7	**16.2**	4.4	4.4	2.0	2.5	1.0	**2.2**	6.7	3.9	14.8	32.3	39.6	**3.9**	23.6	
	RS	22.4	75.2	**4.7**	7.2	29.8	4.4	3.8	17.5	4.4	4.1	2.1	2.4	1.0	2.6	6.4	4.2	14.8	**32.3**	39.8	5.2	**22.3**	7/4/10
	BA	23.6	73.5	6.6	7.8	33.4	4.8	3.8	17.3	4.0	4.2	2.1	**2.4**	1.2	2.8	6.6	6.7	**14.2**	33.5	**38.4**	7.0	26.1	
	EN	**21.9**	**73.3**	5.4	**6.2**	-	**3.8**	**3.4**	16.6	**3.8**	**3.4**	**1.8**	2.5	**1.0**	2.2	**5.9**	**3.3**	14.8	32.5	38.5	4.0	23.0	17/0/3
RT	SC	34.0	85.2	16.6	25.8	52.0	15.4	16.6	37.0	30.4	36.5	14.3	3.8	4.7	5.0	34.5	20.2	27.6	40.3	49.8	27.8	56.8	
	RS	36.6	83.9	13.6	24.6	49.7	16.1	16.9	35.9	30.6	34.7	15.4	3.6	5.2	5.1	35.3	19.9	27.6	39.8	49.3	23.9	53.7	12/1/8
	BA	27.6	81.5	**7.2**	11.5	39.6	7.6	6.1	**21.5**	10.4	11.1	4.2	**2.9**	1.7	3.1	16.0	9.7	**18.2**	**36.7**	**42.5**	**8.3**	34.6	
	EN	**26.4**	**80.9**	10.9	**11.3**	**37.4**	**5.7**	**5.6**	21.6	**9.1**	**10.4**	**3.7**	3.1	**1.4**	**3.0**	**12.5**	**7.1**	24.9	37.6	43.8	22.5	**29.8**	14/0/7

Table 3. Averaged error rate (mean±standard deviation) using best base learners.

Dataset	SC	RS-hom	RS-het	BA	EN
audiology	16.38±3.01	19.31±3.99	**14.22±0.75**	19.31±1.86	19.68±0.96
autoUnivau6750	73.4±2.77	74.02±2.17	**70.05±0.24** •	73.54±1.75	72.72±0.38
car	1.13±0.58	3.84±0.75 ○	**0.10±0.10** •	1.75±0.70	0.42±0.12 •
cnae9	5.90±1.16	5.80±1.20	**4.60±0.16** •	6.83±1.46	5.61±0.53 •
fbis.wc	18.19±1.05	18.23±1.93	19.99±1.19	**15.13±1.25**	15.62±0.40
kropt	29.74±0.38	29.77±0.59	30.17±0.11 ○	32.22±0.46	**29.36±0.43** •
letter	4.22±0.28	4.35±0.19	4.13±0.07	4.80±0.22	**3.84±0.07** •
mfeat-factors	2.41±0.50	2.60±0.62	**1.95±0.42**	2.20±0.28	2.09±0.20
mfeat-fourier	15.88±1.21	16.84±1.42	**15.76±0.51**	16.93±1.11	16.48±0.52
mfeat-karhunen	3.76±0.34	3.81±0.42	**2.97±0.25**	3.61±0.57	3.46±0.24
mfeat-pixel	2.59±0.47	2.88±0.39	**2.46±0.25**	2.58±0.48	2.48±0.11
optdigits	1.51±0.26	1.50±0.35	**1.11±0.15**	1.50±0.26	1.38±0.08
page-blocks	2.51±0.38	2.44±0.25	**2.00±0.15** •	2.37±0.31	2.45±0.09
pendigits	0.77±0.13	0.75±0.12	0.73±0.18	0.84±0.16	**0.70±0.03** •
segment	2.16±0.49	2.58±0.48 ○	**2.04±0.44**	2.76±0.65	2.19±0.19 •
semeion	6.65±0.96	6.52±0.92	6.77±0.10	6.61±0.64	**5.92±0.28** •
vowel	1.78±0.98	2.20±1.23	**1.05±0.55**	4.69±2.23	2.02±0.23 •
waveform	13.04±0.78	14.82±0.56 ○	13.08±0.17	**13.01±0.58**	14.75±0.31 ○
winequality	32.29±1.48	32.54±1.09	**31.12±0.71**	33.53±1.12	32.49±0.30 •
yeast	39.58±2.09	39.79±2.75	**36.46±0.81** •	38.38±2.25	38.51±0.63
zoo	3.48±2.61	4.35±4.86	**0.00±0.00** •	4.35±3.37	3.67±0.68

of only choosing classifiers from a portfolio, the results motivate to actively decompose multi-class problems in the context of AutoML, and further tailoring the base learners to the respective subproblems. From an AutoML perspective, homogeneous reduction stumps are less attractive, as they seem to never achieve globally the best performance; thus, the effort for considering them would not be justifiable. In contrast to this, it is worth considering ensembles of reduction stumps, which in some cases perform best, especially since the effort for building them is relatively low.

5 Conclusion

In this paper, we proposed a meta-classifier called reduction stump, which can be seen as the simplest reduction scheme for multi-class classification problems: the original problem is decomposed into three subproblems of smaller size, two multi-class problems on subsets of the original set of classes, and one binary problem on the two respective meta-classes. In spite of their simplicity, reduction stumps show promising performance in our experiments, especially in their heterogeneous version.

Our main motivation for analyzing reduction stumps originates from the field of automated machine learning. For the reasons already explained, reduction can

be useful in AutoML, but should be applied with caution, in order to keep the complexity manageable. Encouraged by the results of this paper, our next step is to incorporate reduction stumps into the toolbox of AutoML. An additional interesting research question that arises from the observation that some datasets are more amenable to reduction than others is whether one can predict the benefit of applying reduction based on the properties of a dataset.

Acknowledgements. This work was partially supported by the German Research Foundation (DFG) within the Collaborative Research Center "On-The-Fly Computing" (SFB 901).

References

1. Dheeru, D., Karra Taniskidou, E.: UCI machine learning repository (2017). http://archive.ics.uci.edu/ml
2. Dong, Lin, Frank, Eibe, Kramer, Stefan: Ensembles of balanced nested dichotomies for multi-class problems. In: Jorge, Alípio Mário, Torgo, Luís, Brazdil, Pavel, Camacho, Rui, Gama, João (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 84–95. Springer, Heidelberg (2005). https://doi.org/10.1007/11564126_13
3. Duarte-Villaseñor, Miriam Mónica, Carrasco-Ochoa, Jesús Ariel, Martínez-Trinidad, José Francisco, Flores-Garrido, Marisol: Nested dichotomies based on clustering. In: Alvarez, Luis, Mejail, Marta, Gomez, Luis, Jacobo, Julio (eds.) CIARP 2012. LNCS, vol. 7441, pp. 162–169. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33275-3_20
4. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J.T., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, Quebec, Canada, 7–12 December 2015, pp. 2962–2970 (2015)
5. Frank, E., Kramer, S.: Ensembles of nested dichotomies for multi-class problems. In: Proceedings ICML, 21st International Conference on Machine Learning. Banff, Alberta, Canada (2004)
6. Fürnkranz, J.: Round robin classification. J. Mach. Learn. Res. **2**, 721–747 (2002). http://www.jmlr.org/papers/v2/fuernkranz02a.html
7. Kajdanowicz, T., Kazienko, P.: Multi-label classification using error correcting output codes. Appl. Math. Comput. Sci. **22**(4), 829–840 (2012). http://www.degruyter.com/view/j/amcs.2012.22.issue-4/v10006-012-0061-2/v10006-012-0061-2.xml
8. Leathart, Tim, Pfahringer, Bernhard, Frank, Eibe: Building Ensembles of adaptive nested dichotomies with random-pair selection. In: Frasconi, Paolo, Landwehr, Niels, Manco, Giuseppe, Vreeken, Jilles (eds.) ECML PKDD 2016. LNCS (LNAI), vol. 9852, pp. 179–194. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46227-1_12
9. Melnikov, V., Hüllermeier, E.: On the effectiveness of heuristics for learning nested dichotomies: an empirical analysis. Mach. Learn. **107**(8), 1537–1560 (2018)
10. Mohr, F., Wever, M., Hüllermeier, E.: Ml-Plan: automated machine learning via hierarchical planning. Mach. Learn. **107**(8), 1495–1515 (2018)

11. Olson, R.S., Moore, J.H.: TPOT: A tree-based pipeline optimization tool for automating machine learning. In: Proceedings of the 2016 Workshop on Automatic Machine Learning, AutoML 2016, Co-located with 33rd International Conference on Machine Learning (ICML 2016), New York City, NY, USA, 24 June 2016, pp. 66–74 (2016)
12. Thornton, C., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In: The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, 11–14 August 2013, pp. 847–855 (2013)
13. Wever, M., Mohr, F., Hüllermeier, E.: Ensembles of evolved nested dichotomies. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2018, Kyoto, Germany, 15–19 July 2018 (2018)