Machine Learning, 54, 255–273, 2004

Is Combining Classifiers with Stacking Better than Selecting the Best One?

SASO DŽEROSKI saso.dzeroski@ijs.si
BERNARD ŽENKO bernard.zenko@ijs.si
Department of Knowledge Technologies, Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia

Editors: Christophe Giraud-Carrier, Ricardo Vilalta and Pavel Brazdil

Abstract. We empirically evaluate several state-of-the-art methods for constructing ensembles of heterogeneous classifiers with stacking and show that they perform (at best) comparably to selecting the best classifier from the ensemble by cross validation. Among state-of-the-art stacking methods, stacking with probability distributions and multi-response linear regression performs best. We propose two extensions of this method, one using an extended set of meta-level features and the other using multi-response model trees to learn at the meta-level. We show that the latter extension performs better than existing stacking approaches and better than selecting the best classifier by cross validation.

Keywords: multi-response model trees, stacking, combining classifiers, ensembles of classifiers, meta-learning

1. Introduction

An ensemble of classifiers is a set of classifiers whose individual predictions are combined in some way (typically by voting) to classify new examples. One of the most active areas of research in supervised learning has been to study methods for constructing good ensembles of classifiers (Dietterich, 1997). The attraction that this topic exerts on machine learning researchers is based on the premise that ensembles are often much more accurate than the individual classifiers that make them up (Dietterich, 1997; Gams, Bohanec, & Cestnik, 1994).

Most of the research on classifier ensembles is concerned with generating ensembles by using a single learning algorithm (Dietterich, 2000), such as decision tree learning or neural network training. Different classifiers are generated by manipulating the training set (as done in boosting or bagging), manipulating the input features, manipulating the output targets or injecting randomness in the learning algorithm. The generated classifiers are then typically combined by majority or weighted voting.

Another approach is to generate classifiers by applying different learning algorithms (with heterogeneous model representations) to a single dataset (see, e.g., Merz 1999). More complicated methods for combining classifiers are typically used in this setting. Stacking (Wolpert, 1992) is often used to learn a combining method in addition to the ensemble of classifiers. Voting is then used as a baseline method for combining classifiers against which the learned combiners are compared. Typically, much better

performance is achieved by stacking as compared to voting (see, e.g., Todorovski & Džeroski, 2002).

The work presented in this paper is set in the stacking framework. We argue that selecting the best of the classifiers in an ensemble generated by applying different learning algorithms should be considered as a baseline to which the stacking performance should be compared. Our empirical evaluation of several recent stacking approaches shows that they perform comparably to the best of the individual classifiers as selected by cross validation, but not better.

The best among state-of-the-art methods is stacking with probability distributions (PDs) and multi-response linear regression (MLR) (Ting & Witten, 1999). We propose two extensions of this method, one using an extended set of meta-level features and the other using multi-response model trees to learn at the meta-level. We show that the latter extension performs better than existing stacking approaches and selecting the best classifier by cross validation.

To place our work in a wider context, note that combining classifiers with stacking can be considered as meta-learning. Literally, meta-learning means learning about learning. In practice, meta-learning takes as input results produced by learning and generalizes over them. The following meta-learning tasks have been considered within the machine learning community: learning to select an appropriate learner, learning to dynamically select an appropriate bias, and learning to combine predictions of base-level classifiers. Below we briefly describe each of these.

- Learning how to select the most appropriate learner. The appropriateness of a learner for a given domain is assessed according to some criterion, which is usually predictive acuracy. The general idea is that we describe each domain by a set of meta-features that are relevant to the performance of learning algorithms. The descriptions of several domains in terms of these meta-features, together with algorithm performances on these domains, constitute a meta-domain to which a meta-learner is applied. The resulting meta-classifier should be able to recommend the most appropriate learner for a new domain.
- Learning how to dynamically select a bias for a learning algorithm. The goal here is to construct a learning algorithm that would be able to modify its hypothesis space in order to have better coverage of the domain at hand.
- Learning how to combine the predictions of base-level classifiers. The predictions of base-level classifiers (or some properties thereof), together with the correct class values constitute a meta-level dataset. This is the type of meta-learning addressed in this paper.

For more details on meta-learning, we refer the reader to the excellent review of different aspects of meta-learning by Vilalta and Drissi (2002).

The remainder of this paper is organized as follows. Section 2 first summarizes the stacking framework, then describes stacking with probability distributions and multi-response linear regression, and finally surveys some other recent stacking approaches and results. Section 3 introduces our two extensions to stacking with PDs and MLR: the use of an extended set of meta-level features and classification via model trees at the meta-level. The setup for the experimental comparison of several stacking methods, voting and selecting the

best classifier is described in Section 4. Section 5 presents and discusses the experimental results and Section 6 concludes.

2. Stacking: State-of-the-art

We first give a brief introduction to the stacking framework, introduced by Wolpert (1992). We then describe the stacking approach proposed by Ting and Witten (1999) and review several recent studies in stacking (Merz, 1999; Ting & Witten, 1999; Todorovski & Džeroski, 2000; Seewald, 2002; Todorovski & Džeroski, 2002).

2.1. *The stacking framework*

Stacking is concerned with combining multiple classifiers generated by using different learning algorithms $L_1, \ldots, L_N$ on a single dataset S, which consists of examples $s_i = (x_i, y_i)$, i.e., pairs of feature vectors (x_i) and their classifications (y_i). In the first phase, a set of base-level classifiers $C_1, C_2, \ldots, C_N$ is generated, where $C_i = L_i(S)$. In the second phase, a meta-level classifier is learned that combines the outputs of the base-level classifiers.

To generate a training set for learning the meta-level classifier, a leave-one-out or a cross validation procedure is applied. For leave-one-out, we apply each of the base-level learning algorithms to almost the entire dataset, leaving one example for testing: $\forall i = 1, \ldots, n : \forall k = 1, \ldots, N : C_k^i = L_k(S - s_i)$. We then use the learned classifiers to generate predictions for s_i: $\hat{y}_i^k = C_k^i(x_i)$. The meta-level dataset consists of examples of the form $((\hat{y}_i^1, \ldots, \hat{y}_i^n), y_i)$, where the features are the predictions of the base-level classifiers and the class is the correct class of the example at hand. When performing, say, 10-fold cross validation, instead of leaving out one example at a time, subsets of size one-tenth of the original dataset are left out and the predictions of the learned base-level classifiers obtained on these.

In contrast to stacking, no learning takes place at the meta-level when combining classifiers by a voting scheme (such as plurality, probabilistic or weighted voting). The voting scheme remains the same for all different training sets and sets of learning algorithms (or base-level classifiers). The simplest voting scheme is the plurality vote. According to this voting scheme, each base-level classifier casts a vote for its prediction. The example is classified in the class that collects the most votes.

2.2. *Stacking with probability distributions and multi-response linear regression*

Ting and Witten (1999) stack base-level classifiers whose predictions are probability distributions (PDs) over the set of class values, rather than single class values. The meta-level attributes are thus the probabilities of each of the class values returned by each of the base-level classifiers. The authors argue that this allows them to use not only the predictions, but also the confidence of the base-level classifiers.

Each base-level classifier predicts a PD over the possible class values. The prediction of the base-level classifier C applied to example x is a PD:

$$\mathbf{p}^C(x) = (p^C(c_1 \mid x), p^C(c_2 \mid x), \ldots p^C(c_m \mid x)),$$

where $\{c_1, c_2, \ldots c_m\}$ is the set of possible class values and $p^C(c_i \mid x)$ denotes the probability that example x belongs to class c_i as estimated (and predicted) by classifier C. The class c_j with the highest class probability $p^C(c_j \mid x)$ is predicted by classifier C. The meta-level attributes (Ting & Witten, 1999) are the probabilities predicted for each possible class by each of the base-level classifiers, i.e.,

$$p^{C_j}(c_i \mid x)$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, N$.

Multi-response linear regression (MLR) is recommended for meta-level learning, while several other learning algorithms are shown not to be suitable for this task (Ting & Witten, 1999). MLR is an adaptation of linear regression. For a classification problem with m class values $\{c_1, c_2, \ldots, c_m\}$, m regression problems are formulated: for each class c_j, a linear equation LR_j is constructed to predict a binary variable, which has value one if the class value is c_j and zero otherwise. Given a new example x to classify, $LR_j(x)$ is calculated for all j, and the class k is predicted with maximum $LR_k(x)$.

Seewald (2002) recently proposed that MLR should use different sets of meta-level attributes for each of the m binary prediction problems. Only the probabilities of class c_j predicted by the different classifiers, i.e., $p^{C_i}(c_j \mid x)$ for $i = 1, \ldots, N$, are used to construct equation LR_j. Each of the meta-level learning problems thus has N instead of mN attributes. Improved performance is reported over stacking with the full set of probability distributions.

2.3. Other recent advances in stacking

The most important issues in stacking are probably the choice of the features and the algorithm for learning at the meta-level. Stacking with PDs and MLRs addresses both. Below we review some other recent research on stacking that addresses these issues.

It is common knowledge that ensembles of diverse base-level classifiers (with weakly correlated predictions) yield good performance. Merz (1999) proposes a stacking method called SCANN that uses correspondence analysis to detect correlations between the predictions of base-level classifiers. The original meta-level feature space (the class-value predictions) is transformed to remove these correlations, and a nearest neighbor method is used as the meta-level classifier on this new feature space.

Todorovski and Džeroski (2000) introduce a new meta-level learning method for combining classifiers with stacking: meta decision trees (MDTs) have base-level classifiers in the leaves, instead of class-value predictions. Properties of the probability distributions predicted by the base-level classifiers (such as entropy and maximum probability) are used as meta-level attributes, rather than the distributions themselves. These properties reflect the confidence of the base-level classifiers and give rise to very small MDTs, which can (at least in principle) be inspected and interpreted.

Todorovski and Džeroski (2002) report that stacking with MDTs clearly outperforms voting and stacking with decision trees, as well as boosting and bagging of decision trees. On the other hand, MDTs perform only slightly better than SCANN and selecting the best classifier with cross validation (SelectBest). Ženko et al. (2001) report that MDTs perform slightly worse as compared to stacking with MLR. Overall, SCANN, MDTs, stacking with MLR and SelectBest seem to perform at about the same level.

It would seem natural to expect that ensembles of classifiers induced by stacking would perform better than the best individual base-level classifier: otherwise the extra work of learning a meta-level classifier doesn't seem justified. The experimental results mentioned above, however, do not show clear evidence of this. This has motivated us to investigate the performance of state-of-the-art stacking methods in comparison to SelectBest and seek new stacking methods that would be clearly superior to SelectBest.

3. Extending stacking with MLR

The experimental evidence mentioned above indicates that although SCANN, MDTs, stacking with MLR and SelectBest seem to perform at about the same level, stacking with MLR has a slight advantage over the other methods. It would thus seem as a suitable starting point in the search for better stacking approaches. Here we propose two extensions of stacking with MLR, one along the dimension of meta-level features and the other along the dimension of meta-level learning algorithms.

3.1. An extended set of meta-level features

Recall that Ting and Witten (1999) propose to use as meta-level features the probabilities predicted for each possible class by each of the base-level classifiers, i.e.,

$$p^{C_j}(c_i \mid x)$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, N$. They propose the use of MLR as the meta-level learning algorithm.

In our first extension, we use MLR at the meta-level, just like Ting and Witten. We use the original set of $p^{C_j}(c_i \mid x)$ attributes, augmented with two additional sets of meta-level attributes:

- the probability distributions multiplied by the maximum probability

$$P_{C_j} = p^{C_j}(c_i \mid x) \times M_{C_j(x)} = p^{C_j}(c_i \mid x) \times \max_{i=1}^{m}(p^{C_j}(c_i \mid x))$$

  for $i = 1, \ldots, m$ and $j = 1, \ldots, N$ and
- the entropies of the probability distributions

$$E_{C_j}(x) = -\sum_{i=1}^{m} p_{C_j}(c_i \mid x) \cdot \log_2 p_{C_j}(c_i \mid x).$$

The total number of meta-level attributes in this approach is thus $N(2m + 1)$.

The motivation for considering these two additional sets of meta-level attributes is as follows. Already Ting and Witten (1999) state that the use of probability distributions has the advantage of capturing not only the predictions of the base-level classifiers, but also their certainty. The attributes we have added try to capture the certainty of the predictions more explicitly. Note that the attributes M_{C_j} and E_{C_j} are the only (ordinary) attributes used in the construction of meta decision trees (MDTs) (Todorovski & Džeroski, 2000): they are responsible for the good performance of MDTs and useful for learning at the meta-level.

Entropy (as captured by the entropies E_{C_j}) is a measure of uncertainty. The higher the entropy of a predicted probability distribution, the lower the certainty of the prediction. The maximum probability in a predicted probability distribution M_{C_j} also contains information on the certainty of the prediction: a high value of M_{C_j} means we have a prediction with high certainty, and a low value of M_{C_j} means we have an uncertain prediction. The attributes P_{C_j} combine the the predictions themselves (the individual probabilities) and the certainty of the predictions (as measured by the maximal probabilities M_{C_j} in a predicted distribution). We have added this combination in the hope that it will be easier to use for the meta-level learning algorithm in combined form, as compared to the learning algorithm discovering that information on predictions and their certainty should be combined.

It should be noted here that we have performed preliminary experiments using only the attributes P_{C_j} and E_{C_j} (without the original probability distributions). The results of these experiments showed no significant improvement over using the original probability distributions only. Any potential performance improvements thus result from the synergy of using all three sets of attributes at the same time.

3.2. Stacking with multi-response model trees

In our second extension of the Ting and Witten (1999) approach, we keep the original set of meta-level features, i.e., the probability distributions predicted by the base-level classifiers, and consider an alternative to MLR for learning at the meta-level.

Stacking with MLR uses linear regression (LR) to perform classification. Model trees can be viewed as an extension of linear models to piece-wise linear modes, and model tree induction as an extension of LR. When considering an alternative for MLR at the meta-level, a natural direction to look into is thus the use of model trees. Model trees have been shown to perform better than MLR for classification via regression (Frank, et al., 1998).

Recall that MLR formulates a binary classification problem for each possible class value. For each class c_j, a linear equation LR_j is constructed to predict a binary variable. Given a new example x to classify, $LR_j(x)$ is calculated for all j, and the class k is predicted with maximum $LR_k(x)$.

In our approach, we use model tree induction instead of linear regression and keep everything else the same. Instead of m linear equations LR_j, we induce m model trees MT_j. Given a new example x to classify, $MT_j(x)$ is calculated for all j, and the class k is predicted with maximum $MT_k(x)$. We call our approach stacking with multi-response model trees, analogously to stacking with MLR.

4. Experimental setup

In the experiments, we investigate the following issues:

- The (relative) performance of existing state-of-the-art stacking methods, especially in comparison to SelectBest.
- The performance of stacking with an extended set of meta-level features relative to the above methods.
- The performance of stacking with multi-response model trees relative to the above methods.
- The influence of the number of base-level classifiers on the (relative) performance of the above methods.

We look into the last topic because the recent studies mentioned above use different numbers of base-level classifiers, ranging from three to eight. The Weka data mining suite (Witten & Frank, 1999) was used for all experiments, within which all the base-level and meta-level learning algorithms used in the experiments have been implemented. Ten-fold cross validation is used to construct the meta-level s for all combining methods.

4.1. Datasets

In order to evaluate the performance of the different combining algorithms, we perform experiments on a collection of thirty datasets from the *UCI Repository of machine learning databases* (Blake & Merz, 1998). These datasets have been widely used in other comparative studies. The datasets and their properties (number of examples, classes, (discrete/continuous) attributes, probability of the majority class, entropy of the class probability distribution) are listed in Table 1.

4.2. Base-level algorithms

We performed two batches of experiments: one with three and one with seven base-level learners. The set of three contains the following algorithms:

- J4.8: a Java re-implementation of the decision tree learning algorithm C4.5 (Quinlan, 1993),
- IBk: the k-nearest neighbor algorithm of Aha, Kibler, and Albert (1991) and
- NB: the naive Bayes algorithm of John and Langley (1995).

The second set of algorithms contains, in addition to the above three, also the following four algorithms:

- K*: an instance-based algorithm which uses an entropic distance measure (Cleary & Trigg, 1995),
- KDE: a simple kernel density estimation algorithm,

Table 1. The datasets used and their properties: number of examples, classes, (discrete/continuous) attributes, probability of the majority class, and entropy of the class probability distribution.

Dataset	EXS	CLS	(D/C)	ATT	MAJ	ENT
Australian	690	2	(8/6)	14	0.56	0.99
Balance	625	3	(0/4)	4	0.46	1.32
Breast-cancer	286	2	(9/0)	9	0.70	0.88
Breast-W	699	2	(9/0)	9	0.66	0.92
Bridges-TD	102	2	(4/3)	7	0.85	0.61
Car	1728	4	(6/0)	6	0.70	1.21
Chess	3196	2	(36/0)	36	0.52	0.99
Contraceptive	1473	3	(4/5)	9	0.43	1.54
Diabetes	768	2	(0/8)	8	0.65	0.93
Dis	3772	2	(22/6)	28	0.98	0.11
Echo	131	2	(1/5)	6	0.67	0.91
German	1000	2	(13/7)	20	0.70	0.88
Glass	214	6	(0/9)	9	0.36	2.18
Heart-C	303	5	(7/6)	13	0.54	0.99
Heart-H	294	5	(7/6)	13	0.64	0.94
Heart	270	2	(6/7)	13	0.56	0.99
Hepatitis	155	2	(13/6)	19	0.79	0.74
Hypo	3163	2	(18/7)	25	0.95	0.29
Image	2310	7	(0/19)	19	0.14	2.78
Ionosphere	351	2	(0/34)	34	0.64	0.94
Iris	150	3	(0/4)	4	0.33	1.58
Solar-flare-C	1389	8	(10/0)	10	0.84	0.88
Solar-flare-M	1389	6	(10/0)	10	0.95	0.34
Solar-flare-X	1389	3	(10/0)	10	0.99	0.75
Sonar	208	2	(0/60)	60	0.53	1.00
Soya	683	19	(35/0)	35	0.13	3.79
Tic-tac-toe	958	2	(9/0)	9	0.65	0.93
Vote	435	2	(16/0)	16	0.61	0.96
Waveform	5000	3	(0/21)	21	0.34	1.58
Wine	178	3	(0/13)	13	0.40	1.56

Dataset source: *UCI Repository of machine learning databases* (Blake & Merz, 1998).

- DT: the decision table majority algorithm of Kohavi (1995),
- MLR: the multi-response linear regression algorithm, as used by Ting and Witten (1999) and described in Section 2.3.

All algorithms are used with their default parameter settings, with the exception of NB algorithm which uses the kernel density estimator rather than assume normal distributions for numeric attributes.

4.3. Meta-level algorithms

At the meta-level, we evaluate the performance of six different schemes for combining classifiers (listed below), each applied with the two different sets of base-level algorithms described above.

- VOTE: The simple plurality vote scheme (see Section 2.1),
- SELB: The SelectBest scheme (called MultiScheme in Weka (Witten & Frank, 1999)) selects the best of the base-level classifiers by cross validation.
- SMDT: Stacking with meta decision-trees as introduced by Todorovski and Džeroski (2000) and briefly described in Section 2.3.
- SMLR: Stacking with multi-response linear regression (MLR) as used by Ting and Witten (1999) and described in Section 2.2.
- SCMLR: Stacking with MLR and a reduced set of attributes as used by Seewald (2002) and described in Section 2.2.
- SMLRE: Stacking with MLR and an extended set of meta-level attributes, as proposed by this paper and described in Section 3.1.
- SMM5: Stacking with multi-response model trees, as proposed by this paper and described in Section 3.2. M5′ (Wang & Witten, 1997), a re-implementation of M5 (Quinlan, 1992) included in the data mining suite Weka (Witten & Frank, 1999), is used to induce the model trees at the meta level.

4.4. Evaluating and comparing algorithms

This subsection describes several aspects of the evaluation of the different combining schemes, or to be more precise, ensembles of learned base-level classifiers and a (learned) combiner. The evaluation aspects include the estimation of error rates and pairwise comparisons of classifiers/ensembles.

4.4.1. Estimating error rates. The classification errors of the combining schemes are estimated using ten-fold stratified cross validation. Cross validation is repeated ten times using different random generator seeds resulting in ten different sets of folds. The same folds (random generator seeds) are used in all experiments. The classification error of a classification algorithm C for a given dataset as estimated by averaging over the ten runs of ten-fold cross validation is denoted $\mathsf{error}(C)$. For pair-wise comparisons of classification algorithms, we calculate the relative improvements and paired t-tests, as described below.

4.4.2. Relative improvement. In order to evaluate the accuracy improvement achieved in a given domain by using classifier/combiner C_1 as compared to using C_2, we calculate the relative improvement: $1-\mathsf{error}(C_1)/\mathsf{error}(C_2)$. The average relative improvement (ARI) across all domains is calculated using the geometric mean of error reduction in individual

domains: $1-\mathsf{geometric\_mean}(\mathsf{error}(C_1)/\mathsf{error}(C_2))$. Note that this comparison is not symmetric, i.e., ARI of C_1 over C_2 calculated as $1-\mathsf{geometric\_mean}(\mathsf{error}(C_1)/\mathsf{error}(C_2))$ may be different from the negative value of ARI of C_2 over C_1 calculated as $\mathsf{geometric\_mean}(\mathsf{error}(C_2)/\mathsf{error}(C_1)) - 1$.

The classification errors of C_1 and C_2 averaged over the ten runs of 10-fold cross validation are compared for each dataset ($\mathsf{error}(C_1)$ and $\mathsf{error}(C_2)$ refer to these averages).

4.4.3. Statistical significance. The statistical significance of the difference in performance is tested using a paired t-test (exactly the same folds are used for C_1 and C_2) with significance level of 95%: $+/-$ to the right of a figure in the tables with results means that the classifier/ensemble C_1 is significantly better/worse than C_2. Our earlier studies (Todorovski & Džeroski, 2002; Ženko, Todorovski, & Džeroski, 2001), as well as some other recent studies (e.g., (Seewald, 2002)) used the pairs of error rates corresponding to each of the ten repetitions of ten-fold cross validation. We refer to results of these tests as 10×10 t-test results.

However, in the ten repeats of ten-fold cross validation, we have overlapping test sets as well as training sets. This is likely to lead to an underestimate of the true variance of the algorithms, and therefore will tend to make Type I errors (report significant differences when there are none). We thus consider two other variants of the t-test, whose behavior for the purpose of comparing learning algorithms has been studied by Dietterich (1998).

In the first, the pairs of error rates are taken from the ten folds of a single ten-fold cross validation (1×10 t-test). While the training sets overlap, the test sets are independent, and the test is less likely to make Type I errors. In the second, five two-fold cross validations are performed, and the five pairs of error rates are taken to compare two learning algorithms (5×2 t-test). Here the train and test sets do not overlap, and this test is less likely still to make Type I errors. Unfortunately, it is also less sensitive (more likely not to detect differences in performance when they actually exist).

We have performed all three types of tests. We will focus our presentation on the 1×10 t-test results, since this represents a reasonable compromise between the probability of Type I error and sensitivity. We will mention results of the other two t-tests occasionally for illustrative purposes.

5. Experimental results

The error rates of the 3-classifier and 7-classifier ensembles induced as described above on the thirty datasets and combined with the different combining methods are given in Tables 6 and 7. However, for the purpose of comparing the performance of different combining methods, Tables 2 and 3 are of much more interest: they give the average relative improvement of X over Y for each pair of combining methods X and Y, as well as the number of significant wins:losses (according to the 1×10 t-test). Tables 4 and 5 present a more detailed comparison (per dataset) of SMM5 (which turns out to perform best) to the other combining methods. Below we highlight some of our more interesting findings.

Table 2. The relative performance of 3-classifier ensembles with different combining methods. The entry in row X and column Y gives the relative improvement of X over Y in % and the number of wins: losses (according to the 1×10 t-test).

	VOTE		SELB		SMDT		SMLR		SCMLR		SMLRE		SMM5		Total
VOTE			−20.28	2:7	−18.14	1:8	−22.71	1:8	−24.46	1:8	−18.70	0:9	−38.14	0:8	5:48
SELB	16.86	7:2			2.84	0:2	−2.02	1:2	−3.48	0:3	1.31	0:4	−14.85	0:4	8:17
SMDT	15.35	8:1	−1.81	2:0			−3.87	1:2	−5.35	1:2	−0.48	0:2	−16.93	0:4	12:11
SMLR	18.50	8:1	1.98	2:1	3.72	2:1			−1.43	1:1	3.26	0:3	−12.58	0:4	13:11
SCMLR	19.65	8:1	3.36	3:0	5.08	2:1	1.14	1:1			4.63	0:4	−10.99	0:4	14:11
SMLRE	15.76	9:0	−1.32	4:0	0.48	2:0	−3.37	3:0	−4.85	4:0			−16.37	0:5	22:5
SMM5	27.61	8:0	12.93	4:0	14.48	4:0	11.17	4:0	9.90	4:0	14.07	5:0			29:0

Table 3. The relative performance of 7-classifier ensembles with different combining methods. The entry in row X and column Y gives the relative improvement of X over Y in % and the number of wins: losses (according to the 1×10 t-test).

	VOTE		SELB		SMDT		SMLR		SCMLR		SMLRE		SMM5		Total
VOTE			−15.72	1:10	−14.85	1:8	−18.18	0:8	−21.68	0:8	−12.77	0:9	−30.98	0:7	2:50
SELB	13.55	10:1			0.74	0:2	−2.14	2:2	−5.16	2:1	2.53	2:2	−13.20	2:4	18:12
SMDT	18.22	7:1	−0.73	2:0			−2.92	2:1	−6.03	2:1	1.85	1:2	−13.96	1:4	15:9
SMLR	15.39	7:0	2.18	2:2	2.92	1:3			−2.89	1:3	4.62	0:3	−10.73	1:4	12:15
SCMLR	17.77	8:0	5.01	1:2	5.75	1:3	2.71	3:1			7.25	3:3	−7.49	2:3	18:12
SMLRE	11.53	9:0	−1.92	2:2	−1.24	2:1	−4.70	2:0	−7.60	2:3			−15.41	1:2	18:8
SMM5	23.57	7:0	11.66	3:2	12.33	4:1	9.66	4:1	7.06	3:2	13.84	2:1			23:7

5.1. State-of-the-art stacking methods

We first focus on the existing stacking methods SMDT, SMLR and SCMLR. Inspecting Table 2 (three base-level classifiers), we find that all of these combining approaches perform much better than VOTE and so does SELB. The difference in performance is quite high: 15% to 20% average relative improvement (ARI), 5 to 7 more wins than losses. While all three stacking methods perform better than SELB, the difference in performance is slight. The wins-loss difference ranges from 1 to 3, while the average relative improvement ranges from −2% to 3%.

When we move from three to seven base-level classifiers (Table 3), it still holds that the three stacking approaches and SELB perform much better than VOTE. However, the difference in performance between the stacking approaches and SELB can now hardly be characterized as positive. The wins:losses ratios wrt. SELB for SMDT, SMLR and SCMLR are 2:0, 2:2, and 1:2, and the ARI of SMDT over SELB is −1%.

In terms of ARI, SCMLR performs slightly better than SMLR, which in turn performs slightly better than SMDT. This holds for both three and seven base-level classifiers, with

the differences in performance being more pronounced in the latter case. However, the ARIs are not large (6% at most) and are not consistent with the wins:losses ratios/differences. In sum, the state-of-the-art stacking approaches perform comparably to each other and do not perform better than selecting the best classifier by cross validation.

5.2. *Extended set of meta-level attributes*

For three base-level classifiers, stacking with MLR and an extended set of meta-level attributes (SMLRE) performs better than the state-of-the-art stacking approaches and SELB in terms of the wins-loss difference. It has no significant losses and has two to four wins. It performs worse, however, in terms of the ARI: It only has a positive relative improvement over SMDT. For seven base-level classifiers, SMLRE has no positive ARI over any of the state-of-the-art stacking methods and has 2:1 and 2:0 wins:losses wrt. SMDT and SMLR. In sum, SMLRE does not perform clearly better than SELB or other state-of-the-art stacking methods.

5.3. *Multi-response model trees*

Returning to Table 2, this time paying attention to the relative performance of SMM5 to the other combining methods, we find that SMM5 is in a league of its own. It clearly outperforms all the other combining methods. It has no significant losses and at least 4 wins wrt. each other method. The ARI wrt. each other method is at least 10% (smallest when compared to SCMLR).

For seven base-level classifiers, SMM5 still has an ARI of over 7% over each other method. However, the wins − losses difference wrt. SELB, SCMLR and SMLRE is only one (according to the 1×10 t-test). On the other hand, according to the 5×2 t-test, the wins:losses ratios of SMM5 to SELB, SMDT, SMLR, and SMLRE are all 2:0. The wins:losses ratio to the closest competitor SCMLR is 4:0 according to the 5×2 t-test. In sum, SMM5 performs better than SELB and other state-of-the-art stacking methods.

5.4. *The influence of the number of base-level classifiers*

Studying Tables 2 and 3, we can note that VOTE remains the worst and SMM5 remains the best for both three and seven base-level classifiers. The relative performance of the other combining methods is affected by the change of the number of base-level classifiers. Looking at the total of wins − losses, SELB improves most in relative terms (from -9 for three to $+6$ for seven base-level classifiers), followed by SMDT (from $+1$ to $+6$) and SCMLR (from $+3$ to $+6$). On this metric, SMLR (from $+2$ to -3) and SMLRE (from $+17$ to $+10$) perform worse for a larger number of base-level classifiers.

It is interesting to note that the performance of the individual stacking schemes increases only slightly when we move from 3 to 7 base level classifiers (it increases most for voting and least for SMM5: the wins:losses ratio is 1:0 in the latter case). It is even more interesting to note that the performance of SMM5 with 3 base level classifiers is better than the performance

of the other stacking schemes with 7 base level classifiers. Its wins:losses ratios are almost the same as for SMM5 with 7 base level classifiers.

5.5. Discussion of experimental results

Most of the combining methods we consider are variants of stacking with MLR. Seewald (2002) presents empirical evidence that stacking with MLR (SMLR) performs worse on multi-class datasets (as compared to two-class datasets). He cites the dimensionality of the meta-data as a probable cause, and argues that the reduction of this dimensionality by reducing the set of meta-level features helps (making SCMLR perform better).

In our experiments, one way to increase the dimensionality of the meta-level data is to add more base-level classifiers. Note that the (relative) performance of SMLR decreases and that of SCMLR increases with the number of base-level classifiers. This is consistent with the argument of Seewald about the dimensionality of the meta-data.

Another way to increase dimensionality is to add new meta-level attributes as in SMLRE. With a small number of base-level classifiers the effect of providing additional information about the certainty of predictions prevails. With a larger number of base-level classifiers, however, this effect is countered by the adverse effect of the increase of dimensionality of the meta-data. SMLRE thus provides only limited advantage over SMLR.

It is not a surprise that stacking with multi-response model trees performs better than stacking with multi-response linear regression. The results of Frank et al. (1998), who investigate classification via regression, show that classification via model trees performs extremely well, i.e., better than multi-response linear regression and better than C5.0 (a successor of C4.5 (Quinlan, 1993)), especially in domains with continuous attributes. Given that the meta-level attributes are probabilities (and thus continuous), multi-response model trees are a very suitable choice for learning at the meta-level. This is confirmed by our experimental results.

SMM5 performs better than SMLR, but also than SCMLR. The advantage of SCMLR over SMLR derives from the reduction of the dimensionality problem, but note that some potentially useful information is thrown away (only the probabilities of one class value are kept). Model tree induction (M5′) within SMM5 is apparently able to handle the dimensionality problem well without throwing away, and indeed by making use of, this information. This is why SMM5 performs best among the stacking approaches studied, including SMLRE.

6. Conclusions

We have empirically evaluated several state-of-the-art methods for constructing ensembles of heterogeneous classifiers with stacking and shown that they perform (at best) comparably to selecting the best classifier from the ensemble by cross validation. We have proposed two new methods for stacking, extending stacking with probability distributions and multi-response linear regression: one uses an extended set of meta-level features, while the other using multi-response model trees to learn at the meta-level. We show that the latter extension performs better than existing stacking approaches and selecting the best classifier from the ensemble by cross validation.

Note that our approach is intended for combining classifiers that are heterogeneous (derived by different learning algorithms, using different model representations) and strong (i.e., each of the base-level classifiers performs relatively well in its own right), rather than homogeneous and weak. It is not intended, for example, for combining many classifiers derived by a single learning algorithm on subsamples of the original dataset. Given this, however, our experimental results indicate that stacking with multi-response model trees is a good choice for learning at the meta-level, regardless of the choice of the base-level classifiers.

While conducting this study and a few other recent studies (Ženko, Todorovski, & Džeroski, 2001; Todorovski & Džeroski, 2002), we have encountered quite a few contradictions between claims in the recent literature on stacking and our experimental results. These are, however, most likely due to differences in the experimental methodology used and possibly also the different collections of datasets considered. For example, while Merz (Merz, 1999) claims that SCANN is clearly better than the oracle selecting the best classifier (which should perform even better than SelectBest), Todorovski and Džeroski (2002) show that it performs comparably to SelectBest, but not better. This is probably due to the fact that Merz uses significance tests on repeated hold-out experiments, a methodology that is shown to be flawed (Dietterich, 1998), as it has a high probability of reporting significant differences in performance even if there are none. Our comparative study includes a large number of datasets and uses a carefully chosen experimental methodology: it thus provides a clearer picture of the relative performance of different stacking approaches.

Appendix: Tables with Experimental Results

Table 4. Relative improvement in accuracy (in %) of stacking with multi-response model trees (SMM5) as compared to other combining algorithms and its significance (+/− means significantly better/worse, '.' means insignificant) for 3 base-level classifiers.

Dataset	VOTE	SELB	SMDT	SMLR	SCMLR	SMLRE
Australian	−1.94 .	1.97 .	−1.01 .	−1.22 .	−1.22 .	8.03 .
Balance	65.75 +	45.85 +	45.85 +	54.80 +	51.44 +	30.51 +
Breast-cancer	−9.12 .	1.86 .	1.50 .	0.38 .	−0.64 .	−0.38 .
Breast-W	19.42 .	−3.72 .	−3.72 .	−2.09 .	−2.09 .	−8.33 .
Bridges-TD	6.41 .	9.32 .	10.98 .	−1.39 .	−1.39 .	0.68 .
Car	76.38 +	73.71 +	69.47 +	72.68 +	76.08 +	62.73 +
Chess	57.55 +	−1.57 .	−1.57 .	−1.57 .	−1.57 .	−1.57 .
Contraceptive	3.58 .	2.65 .	4.41 .	−0.34 .	−0.31 .	0.35 .
Diabetes	1.83 .	5.24 .	4.40 .	−0.27 .	−0.33 .	1.83 +
Dis	21.96 +	−9.52 .	−8.91 .	−8.01 .	−8.01 .	−8.61 .
Echo	10.65 .	−2.50 .	3.91 .	0.54 .	2.64 .	−1.65 .
German	−0.40 .	1.61 .	0.44 .	−1.58 .	−1.50 .	−0.77 .
Glass	−6.56 .	2.49 .	−0.30 .	−0.91 .	−5.55 .	9.26 .
Heart-C	10.65 .	−4.65 .	−4.43 .	7.13 .	−0.20 .	18.18 .

(*Continued on next page*.)

Table 4. (*Continued*).

Dataset	VOTE	SELB	SMDT	SMLR	SCMLR	SMLRE
Heart-H	9.20 .	−0.45 .	1.11 .	10.12 .	−2.54 .	18.98 .
Heart	12.20 .	−0.70 .	−0.70 .	−1.65 .	−3.35 .	14.12 .
Hepatitis	14.75 .	3.66 .	7.06 .	3.27 .	5.20 .	6.69 .
Hypo	44.29 +	−4.37 .	43.09 .	−4.37 .	−4.37 .	−4.37 .
Image	4.72 .	1.97 .	1.52 .	2.12 .	−0.47 .	−0.00 .
Ionosphere	−9.56 .	6.78 .	8.94 .	−6.59 .	−6.59 .	−5.36 .
Iris	0.00 .	8.33 .	8.33 .	7.04 .	4.35 .	10.81 .
Solar-flare-C	1.07 .	−1.88 .	−1.88 .	−0.50 .	−1.88 .	1.60 .
Solar-flare-M	0.98 .	−3.68 .	−2.47 .	1.81 .	−3.68 .	5.37 .
Solar-flare-X	12.95 .	0.00 .	0.00 .	3.20 .	−0.83 .	8.33 .
Sonar	21.11 .	−1.07 .	5.96 .	−1.07 .	−1.07 .	1.73 .
Soya	5.02 .	11.76 .	10.68 .	9.94 .	5.64 .	11.59 .
Tic-tac-toe	97.18 +	72.83 +	72.83 +	55.35 +	55.35 +	81.20 +
Vote	50.33 +	3.25 .	3.25 .	3.25 .	3.25 .	3.87 .
Waveform	19.72 +	23.17 +	16.17 +	12.86 +	12.86 +	6.44 +
Wine	−19.35 .	28.85 .	28.85 .	27.45 .	24.49 .	37.29 .
Average	27.61	12.93	14.48	11.17	9.90	14.07
Win/Lose	8+/0−	4+/0−	4+/0−	4+/0−	4+/0−	5+/0−

Table 5. Relative improvement in accuracy (in %) of stacking with multi-response model trees (SMM5) as compared to other combining algorithms and its significance (+/− means significantly better/worse, '.' means insignificant) for 7 base-level classifiers.

Dataset	VOTE	SELB	SMDT	SMLR	SCMLR	SMLRE
Australian	−2.38 .	3.52 .	4.91 .	−2.49 .	−2.07 .	16.27 .
Balance	49.84 +	40.00 +	39.77 +	51.60 +	48.29 +	11.91 .
Breast-cancer	−8.68 .	0.25 .	0.99 .	−0.25 .	−0.12 .	1.96 .
Breast-W	25.49 .	−1.06 .	−1.06 .	0.00 .	0.00 .	−4.97 .
Bridges-TD	−2.55 .	3.59 .	8.52 .	−1.26 .	−1.90 .	−0.63 .
Car	81.51 +	78.13 +	66.56 +	70.63 +	75.12 +	64.64 +
Chess	60.90 +	−4.19 −	−4.19 −	−4.19 −	−4.19 −	−4.19 −
Contraceptive	3.56 .	0.83 .	4.33 +	−1.40 .	−1.70 .	0.25 .
Diabetes	0.32 .	−3.94 −	1.02 .	−1.37 .	−1.77 −	−0.05 .
Dis	19.16 .	−11.26 .	−10.66 .	−7.71 .	−8.29 .	−4.65 .
Echo	1.73 .	−6.13 .	0.25 .	−2.84 .	−8.74 .	−0.51 .
German	3.49 .	5.80 .	4.32 .	−0.17 .	−0.48 .	1.86 .
Glass	2.36 .	−0.00 .	1.46 .	−2.28 .	−10.91 .	17.58 .
Heart-C	11.05 .	−1.66 .	−1.24 .	11.21 .	−1.03 .	17.48 .

(*Continued on next page.*)

Table 5. (*Continued*).

Dataset	VOTE	SELB	SMDT	SMLR	SCMLR	SMLRE
Heart-H	3.28 .	−11.58 .	−12.65 .	3.28 .	−5.59 .	5.98 .
Heart	7.08 .	0.69 .	3.56 .	0.00 .	−0.46 .	9.41 .
Hepatitis	−0.39 .	−0.39 .	1.92 .	2.30 .	4.85 .	8.27 .
Hypo	49.29 +	−4.60 .	41.45 .	−2.88 .	−2.88 .	−3.73 .
Image	−7.43 .	31.76 .	16.02 .	−2.14 .	−6.47 .	−5.77 .
Ionosphere	11.71 .	11.71 .	14.29 .	−5.60 .	−6.02 .	1.86 .
Iris	4.00 .	0.00 .	0.00 .	2.70 .	−5.88 .	22.58 .
Solar-flare-C	−0.45 .	−2.65 .	−2.60 .	1.06 .	−2.79 .	3.85 .
Solar-flare-M	2.86 .	−4.09 .	−2.89 .	1.25 .	−4.39 .	7.28 .
Solar-flare-X	10.07 .	−2.46 .	−2.46 .	−0.81 .	−4.17 .	8.09 .
Sonar	0.00 .	−3.92 .	−2.91 .	−5.30 .	−7.07 .	12.88 .
Soya	4.49 .	−2.03 .	0.66 .	11.50 .	6.03 .	13.83 .
Tic-tac-toe	93.00 +	73.91 +	73.91 +	60.65 +	48.93 .	80.80 +
Vote	41.91 +	7.60 .	7.60 .	3.07 .	2.47 .	8.67 .
Waveform	16.68 +	3.10 .	1.34 .	11.95 +	14.23 +	5.56 .
Wine	−34.62 .	14.63 .	10.26 .	5.41 .	2.78 .	12.50 .
Average	23.57	11.66	12.33	9.66	7.06	13.84
Win/Lose	7+/0−	3+/2−	4+/1−	4+/1−	3+/2−	2+/1−

Table 6. Error rates (in %) of the learned ensembles of classifiers for 3 base-level classifiers.

Dataset	VOTE	SELB	SMDT	SMLR	SCMLR	SMLRE	SMM5
Australian	14.17	14.74	14.30	14.28	14.28	15.71	14.45
Balance	13.41	8.48	8.48	10.16	9.46	6.61	4.59
Breast-cancer	25.31	28.15	28.04	27.73	27.45	27.52	27.62
Breast-W	3.46	2.69	2.69	2.73	2.73	2.58	2.79
Bridges-TD	15.29	15.78	16.08	14.12	14.12	14.41	14.31
Car	6.49	5.83	5.02	5.61	6.41	4.11	1.53
Chess	1.43	0.60	0.60	0.60	0.60	0.60	0.61
Contraceptive	47.78	47.33	48.19	45.91	45.93	46.23	46.07
Diabetes	24.22	25.09	24.87	23.71	23.70	24.22	23.78
Dis	1.33	0.95	0.95	0.96	0.96	0.95	1.04
Echo	31.53	27.48	29.31	28.32	28.93	27.71	28.17
German	24.92	25.43	25.13	24.63	24.65	24.83	25.02
Glass	29.21	31.92	31.03	30.84	29.49	34.30	31.12
Heart-C	18.28	15.61	15.64	17.59	16.30	19.97	16.34
Heart-H	16.63	15.03	15.27	16.80	14.73	18.64	15.10

(*Continued on next page*.)

Table 6. (*Continued*).

Dataset	VOTE	SELB	SMDT	SMLR	SCMLR	SMLRE	SMM5
Heart	18.22	15.89	15.89	15.74	15.48	18.63	16.00
Hepatitis	17.94	15.87	16.45	15.81	16.13	16.39	15.29
Hypo	1.36	0.72	1.33	0.72	0.72	0.72	0.76
Image	2.94	2.85	2.84	2.86	2.78	2.80	2.80
Ionosphere	7.15	8.40	8.60	7.35	7.35	7.44	7.83
Iris	4.40	4.80	4.80	4.73	4.60	4.93	4.40
Solar-flare-C	16.16	15.69	15.69	15.91	15.69	16.25	15.99
Solar-flare-M	5.13	4.90	4.95	5.17	4.90	5.36	5.08
Solar-flare-X	1.00	0.87	0.87	0.90	0.86	0.95	0.87
Sonar	17.31	13.51	14.52	13.51	13.51	13.89	13.65
Soya	6.71	7.22	7.13	7.07	6.75	7.20	6.37
Tic-tac-toe	9.24	0.96	0.96	0.58	0.58	1.39	0.26
Vote	6.90	3.54	3.54	3.54	3.54	3.56	3.43
Waveform	18.42	19.24	17.64	16.97	16.97	15.80	14.78
Wine	1.74	2.92	2.92	2.87	2.75	3.31	2.08
Average	13.60	12.75	12.79	12.59	12.41	12.90	12.07

Table 7. Error rates (in %) of the learned ensembles of classifiers for 7 base-level classifiers.

Dataset	VOTE	SELB	SMDT	SMLR	SCMLR	SMLRE	SMM5
Australian	13.99	14.84	15.06	13.97	14.03	17.10	14.32
Balance	10.14	8.48	8.45	10.51	9.84	5.78	5.09
Breast-cancer	25.77	28.08	28.29	27.94	27.97	28.57	28.01
Breast-W	3.65	2.69	2.69	2.72	2.72	2.59	2.72
Bridges-TD	15.39	16.37	17.25	15.59	15.49	15.69	15.78
Car	6.73	5.69	3.72	4.24	5.00	3.52	1.24
Chess	1.59	0.60	0.60	0.60	0.60	0.60	0.62
Contraceptive	47.26	45.95	47.64	44.94	44.81	45.69	45.57
Diabetes	24.10	23.11	24.27	23.70	23.61	24.01	24.02
Dis	1.33	0.96	0.97	1.00	0.99	1.03	1.07
Echo	30.92	28.63	30.46	29.54	27.94	30.23	30.38
German	24.08	24.67	24.29	23.20	23.13	23.68	23.24
Glass	25.79	25.19	25.56	24.63	22.71	30.56	25.19
Heart-C	18.22	15.94	16.01	18.25	16.04	19.64	16.20
Heart-H	16.60	14.39	14.25	16.60	15.20	17.07	16.05
Heart	17.26	16.15	16.63	16.04	15.96	17.70	16.04
Hepatitis	16.39	16.39	16.77	16.84	17.29	17.94	16.45

(*Continued on next page*.)

Table 7. (*Continued*).

Dataset	VOTE	SELB	SMDT	SMLR	SCMLR	SMLRE	SMM5
Hypo	1.56	0.76	1.35	0.77	0.77	0.76	0.79
Image	1.92	3.03	2.46	2.02	1.94	1.95	2.06
Ionosphere	8.52	8.52	8.77	7.12	7.09	7.66	7.52
Iris	5.00	4.80	4.80	4.93	4.53	6.20	4.80
Solar-flare-C	16.10	15.75	15.76	16.34	15.73	16.82	16.17
Solar-flare-M	5.28	4.93	4.99	5.20	4.92	5.54	5.13
Solar-flare-X	1.00	0.88	0.88	0.89	0.86	0.98	0.90
Sonar	15.29	14.71	14.86	14.52	14.28	17.55	15.29
Soya	6.72	6.20	6.40	7.43	7.14	7.62	6.33
Tic-tac-toe	3.58	0.96	0.96	0.64	0.49	1.30	0.25
Vote	6.25	3.93	3.93	3.75	3.72	3.98	3.63
Waveform	16.64	14.04	13.89	15.60	15.79	14.47	13.55
Wine	1.46	2.30	2.19	2.08	2.02	2.25	1.97
Average	12.95	12.30	12.47	12.39	12.09	12.95	12.01

Bibliographic notes

This paper has it origins in three conference papers that propose and partly evaluate the two new approaches to stacking: stacking with multi-response model trees is introduced by Džeroski and Ženko (2002a, 2002b) and the extended set of meta-level attributes is introduced by Ženko and Džeroski (2002). However, this paper significantly extends and upgrades the work presented there. In particular: We consider both new approaches together and provide a comparison; We include another very recent advance in stacking (Seewald, 2002) in the comparison; We consider a larger collection of datasets (thirty as compared to twenty-one); We use a more carefully chosen experimental methodology (t-tests), which results in some changes in the conclusions; We provide a much more comprehensive discussion of the experimental results.

Acknowledgments

Many thanks to Ljupčo Todorovski for the cooperation on combining classifiers with meta-decision trees and the many interesting and stimulating discussions related to this paper. We acknowledge the support of the EU funded project METAL: ESPRIT IV No. 26.357.

References

Aha, D., Kibler, W. D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning, 6*, 37–66.
Blake, C. L., & Merz, C. J. (1998). UCI repository of machine learning databases.

Cleary, J. G., & Trigg, L. E. (1995). K*: An instance-based learner using an entropic distance measure. In *Proceedings of the 12th International Conference on Machine Learning* (pp. 108–114). San Francisco, Morgan Kaufmann.

Dietterich, T. G. (1997). Machine-learning research: Four current directions. *AI Magazine*, *18:4*, 97–136.

Dietterich, T. G. (1998). Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation, 10:7*, 1895–1923.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems* (pp. 1–15). Berlin: Springer.

Džeroski, S., & Ženko, B. (2002). Is combining classifiers better than selecting the best one? In *Proceedings of the Nineteenth International Conference on Machine Learning*, San Francisco: Morgan Kaufmann.

Džeroski, S., & Ženko, B. (2002). Stacking with multi-response model trees. In *Multiple Classifiers Systems, Proceedings of the Third International Workshop*, Berlin: Springer.

Frank, E., Wang, Y., Inglis, S., Holmes, G., & Witten, I. H. (1998). Using model trees for classification. *Machine Learning, 32:1*, 63–76.

Gams, M., Bohanec, M., & Cestnik, B. (1994). A schema for using multiple knowledge. In S. J. Hanson, T. Petsche, M. Kearns, & R. L. Rivest, editors, *Computational Learning Theory and Natural Learning Systems*, volume II (pp. 157–170). Cambridge, Massachusetts: MIT Press.

John, G. H., & Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence* (pp. 338–345). San Francisco, Morgan Kaufmann.

Kohavi, R. (1995). The power of decision tables. In *Proceedings of the Eighth European Conference on Machine Learning* (pp. 174–189).

Merz, C. J. (1999). Using correspondence analysis to combine classifiers. *Machine Learning, 36:1/2*, 33–58.

Quinlan, J. R. (1992). Learning with continuous classes. In *Proceedings of the Fifth Australian Joint Conference on Artificial Intelligence* (pp. 343–348). Singapore, World Scientific.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. San Francisco: Morgan Kaufmann.

Seewald, A. K. (2002). How to make stacking better and faster while also taking care of an unknown weakness. In *Proceedings of the Nineteenth International Conference on Machine Learning*, San Francisco: Morgan Kaufmann.

Ting, K. M., & Witten, I. H. (1999) Issues in stacked generalization. *Journal of Artificial Intelligence Research, 10*, 271–289.

Todorovski, L., & Džeroski, S. (2000). Combining multiple models with meta decision trees. In *Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery* (pp. 54–64). Berlin, Springer.

Todorovski, L., & Džeroski, S. (2002). Combining classifiers with meta decision trees. *Machine Learning, 50:3*, 223–249.

Vilalta, R., & Drissi, Y. (2002). A perspective view and survey of meta-learning. *Artificial Intelligence Review*, *18:2*, 77–95.

Wang, Y., & Witten, I. H. (1997). Induction of model trees for predicting continuous classes. In *Proceedings of the Poster Papers of the European Conference on Machine Learning*, Prague. University of Economics, Faculty of Informatics and Statistics.

Witten, I. H., & Frank, E. (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. San Francisco: Morgan Kaufmann.

Wolpert, D. (1992). Stacked generalization. *Neural Networks, 5:2*, 241–260.

Ženko, B., & Džeroski, S. (2002). Stacking with an extended set of meta-level attributes and MLR. In *Proceedings of the Thirteenth European Conference on Machine Learning*, Berlin: Springer.

Ženko, B., Todorovski, L., & Džeroski, S. (2001). A comparison of stacking with MDTs to bagging, boosting, and other stacking methods. In *Proceedings of the First IEEE International Conference on Data Mining* (pp. 669–670). Los Alamitos, IEEE Computer Society.

Received October 4, 2002
Revised October 4, 2002
Accepted March 11, 2003
Final manuscript April 24, 2003