Stacking with an Extended Set of Meta-Level Attributes and MLR

Bernard Ženko and Sašo Džeroski

Department of Intelligent Systems, Jožef Stefan Institute,
Jamova 39, SI-1000 Ljubljana, Slovenia,
Bernard.Zenko@ijs.si, Saso.Dzeroski@ijs.si

Abstract. We propose a new set of meta-level features to be used for learning how to combine classifier predictions with stacking. This set includes the probability distributions predicted by the base-level classifiers and a combination of these with the certainty of the predictions. We use these features in conjunction with multi-response linear regression (MLR) at the meta-level. We empirically evaluate the proposed approach in comparison to several state-of-the-art methods for constructing ensembles of heterogeneous classifiers with stacking. Our approach performs better than existing stacking approaches and also better than selecting the best classifier from the ensemble by cross validation (unlike existing stacking approaches, which at best perform comparably to it).

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 [3]. 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.

Most of the research on classifier ensembles is concerned with generating ensembles by using a single learning algorithm [5], 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 voting or weighted voting.

Another approach is to generate classifiers by applying different learning algorithms (with heterogeneous model representations) to a single data set (see, e.g., [8]). More complicated methods for combining classifiers are typically used in this setting. Stacking [15] 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.

The work presented in this paper is set in the stacking framework. We propose a new set of meta-level features. We use them in conjunction with multi-response linear regression at the meta-level, and show that this combination does perform better than other combining approaches. 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 approach we propose here performs better than selecting the best individual classifier.

Section 2 first summarizes the stacking framework, then surveys some recent results and finally introduces our stacking approach based on classification via linear regression. The setup for the experimental comparison of several stacking methods, voting and selecting the best classifier is described in Section 3. Section 4 presents and discusses the experimental results and Section 5 concludes.

2 Stacking

We first give a brief introduction to the stacking framework, introduced by Wolpert [15]. We then summarize the results of several recent studies in stacking [8, 11, 12, 10, 13]. Motivated by these, we introduce a modified stacking approach based on classification via linear regression [11].

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 data set 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 data set, 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 data set 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, ten-fold cross validation, instead of leaving out one example at a time, subsets of size one-tenth of the original data set are left out and the predictions of the learned classifiers obtained on these. We use ten-fold cross validation in all our experiments for generating the meta-level training set.

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 Recent Advances

The most important issues in stacking are probably the choice of the features and the algorithm for learning at the meta-level. Below we review some recent research on stacking that addresses the above issues.

It is common knowledge that ensembles of diverse base-level classifiers (with weakly correlated predictions) yield good performance. Merz [8] 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 the dependencies, and a nearest neighbor method is used as the meta-level classifier on this new feature space.

Ting and Witten [11] use base-level classifiers whose predictions are probability distributions 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 to use not only the predictions, but also the confidence of the base-level classifiers. Multi-response linear regression (MLR) is recommended for meta-level learning, while several learning algorithms are shown not to be suitable for this task.

Seewald and Fürnkranz [10] propose a method for combining classifiers called grading that learns a meta-level classifier for each base-level classifier. The meta-level classifier predicts whether the base-level classifier is to be trusted (i.e., whether its prediction will be correct). The base-level attributes are used also as meta-level attributes, while the meta-level class values are + (correct) and − (incorrect). Only the base-level classifiers that are predicted to be correct are taken and their predictions combined by summing up the probability distributions predicted.

Todorovski and Džeroski [12] 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 [13] 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. [16] 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 seek new stacking methods and investigate their performance relative to state-of-the-art stacking methods and SelectBest, in the hope of achieving performance that would be clearly superior to SelectBest.

2.3 Stacking with Multi-Response Linear Regression

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 method for meta-level learning to be used in stacking.

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 problem 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 for which $LR_k(x)$ is the highest.

In seeking to improve upon stacking with MLR, we have explored two possible directions that correspond to the major issues in stacking. Concerning the choice of the algorithm for learning at the meta-level, we have explored the use of model trees instead of LR [6]since model trees naturally extend LR to construct piecewise linear approximations. In this paper, we consider the choice of the meta-level features used for stacking.

2.4 An Extended Set of Meta-Level Features for Stacking

We assume that each base-level classifier predicts a probability distribution over the possible class values. Thus, the prediction of the base-level classifier C when applied to example x is a probability distribution:

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

where $\{c_1, c_2, \ldots c_m\}$ is the set of possible class values and $p^C(c_i|x)$ 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 $p^C(c_j|x)$ is predicted by classifier C.

The meta-level attributes as proposed by [11] are the probabilities predicted for each possible class by each of the base-level classifiers, i.e.,

$$p^{C_j}(c_i|x)$$

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

In our approach, we use two additional sets of meta-level attributes: probability distributions multiplied by maximum probability

$$P_{C_j} = p^{C_j}(c_i|x) \times \; M_C = p^{C_j}(c_i|x) \times \max_{i=1}^{m} \left(p^{C_j}(c_i|x)\right)$$

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

$$E_C = -\sum_{i=1}^{m} p_C(c_i|x) \cdot \log_2 p_C(c_i|x).$$

Therefore the total number of meta-level attributes in our approach is $N(2m+1)$.

The motivation for considering these additional meta-level attributes is as follows. Already Ting and Witten [11] 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 (the entropies E_C) and combine them with the predictions themselves (the products P_{C_j} of the individual probabilities and the maximal probabilities M_C in a predicted distribution). The attributes M_C and E_C have been used in the construction of meta decision trees [12]. It should be noted here that we have performed preliminary experiments using only the attributes P_{C_j} and E_C (without the original probability distributions). The results of these experiments showed no significant improvement over using the original probability distributions only. We can therefore conclude that the synergy of all three sets of attributes is responsible for the performance improvement achieved by our approach.

3 Experimental Setup

In the experiments, we investigate the performance of stacking with multi-response linear regression and the extended set of meta-level attributes. and in particular its relative performance as compared to existing state-of-the-art stacking methods and SelectBest.

The Weka data mining suite [14] was used for all experiments, within which all the base-level and meta-level learning algorithms used in the experiments have been implemented.

3.1 Data Sets

In order to evaluate the performance of the different combining algorithms, we perform experiments on a collection of twenty data sets from the *UCI Repository of machine learning databases* [2]. These data sets have been widely used in other comparative studies. The data sets 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.

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

Data set	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-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
diabetes	768	2	(0/8)	8	0.65	0.93
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	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
soya	683	19	(35/0)	35	0.13	3.79
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

3.2 Base-Level Algorithms

We use three different learning algorithms at the base level:

- J4.8: a Java re-implementation of the decision tree learning algorithm C4.5 [9],
- IBk: the k-nearest neighbor algorithm of [1], and
- NB: the naive Bayes algorithm of [7].

All algorithms are used with their default parameter settings, with the exceptions described below. IBk uses inverse distance weighting and k is selected with cross validation from the range of 1 to 77. The NB algorithm uses the kernel density estimator rather than assume normal distributions for numeric attributes. These settings were chosen in advance and were not tuned to our data sets.

3.3 Meta-Level Algorithms

At the meta-level, we evaluate the performance of six different schemes for combining classifiers (listed below).

Table 2. Error rates (in %) of the learned ensembles of classifiers.

Data set	Vote	Selb	Grad	Smdt	Smlr	Smlr-E
australian	13.81	13.78	14.04	13.77	14.16	13.93
balance	8.91	8.51	8.78	8.51	9.47	6.40
breast-w	3.46	2.69	3.69	2.69	2.73	2.58
bridges-td	15.78	15.78	15.10	16.08	14.12	14.80
car	6.49	5.83	6.10	5.02	5.61	4.11
chess	1.46	0.60	1.16	0.60	0.60	0.60
diabetes	24.01	25.09	24.26	24.74	23.78	24.51
echo	29.24	27.63	30.38	27.71	28.63	27.71
german	25.19	25.69	25.41	25.60	24.36	25.53
glass	29.67	32.06	30.75	31.78	30.93	31.64
heart	17.11	16.04	17.70	16.04	15.30	15.93
hepatitis	17.42	15.87	18.39	15.87	15.68	15.87
hypo	1.32	0.72	0.80	0.79	0.72	0.72
image	2.94	2.85	3.32	2.53	2.84	2.80
ionosphere	7.18	8.40	8.06	8.83	7.35	6.87
iris	4.20	4.73	4.40	4.73	4.47	4.87
soya	6.75	7.22	7.38	7.06	7.22	7.35
vote	7.10	3.54	5.22	3.54	3.54	3.59
waveform	15.90	14.42	17.04	14.40	14.33	13.61
wine	1.74	3.26	1.80	3.26	2.87	2.02
Average	11.98	11.74	12.19	11.68	11.44	11.27

- Vote: The simple plurality vote scheme (results of preliminary experiments showed that this performs better than the probability vote scheme).
- Selb: The SelectBest scheme selects the best of the base-level classifiers by ten-fold cross validation.
- Grad: Grading as introduced by Seewald and Fürnkranz [10] and briefly described in Section 2.2.
- Smdt: Stacking with meta decision-trees as introduced by Todorovski and Džeroski [12] and briefly described in Section 2.2.
- Smlr: Stacking with multiple-response regression as used by Ting and Witten [11] and described in Sections 2.2 and 2.3.
- Smlr-E: Stacking with multiple-response regression and extended set of meta-level attributes, as proposed by this paper and described in Section 2.3.

3.4 Evaluating and Comparing Algorithms

In all the experiments presented here, classification errors are estimated using ten-fold stratified cross validation. Cross validation is repeated ten times using

Table 3. Relative improvement in accuracy (in %) of stacking with multi-response linear regression (SMLR-E) as compared to other combining algorithms and its significance (+/– means significantly better/worse, x means insignificant).

DATA SET	VOTE	SELB	GRAD	SMDT	SMLR
AUSTRALIAN	-0.84 x	-1.05 x	0.83 x	-1.16 x	1.64 x
BALANCE	28.19 +	24.81 +	27.14 +	24.81 +	32.43 +
BREAST-W	25.62 +	4.26 +	30.23 +	4.26 +	5.76 +
BRIDGES-TD	6.21 x	6.21 x	1.95 x	7.93 x	-4.86 x
CAR	36.63 +	29.46 +	32.54 +	17.99 +	26.70 +
CHESS	59.10 +	0.00 x	48.66 +	0.00 x	0.00 x
DIABETES	-2.06 x	2.33 +	-1.02 x	0.95 x	-3.07 –
ECHO	5.22 x	-0.28 x	8.79 +	0.00 x	3.20 x
GERMAN	-1.35 x	0.62 x	-0.47 x	0.27 x	-4.80 –
GLASS	-6.61 –	1.31 x	-2.89 x	0.44 x	-2.27 x
HEART	6.93 +	0.69 x	10.04 +	0.69 x	-4.12 x
HEPATITIS	8.89 x	0.00 x	13.68 +	-0.00 x	-1.23 x
HYPO	45.35 +	0.00 x	9.13 +	8.77 x	0.00 x
IMAGE	4.57 x	1.82 x	15.54 +	-10.60 –	1.37 x
IONOSPHERE	4.37 x	18.31 +	14.84 +	22.26 +	6.59 +
IRIS	-15.87 –	-2.82 x	-10.61 x	-2.82 x	-8.96 x
SOYA	-8.89 –	-1.83 x	0.40 x	-4.15 x	-1.83 x
VOTE	49.51 +	-1.30 x	31.28 +	-1.30 x	-1.30 x
WAVEFORM	14.45 +	5.63 +	20.17 +	5.53 +	5.03 +
WINE	-16.13 x	37.93 +	-12.50 x	37.93 +	29.41 +
AVERAGE	15.24	7.11	13.40	6.37	4.76
W/L	8+/3–	7+/0–	12+/0–	6+/1–	6+/2–

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 data set as estimated by averaging over the ten runs of ten-fold cross validation is denoted with $\mathsf{error}(C)$.

For pair-wise comparisons of classification algorithms, we calculate the relative improvement and the paired t-test, as described below. In order to evaluate the accuracy improvement achieved in a given domain by using classifier C_1 as compared to using classifier C_2, we calculate the relative improvement: $1-\mathsf{error}(C_1)/\mathsf{error}(C_2)$. In Table 3, we compare the performance of SMLR-E to other approaches: C_1 in this table thus refers to ensembles combined with SMLR-E. The average relative improvement 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 may be different from $\mathsf{geometric\_mean}(\mathsf{error}(C_2)/\mathsf{error}(C_1))-1$.

Table 4. The relative performance of ensembles with different combining methods in terms of wins+/loses–. The entry in row X and column Y gives the number of wins+/loses– of X over Y.

	VOTE	SELB	GRAD	SMDT	SMLR	SMLR-E	TOTAL
VOTE	/	7+/9–	6+/4–	6+/10–	5+/10–	3+/8–	27+/41–
SELB	9+/7–	/	10+/3–	0+/2–	2+/4–	0+/7–	21+/23–
GRAD	4+/6–	3+/10–	/	1+/11–	2+/13–	0+/12–	10+/42–
SMDT	10+/6–	2+/0–	11+/1–	/	4+/4–	1+/6–	28+/17–
SMLR	10+/5–	4+/2–	13+/2–	4+/4–	/	2+/6–	33+/19–
SMLR-E	8+/3–	7+/0–	12+/0–	6+/1–	6+/2–	/	39+/6–

The classification errors of C_1 and C_2 averaged over the ten runs of ten-fold cross validation are compared for each data set ($\mathsf{error}(C_1)$ and $\mathsf{error}(C_2)$ refer to these averages). The statistical significance of the difference in performance is tested using the 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 C_1 is significantly better/worse than C_2.

At this place we have to say that we are fully aware of the weakness of our significance testing method described above. Namely, when we repeat ten-fold cross validation ten times we do not get ten independent accuracy assessments as required by the paired t-test. As a result we have a high risk of committing a type I error (incorrectly rejecting the null hypothesis). This means that it is likely that a smaller number of differences between classifiers are statistically significant than reported by our testing method. Due to this problem we have also tried using two significance testing methods proposed by Dietterich [4]: the ten-fold cross validated paired t-test and the 5x2cv paired t-test. The problem with these two tests is that while they have smaller probability of type I error they are much less sensitive. According to these two tests, the differences between the simplest approach (Vote scheme) and a current state-of-the-art approach (stacking with MLR) are hardly significant. Therefore we have decided to use the above described significance testing.

4 Experimental Results

The error rates of the ensembles induced on the twenty data sets and combined with the different combining methods are given in Table 2. However, for the purpose of comparing the performance of different combining methods, Table 4 is of much more interest: it gives the number of significant wins/loses of X over Y for each pair of combining methods X and Y. Table 3 presents a more detailed comparison (per data set) of SMLR-E to the other combining methods. Below we highlight some of our findings.

Inspecting Table 4, to examine the relative performance of SMLR-E to the other combining methods, we find that SMLR-E is in a league of its own. It clearly

outperforms all the other combining methods, with a wins – loss difference of at least 4 and a relative improvement of at least 5% (see Table 3). As expected, the difference is smallest when compared to SMLR.

Returning to Table 4, we find that we can partition the five existing combining algorithms into three groups. VOTE and GRAD are at the lower end of the performance scale, SELB and SMDT are in the middle, while SMLR performs best. While SMLR clearly outperforms VOTE and GRAD in one to one comparison, there is no difference when compared to SMDT (equal number of wins and losses).

None of the existing stacking methods perform clearly better than SELB. SMLR and SMDT have a slight advantage (two more wins than losses), while VOTE and GRAD perform worse. SMLR-E, on the other hand, clearly outperforms SELB with seven wins, no losses, and an average relative improvement of 7%.

5 Conclusions and Further Work

We have proposed a new set of meta-level features to be used for combining heterogeneous classifiers with stacking. These include the probability distributions predicted by the base-level classifiers, their certainty (entropy), and a combination of both (the products of the individual probabilities and the maximal probabilities in a predicted distribution). In conjunction with the multi-response linear regression (MLR) algorithm at the meta-level, this approach outperforms existing stacking approaches. While the existing approaches perform (at best) comparably to selecting the best classifier from the ensemble by cross validation, the proposed approach clearly performs better.

The use of the certainty features in addition to the probability distributions is obviously the key to the improved performance. A more detailed analysis of which of the new attributes are used and their relative importance is an immediate topic for further work. The same goes for the experimental evaluation of the proposed approach in a setting with seven base-level classifiers (as in [6]. Finally, combining the approach proposed here with that of Džeroski and Ženko [6] (i.e., using both a new set of meta-level features and a new meta-level learning algorithm) should also be investigated. Some more general topics for further work are discussed below: these have been also discussed by Džeroski and Ženko [6].

While conducting this study, the study of Džeroski and Ženko [6], and a few other recent studies [16, 13], we have encountered quite a few contradictions between claims in the recent literature on stacking and our experimental results. For example, Merz [8] claims that SCANN is clearly better than the oracle selecting the best classifier (which should perform even better than SelectBest). Ting and Witten [11] claim that stacking with MLR clearly outperforms SelectBest. Finally, Seewald and Fürnkranz [10] claim that both grading and stacking with MLR perform better than SelectBest. A comparative study including the data sets in the recent literature and a few other stacking methods (such as SCANN) should resolve these contradictions and provide a clearer picture of the relative

performance of different stacking approaches. We believe this is a worthwhile topic to pursue in near-term future work.

We also believe that further research on stacking in the context of base-level classifiers created by different learning algorithms is in order, despite the current focus of the machine learning community on creating ensembles with a single learning algorithm with injected randomness or its application to manipulated training sets, input features and output targets. This should include the pursuit for better sets of meta-level features and better meta-level learning algorithms.

Acknowledgements

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. Thanks also to Alexander Seewald for providing his implementation of grading in Weka.

References

1. D. Aha, D. W. Kibler, and M. K. Albert. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.
2. C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998.
3. T. G. Dietterich. Machine-learning research: Four current directions. *AI Magazine*, 18(4):97–136, 1997.
4. T. G. Dietterich. Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923, 1998.
5. T. G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 1–15, Berlin, 2000. Springer.
6. S. Džeroski and B. Ženko. Is combining classifiers better than selecting the best one? In *Proceedings of the Nineteenth International Conference on Machine Learning*, San Francisco, 2002. Morgan Kaufmann.
7. G. H. John and P. Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Francisco, 1995. Morgan Kaufmann.
8. C. J. Merz. Using correspondence analysis to combine classifiers. *Machine Learning*, 36(1/2):33–58, 1999.
9. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, 1993.
10. A. K. Seewald and J. Fürnkranz. An evaluation of grading classifiers. In *Advances in Intelligent Data Analysis: Proceedings of the Fourth International Symposium (IDA-01)*, pages 221–232, Berlin, 2001. Springer.
11. K. M. Ting and I. H. Witten. Issues in stacked generalization. *Journal of Artificial Intelligence Research*, 10:271–289, 1999.
12. L. Todorovski and S. Džeroski. Combining multiple models with meta decision trees. In *Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery*, pages 54–64, Berlin, 2000. Springer.

13. L. Todorovski and S. Džeroski. Combining classifiers with meta decision trees. *Machine Learning*, In press, 2002.
14. I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco, 1999.
15. D. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–260, 1992.
16. B. Ženko, L. Todorovski, and S. Džeroski. A comparison of stacking with MDTs to bagging, boosting, and other stacking methods. In *Proceedings of the First IEEE International Conference on Data Mining*, pages 669–670, Los Alamitos, 2001. IEEE Computer Society.