



DSC 2003 Working Papers
(Draft Versions)

<http://www.ci.tuwien.ac.at/Conferences/DSC-2003/>

Bundling Predictors in R

Torsten Hothorn

Institut für Medizininformatik, Biometrie und Epidemiologie
Friedrich-Alexander Universität Erlangen-Nürnberg
Waldstraße 6, D-91054 Erlangen
Torsten.Hothorn@rzmail.uni-erlangen.de

1 Introduction

The construction of a good classifier based on a learning sample can be seen as a three step procedure. At first, we use the observations in the learning sample to construct different rules. In the second step, we need to choose the best among them. This is usually done by selecting the rule with minimum estimated misclassification error. And at last, but not least, an honest estimate of the misclassification error of the selected procedure is required, for example to decide whether this classifier is good enough to be applied in practical situations or not. A common problem is that only a small learning sample with a large number of possible predictors is available and all three steps have to be performed using this small learning sample.

Two main problems arise. Firstly, we need to choose an appropriate classifier out of a number of possible candidates, for example linear or tree based classifiers, neural networks, nearest neighbors or support vector machines. And secondly, we need to estimate the misclassification error of the selected procedure. It is well known that the selection of a classification rule with minimum estimated misclassification error leads to biased estimates of its performance. Nevertheless, different rules have to be taken into account. In many applications simple rules like naive Bayes, nearest neighbors or linear discriminant analysis (LDA) perform comparably to more advanced classifiers (e.g. [Friedman, 1997](#)). However, the individual classifiers perform well in different situations and fail under different conditions.

One approach to solve both problems simultaneously, i.e. the method selection and error rate estimation problem, is to apply a combination of classifiers. Instead of selecting one single procedure, combining the competitors may improve classification rules. There are several approaches to the combination of different classifiers. A linear combination of the estimated conditional class probabilities is suggested by [LeBlanc and Tibshirani \(1996\)](#) and [Mojirsheibani \(1997\)](#), which is related to linear combinations of regression models ([Breiman, 1996c](#); [LeBlanc and Tibshirani, 1996](#)). [Merz \(1999\)](#) uses correspondence analysis to combine the prediction of different classifiers. Majority voting of the predictions of the different classifiers is introduced by [Mojirsheibani \(1999\)](#).

A combination of LDA and classification trees via bagging was introduced by [Hothorn and Lausen \(2003b\)](#) and generalized to the combination of arbitrary classifiers by [Hothorn and Lausen \(2003a\)](#). The basic idea is to add the outcome of arbitrary classifiers (linear discriminant variables, predicted conditional class probabilities or predicted classes), which were trained using the out-of-bag observations only, to the set of original predictors for bagging of classification trees ([Breiman, 1996a, 1998](#)).

In this paper we will show how the bagging procedure in the `ipred` package ([Peters et al., 2002](#)) can be used to combine different classifiers. Furthermore, some preliminary results of benchmark experiments on combined predictors for regression problems are given.

2 Bundling: Combining Arbitrary Classifiers

Let $\mathcal{L}_n = \{(y_i, \mathbf{x}_i), i = 1, \dots, n\}$ denote a learning sample of n independent observations consisting of p -dimensional vectors of predictors $\mathbf{x}_i = (x_{i1}, \dots, x_{ip}) \in \mathbb{R}^p$ and class labels $y_i \in \{1, \dots, J\}$. A classifier $C(\mathbf{x}; \mathcal{L}_n)$ predicts future y -values for a vector of predictors \mathbf{x} based on a learning sample \mathcal{L}_n . We will use superscripts to distinguish between classifiers of different origin: C^1, \dots, C^K , for example LDA, nearest neighbors, logistic regression, and so on.

Drawing a random sample of size n from the empirical distribution, a bootstrap sample of size n covers approximately $2/3$ of the observations of the learning sample. The observations which are not in the bootstrap sample are called out-of-bag sample and may be used for estimating the misclassification error or for improved class probability estimates, see [Breiman \(1996a,b\)](#).

In our framework, the out-of-bag sample is used to train C^1, \dots, C^K . Each of those classifiers can be used to compute a $J - 1$ dimensional transformation of the observations in the bootstrap sample: the values of a linear discriminant function or estimated conditional class probabilities.

The combination is performed as follows:

- a) Draw a random sample \mathcal{L}_n^* with replacement from \mathcal{L}_n .
- b) Train all classifiers C^1, \dots, C^K using the out-of-bag observations $\mathcal{L}_n \setminus \mathcal{L}_n^*$ only.

- c) Compute the estimated conditional class probabilities (or alternatively the values of the linear discriminant functions for LDA or predicted classes) for all observations in the bootstrap sample \mathcal{L}_n^* and construct a classification tree based on all original variables as well as the estimated conditional class probabilities of C^1, \dots, C^K .

Repeat the procedure B times and classify a new observation \mathbf{x} by majority voting using the predictions of all B trees. Because the outcomes of C^1, \dots, C^K are "bundled" by the classification trees, the procedure is called "bundling".

3 Implementation

Trees for nominal, continuous and censored responses are available in the R system via the `rpart` package (Therneau and Atkinson, 1997). The `rpart` function can easily be used for bagging classification trees: simply call `rpart` for bootstrap samples of the learning sample and concatenate the resulting `rpart` objects into a list. The prediction of a new observation is easy, too. Predict the class of the new observation for each tree in the list and aggregate the predictions by majority voting (for example using `table`).

Two main difficulties arise. For bundling, we need to compute arbitrary, user-specified additional classifiers for each out-of-bag sample and compute their predictions both for the bootstrap sample as well as for any new observation to classify. Therefore, we need to save the additional classifiers for each bootstrap sample. Another major problem is speed. Calling `rpart` 50 times, say, leads to repeated unnecessary computations: formula evaluation, determination of the measurement scale for each predictor and so on. Unfortunately, there is a trade-off between a flexible implementation and speed. We therefore decided to speed up bagging by a modification of the `rpart` routine and to generalize bundling at the price of efficiency.

The implementation of the `rpart` routine currently does not separate the evaluation of formula objects and the construction of appropriate design matrices from the tree construction itself which leads to unnecessary computations if multiple trees are constructed for reweighted observations in the learning sample. Therefore, the `ipred` package implements a modified version called `irpart` which grows multiple trees without reevaluating formula objects in order to save computing time.

Both bagging and bundling are implemented in the generic `ipredbagg` which dispatches on the class of the response: methods for factors (classification) and numeric responses (regression) as well as responses of class `Surv` (censored data) currently exists. A formula based interface to `ipredbagg` is offered by `bagging`, a generic itself which dispatches on the `data` argument.

```
bagging(formula, data, subset, na.action, ...)
```

Currently only a method for data frames is implemented.

As mentioned in the previous Section, the interface to bundling was designed to allow users to specify additional classifiers in a flexible and easy way. Basically,

for each classifier a list with two elements is required: `model` and `predict`, where `model` specifies a function for training of the classifier and `predict` is a function for computing predictions. We require at least two arguments for `model`: `formula` and `data`. For `predict`, exactly two arguments are allowed: `object` and `newdata`. If more than one additional classifier should be used, a list of lists with `model` and `predict` elements can be defined for bundling via the `comb` argument.

For the experiments here, we combine the values of the linear discriminant variables of a stabilized linear discriminant analysis, the predicted classes of nearest neighbors ($k = 5$ and $k = 10$) and the estimated conditional class probability derived from the logistic regression model. The associated list can be defined as given here and passed to `bagging` via its `comb` argument (we use the Ionosphere data as example):

```
R> cbundle <- list(
  # stabilized LDA
  list(model=slda, predict=function(object, newdata)
      predict.slda(object, newdata)$x),
  # 5-NN
  list(model=function(...) ipredknn(..., k=5),
      predict=predict.ipredknn),
  # 10-NN
  list(model=function(...) ipredknn(..., k=10),
      predict=predict.ipredknn),
  # LR or multinomial model, resp.
  list(model=function(...) multinom(..., trace=FALSE),
      predict=function(obj, newdata)
      predict.multinom(obj, newdata, type="prob"))
)
R> library(ipred)

R> data(Ionosphere)
R> Ionosphere$V2 <- NULL
R> cmod <- bagging(Class ~ ., data = Ionosphere, comb = cbundle)
R> predict(cmod, Ionosphere[1:8, ])

[1] good bad good bad good bad good bad
Levels: bad good
```

For each bootstrap sample, the additional classifiers are trained and one single function `bfct` for prediction is created in the environment of the current bootstrap sample. Everytime `bfct(newdata)` is called, the predictions of the additional classifiers are computed in the corresponding environment ("lexical scoping", [Gentleman and Ihaka, 2000](#)) and an explicit knowledge of those objects is not needed.

	sLDA	5-NN	10-NN	LR	Bagg	Bund	RF
Twonorm	2.5	3.9	3.4	5.1	6.9	2.8	4.8
Threenorm	17.4	18.4	16.9	17.9	19.6	16.6	17.8
Ringnorm	38.9	47.4	49.3	38.1	10.0	6.5	7.3
Breast Cancer	3.4	6.7	8.3	7.3	4.0	2.9	3.1
Ionosphere	13.9	15.7	16.6	12.5	7.8	6.0	6.4
Diabetes	26.9	28.6	26.5	22.4	24.3	24.2	23.7
Glass	42.4	32.7	38.1	35.2	23.0	24.2	21.3
Satellite	19.3	8.7	9.6	19.2	8.4	7.2	7.6
Shuttle	8.2	0.4	0.6	2.9	0.1	0.1	0.1
DNA	8.1	18.6	16.4	10.4	4.6	2.7	5.5

Table 1: Estimated misclassification errors for some of the UCI benchmark problems.

4 Benchmark Experiments: Classification

In this Section we illustrate the performance of the combination of classifiers via bundling using three artificial, four small and three larger benchmark problems. The datasets and simulation models are assembled in the `m1bench` package (Leisch and Dimitriadou, 2001).

We study bundling of three individual classifiers: stabilized linear discriminant analysis (sLDA, see Hothorn and Lausen, 2003a), k nearest neighbors (k -NN, with $k = 5$ and $k = 10$) as well as the logistic regression model (LR). The multinomial model is used for problems with more than two classes. The values of the linear discriminant functions of the stabilized LDA as well as the predicted conditional class probabilities of nearest neighbors and the logistic regression model are combined. For bagging (Bagg) and bundling (Bund), 100 unpruned trees are used. We additionally report the error rates of random forests with 100 trees (Forest-RI, Breiman, 2001), R package `randomForest` ("RF", Liaw and Wiener, 2002, version 3.4-4), where the number of randomly selected predictors in each node is chosen as the ceiling of $\log_2(p + 1)$.

The misclassification error for the artificial problems is the average over 100 simulation runs, where the learning samples are of size 300 and the error rate is computed using one single test sample of size 18000. For the larger datasets, a test sample is selected randomly for the larger problems. The misclassification error for the smaller problems is estimated by averaging the misclassification error of ten independent runs of 10-fold cross-validation.

The simulated or estimated misclassification errors for the artificial and real world benchmark datasets are given in Table 1. A graphical representation of the simulation results for the artificial problems is shown in Figure 1.

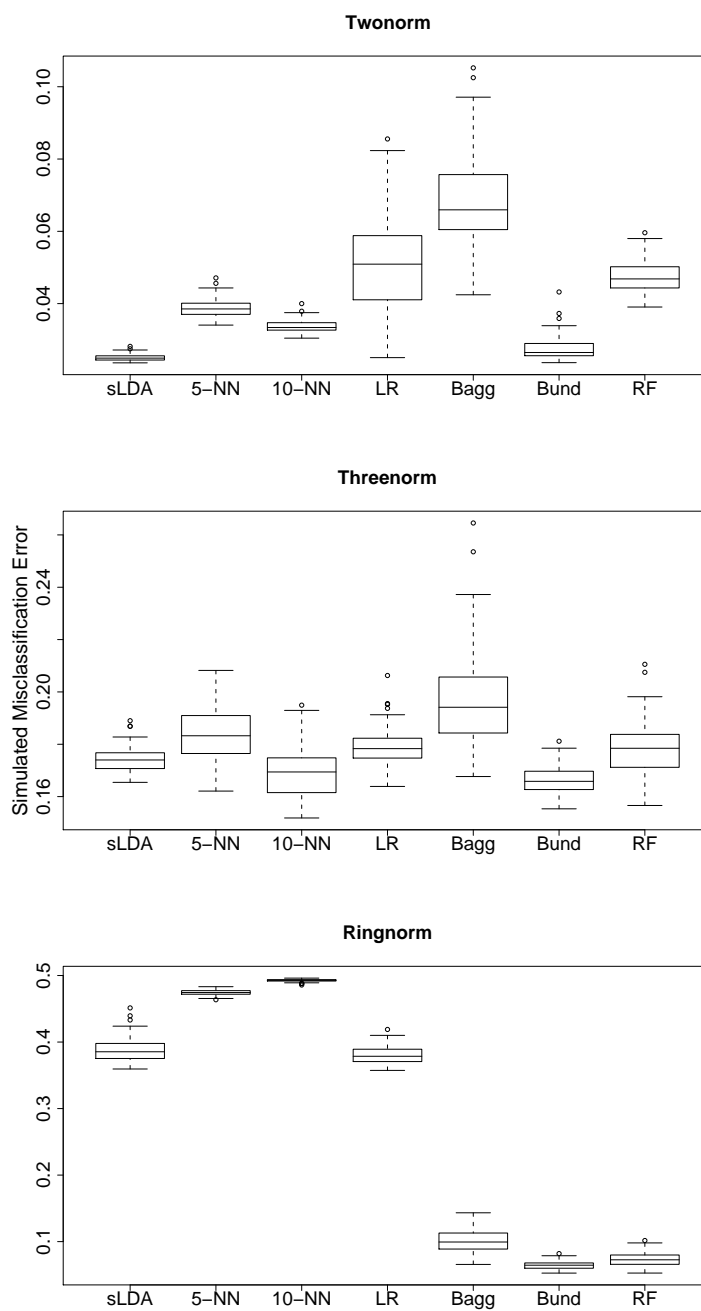


Figure 1: Misclassification error of 100 simulation runs for the artificial classification problems.

Survial-Tree	Bagging	Bundling	Cox-Model
0.174	0.163	0.160	0.163

Table 2: Integrated Brier scores for GBSG2 data.

5 Extention on Regression Problems

The idea of combining classifiers given in Section 2 can easily be extended to regression and survival problems, where the responses are real valued observations $y_i \in \mathbb{R}$ or censored: $y_i \in \mathbb{R} \times \{0, 1\}$. In the regression context, the coefficients of a linear model can be estimated by using the out-of-bag sample and its predictions on the bootstrap sample can be used as an additional predictor for regression trees. For censored responses, the linear predictor of a Cox model can be incorporated into bagging of survival trees (Hothorn et al., 2002) by the same procedure. In contrast to bagging of classification trees, where the trees are grown until the nodes are pure, it is not obvious when to stop the tree growing for bagging of regression or survival trees.

The user interface is exactly the same as for classification problems. A combination of a linear model and regression trees for the Ozone data can be trained by

```
R> rbundle <- list(list(model = lm, predict = predict.lm))
R> data(airquality)
R> rmod <- bagging(Ozone ~ ., data = airquality, comb = rbundle,
+   control = rpart.control(minsplit = 2, xval = 0, cp = 0))
R> predict(rmod, airquality[1:3, ])

[1] 35.32 30.80 18.20
```

Using the linear predictor of a Cox model as an additional variable to a survival tree is possible along the following lines, data from the German Breast Cancer Study Group 2 are used as example:

```
R> sbundle <- list(list(model = coxph, predict = predict.coxph))
R> data(GBSG2)
R> smod <- bagging(Surv(time, cens) ~ ., data = GBSG2, comb = sbundle,
+   control = rpart.control(xval = 0))
R> predict(smod, GBSG2[1, ])

[[1]]
Call: survfit(formula = Surv(aggsample[[j]], aggcens[[j]]))

           n   events    rmean se(rmean)   median  0.95LCL  0.95UCL
2543.0  1070.0  1722.5    19.7   1814.0   1814.0   1918.0
```

The integrated Brier score (function `sbrier`, Graf et al., 1999) can be used as a measure of goodness-of-prediction. Table 2 gives the average of ten times ten-fold cross-validated integrated Brier scores for the GBSG2 data.

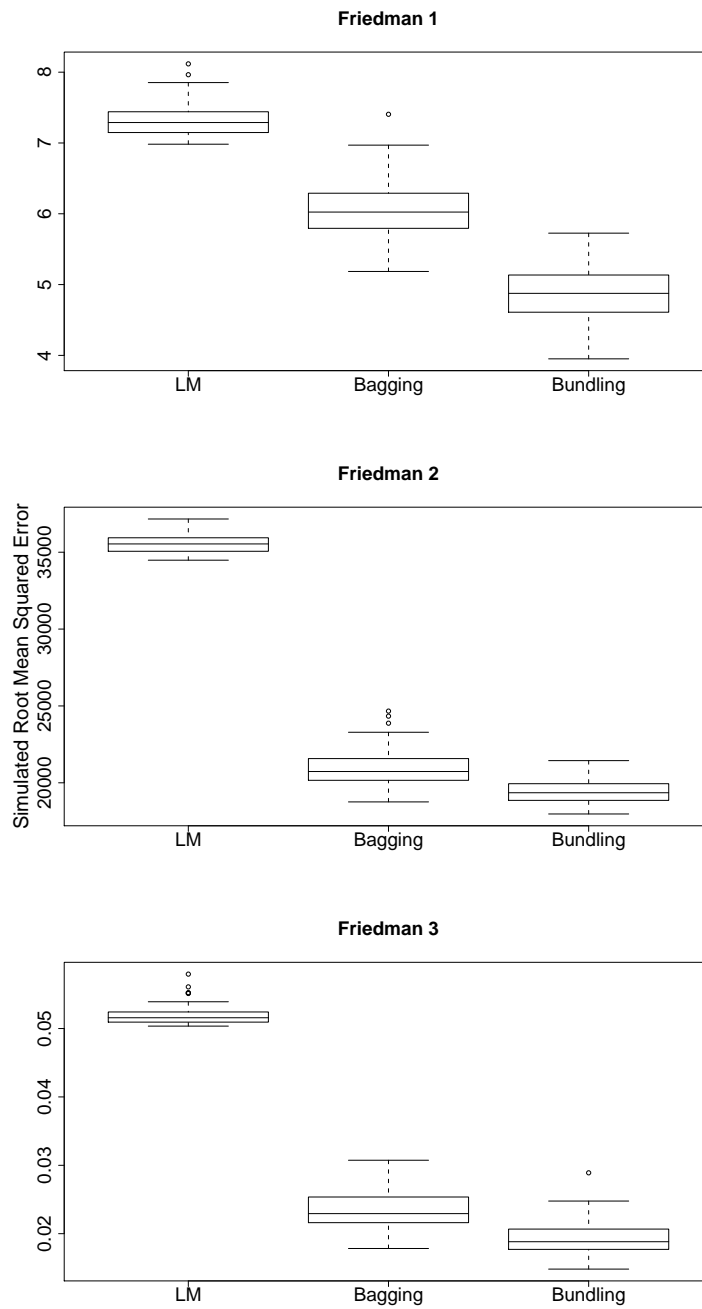


Figure 2: Root mean squared error of 100 simulation runs for the artificial regression problems.

6 Benchmark Experiments: Regression

For the artificial problems Friedman 1, 2 and 3, we use learning samples of size 200 and average the root mean squared errors over 100 simulation runs. For the Ozone, Airquality and BostonHousing data, ten independent runs of ten-fold cross-validation are used to estimate the root mean squared error. For all experiments, 100 trees are grown for bagging and bundling. Each tree is grown to the maximum size, although for some of the problems smaller trees perform better. The results are given in Table 3 and Figure 2.

	LM	Bagging	Bundling
Friedman 1	7.3	6.0	4.9
Friedman 2 ($\times 10^3$)	35.6	20.9	19.5
Friedman 3 ($\times 10^{-3}$)	51.9	23.4	19.2
Airquality	465.2	316.1	302.4
BostonHousing	23.8	10.6	11.5
Ozone	20.0	18.5	16.6

Table 3: Estimated root mean squared error for some regression problems.

7 Summary

The bagging procedure can be used to combine arbitrary predictors in the recursive partitioning framework for classification, regression and survival problems. The interface design is rather general, however, each model needs to implement a formula based interface.

Benchmark experiments for classification problems show that a bundle of different classifiers performs at least comparably to each of the competitors or even leads to an improvement with respect to misclassification error. Although the results for combined regression models are preliminary, they indicate the gain of model combination for regression problems.

I would like to thank Kurt Hornik for discussions and suggestions with respect to the design of the user interface to bagging.

References

- Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996a.
- Leo Breiman. Out-of-bag estimation. Technical report, Statistics Department, University of California Berkeley, Berkeley CA 94708, 1996b. URL <ftp://ftp.stat.berkeley.edu/pub/users/breiman/>.
- Leo Breiman. Stacked regressions. *Machine Learning*, 24:49–64, 1996c.

- Leo Breiman. Arcing classifiers. *The Annals of Statistics*, 26(3):801–824, 1998.
- Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
- Jerome H. Friedman. On bias, variance, 0/1-loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, 1(1):55–77, 1997.
- Robert Gentleman and Ross Ihaka. Lexical scope and statistical computing. *Journal of Computational and Graphical Statistics*, 9:491–508, 2000.
- Erika Graf, Claudia Schmoor, Willi Sauerbrei, and Martin Schumacher. Assessment and comparison of prognostic classification schemes for survival data. *Statistics in Medicine*, 18(17-18):2529–2545, 1999.
- Torsten Hothorn and Berthold Lausen. Bundling classifiers by bagging trees. *Preprint, Friedrich-Alexander-University Erlangen-Nuremberg*, 2003a. URL <http://www.mathpreprints.com/>.
- Torsten Hothorn and Berthold Lausen. Double-bagging: Combining classifiers by bootstrap aggregation. *Pattern Recognition*, 36(6):1303–1309, 2003b.
- Torsten Hothorn, Berthold Lausen, Axel Benner, and Martin Radespiel-Tröger. Bagging survival trees. *Preprint, Friedrich-Alexander-University Erlangen-Nuremberg*, 2002. URL <http://www.mathpreprints.com/>.
- Michael LeBlanc and Robert Tibshirani. Combining estimates in regression and classification. *Journal of the American Statistical Association*, 91(436):1641–1650, 1996.
- Friedrich Leisch and Evgenia Dimitriadou. mlbench – A collection for artificial and real-world machine learning benchmarking problems, 2001. URL <http://CRAN.R-project.org>. R package version 0.5-4.
- Andy Liaw and Matthew Wiener. Classification and regression by randomForest. *R News*, 2(3):18–22, 2002. URL <http://CRAN.R-project.org/doc/Rnews/>.
- Christopher J. Merz. Using correspondence analysis to combine classifiers. *Machine Learning*, 36(1-2):33–58, 1999.
- Majid Mojirsheibani. A consistent combined classification rule. *Statistics & Probability Letters*, 36(1):43–47, 1997.
- Majid Mojirsheibani. Combining classifiers via discretization. *Journal of the American Statistical Association*, 94(446):600–609, 1999.
- Andrea Peters, Torsten Hothorn, and Berthold Lausen. ipred: Improved predictors. *R News*, 2(2):33–36, 2002. URL <http://CRAN.R-project.org/doc/Rnews/>.
- Terry M. Therneau and Elizabeth J. Atkinson. An introduction to recursive partitioning using the rpart routine. Technical Report 61, Section of Biostatistics, Mayo Clinic, Rochester, 1997. URL <http://www.mayo.edu/hsr/techrpt/61.pdf>.