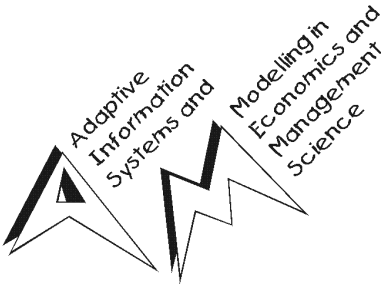


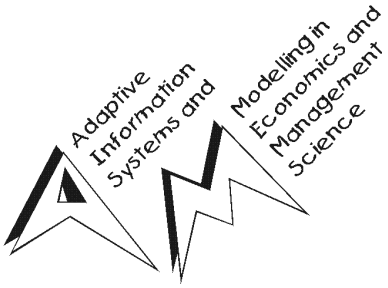
Report Series

Benchmarking Support Vector Machines

David Meyer
Friedrich Leisch
Kurt Hornik

Report No. 78
November 2002

Report Series



November 2002

SFB
'Adaptive Information Systems and Modelling in Economics and Management Science'

Vienna University of Economics
and Business Administration
Augasse 2–6, 1090 Wien, Austria

in cooperation with
University of Vienna
Vienna University of Technology

http://www.wu-wien.ac.at/am

Papers published in this report series
are preliminary versions of journal articles
and not for quotations.

This piece of research was supported by the Austrian Science Foundation (FWF) under grant SFB#010 ('Adaptive Information Systems and Modelling in Economics and Management Science').

Abstract

Support Vector Machines (SVMs) are rarely benchmarked against other classification or regression methods. We compare a popular SVM implementation (`libsvm`) to 16 classification methods and 9 regression methods—all accessible through the software R—by the means of standard performance measures (classification error and mean squared error) which are also analyzed by the means of bias-variance decompositions. SVMs showed mostly good performances both on classification and regression tasks, but other methods proved to be very competitive.

1 Introduction

Support vector machines are currently a hot topic in the machine learning community, creating a similar enthusiasm at the moment as artificial neural networks did previously. New variants arise every week, claiming for superiority in certain situations compared to the "standard" SVM or to popular implementations, but they are seldom compared to other methods (e.g. Ralaivola and d'Alché Buc, 2001; Ben-Hur et al., 2001; Mangasarian and Musicant, 2000), or only to a few—mostly to neural networks (see, e.g., Mayoraz, 2001). Even rich methodological text books like Hastie et al. (2001) restrict themselves to a few comparisons. This is all the more surprising as benchmarking is a common exercise in the machine learning community: see, e.g., the ESPRIT StatLog project reports (very comprehensive, but meanwhile outdated: Michie et al. (1994), and also: King et al. (1995)) or the more recent comparison from Lim et al. (2000) which focuses on tree based methods. Our contribution aims at filling this gap by comparing SVMs to several classification and regression methods on real and artificial standard benchmarking problems.

2 Methods

For our benchmark study, we compare SVMs to 16 methods for classification and to 9 for regression. The methodological spectrum includes "conventional" methods (e.g., linear models) as well as "modern" and flexible methods (trees, splines, neural networks, ...). All methods used are accessible from the R software (see Ihaka and Gentleman (1996) and `http://www.R-project.org`), an open source implementation of the computer language S (see Chambers, 1998)—either by a native implementation or an interface to existing libraries. In Table 11 (Appendix), we list the corresponding R packages along with their version numbers. We used R version 1.6.1.

For SVMs, we use the C++ library `LIBSVM` (Version 2.35) from Chang and Lin (2001), which performed well on several occasions[1] and for which an R interface is available (Meyer, 2001). For our study, we use C-classification and ϵ-support vector regression with the radial basis function (RBF) kernel. The classification model implemented in `libsvm` is basically a binary classifier, made multi-class aware by a voting mechanism. Other SVM formulations for "true" multi-class classification exist (see, e.g., Weston and Watkins, 1998; Guermeur et al., 2000) and the discussion still goes on (see, e.g., Hsu and Lin, 2001). In order to avoid the introduction of additional complications, we therefore restrict our study to binary classification problems—multi-class classification probably deserving a separate study, taking into account the various kinds of "multi-class awareness".

Most methods, especially their implementations, are described in Venables and Ripley (1996), and also in Ripley (1996). Background information on MARS and BRUTO, as well as for MDA and FDA, can be found in Hastie et al. (2001). Random forests are described in Breiman (2001). For MART, see Friedman (2001) and Friedman (2002). Further information and references can

[1] The library won EUNITE world wide competition on electricity load prediction, December 2001 and the IJCNN 2001 Challenge by solving two of three problems: the Generalization Ability Challenge (GAC) and the Text Decoding Challenge (TDC)

be found in the help pages of the packages. In the following, we give a short overview of the methods used—the corresponding acronyms in parentheses correspond to the labels in the tables and figures.

2.1 Classification

Support Vector Machines (SVM) are used for support vector C-classification with RBF kernel. **[svm]**

Classification trees try to find an optimal partitioning of the space of possible observations, mainly by the means of subsequent recursive splits. . . . . **[rpart]**

Linear Discriminant Analysis (LDA) seeks a linear combination of the variables maximizing the ratio of between-class variance to within-class variance, thus yielding a set of separating hyperplanes (One for each pair of classes). . . . . **[lda]**

Quadratic Discriminant Analysis (QDA) is similar to LDA, but the boundaries are approximated by quadratic functions. QDA cannot handle categorical predictors. . . . . **[qda]**

Neural Networks in R include the most popular architecture: a single hidden layer perceptron with optional short cut connections. We restrict the so-defined model class (which also includes the linear model) to architectures with at least one hidden unit and no short cut connections to prevent a "fall-back" to LDA. . . . . **[nnet]**

Generalized Linear Models (GLM) with binomial link model the logit of the class probabilities by a linear combination of the input variables. The parameters are fitted by a reweighted least squares procedure. . . . . **[glm]**

Multinomial Logit Models (MLM) try to explain a categorical response by linear predictors. For a binary response, this model is equivalent to a GLM with binomial link, but the parameter estimation procedure is different: the MLM is implemented in R through a neural network with shortcut connections only, which, in turn, uses a quasi-Newton optimization procedure instead of the iterated weighted least squares (IWLS) procedure used by GLMs. **[multinom]**

Nearest Neighbour (NN) classifier use the training data to assign new features to the class determined by the nearest data point. It uses the Euclidean distance measure (not suitable for categorical predictors). . . . . **[nn]**

Learning Vector Quantization (LVQ) replaces the training data by class prototypes which are iteratively moved close to data points of the corresponding class. We use the oLVQ1 variant. **[lvq]**

Flexible Discriminant Analysis (FDA) is basically inspired by LDA, which can also be performed by first mapping the classes into $\mathbb{R}$ and then fitting a sequence of linear regression models. Predictions for new examples are then labeled via nearest class centroid assignment in the space of fits. FDA extends this idea by replacing the linear predictors by more sophisticated ones, e.g. MARS or BRUTO (see next section on regression methods). We refer to the resulting classifiers as to . . . . . **[f.mars]** and to **[f.bruto]**.

Mixture Discriminant Analysis (MDA) implements a mixture of classifiers, the parameters being estimated via Maximum Likelihood using the EM-Algorithm. Hastie et al. (2001) propose to use mixtures of FDA classifiers (again with MARS and BRUTO methods); accordingly, we refer to them as to . . . . . **[m.mars]** and **[m.bruto]**.

Bagging of trees combines several tree predictors trained on bootstrap samples of the training data; prediction is done by majority vote. . . . . **[bagg]**

Double Bagging is an extension described in Hothorn and Lausen (2002), where the predictor set is expanded by discriminating variables generated by ordinary LDA trained on the out-of-bag samples. As for ordinary bagging. . . . . **[dbagg]**

Random Forests is based on bagged trees, but in addition uses random feature selection at each node for the set of splitting variables. . . . . **[rForst]**

Multiple Additive Regression Trees (MART) implement a tree boosting algorithm based on a gradient descent search. . . . . **[MART]**

2.2 Regression

Linear Regression is also included as baseline method although the test data is supposed to be nonlinear. . . . . **[lm]**

ϵ-Support Vector Regression is used for regression (with $\epsilon = 0.5$), again using the RBF-kernel. **[svm]**

Neural Networks are also applied to regression, using the linear output option. . . . . **[nnet]**

Regression trees (like their classification counterparts) also split the feature space into rectangles, and fit a simple (usually constant) model in each partition. The partitioning is chosen as to minimize the sum of squared errors. The result is an multidimensional step function. **[rpart]**

Projection Pursuit Regression (PPR) explains the target variable by a sum of spline functions of projections of the input variables. The number of these spline components is specified by a range (we choose 2 to 5). . . . . **[ppr]**

Multivariate Adaptive Regression Splines (MARS) model the target variable by a linear combination of splines, which are automatically built up from an increasing set of piecewise defined linear basic functions. To avoid overfitting, a pruning mechanism is applied afterwards to reduce the model complexity. . . . . **[mars]**

Additive Spline Models by Adaptive Backfitting (BRUTO) are similar, but build up the whole basis function set at the beginning and try to estimate the parameters in a way to cancel unnecessary components. BRUTO cannot handle categorical predictors. . . . . **[bruto]**

bagging of trees, **random forests** and **MART** are also used for regression purposes (all averaging the numerical predictions of the predictor trees). . . . . **[bagg], [rForst], [MART]**

3 Data Sets

For our benchmark study, we use 21 data sets for classification and 12 data sets for regression. Most data originate from the UCI machine learning database (Blake and Merz, 1998)—with the exception of the "abalone" and the "cpuSmall" ("cpu comp-activ") data which are taken from the DELVE[2] project (Neal, 1998), and the "SLID data" which is the result of the 1994 wave of the Canadian Survey of Labour and Income Dynamics (SLID). Most of them are standard in benchmarking (see, e.g., "Proben1" from Prechelt, 1994). Some characteristics are listed in Tables 12 and 13 (Appendix). The data has been assembled in the R package `mlbench` (Leisch and Dimitriadou, 2001).

As Prechelt (1995) suggests, we also use artificial data. For regression, we choose the well-known Friedman problems 1–3 (see Friedman, 1991)—the standard deviations used were 1, 125 and 0.1, respectively (the latter two values yielding a 3:1 signal to noise ratio as proposed by the author). For classification, we use 5 problems: "2 noisy spirals" (two entangled spirals with noise, $\sigma = 0.05$), "circle in a square" (a hypersphere lying in the middle of a hypercube), "twonorm" (two multinormal distributions), "threenorm" (three multinomial distributions, two of them forming one class) and "ringnorm" (one smaller multinomial distribution 'inside' a bigger one). For all these artificial problems, data can be created by functions again provided in `mlbench`.

[2] `http://www.cs.toronto.edu/~delve/`

Some data preprocessing was done as follows: missing values were omitted. Further, not all methods can handle categorical data: most implementations (especially the `svm` interface) therefore use the usual binary coding scheme for design matrices of linear models. We scaled all metric variables to zero mean and unit variance to prevent numeric problems—especially recommended for support vector machines and neural networks which may perform badly on some tasks with values in a large range (see, e.g., Hastie et al., 2001). To obtain error measures comparable to published results, the predictions were scaled back before computing the error estimates.

4 Simulation Procedure

The simulation was set up as follows. From each data set, 100 training and 100 test sets were produced: for artificial data, we generated 100 training sets (200 examples) and 100 test sets (1000 examples); for real data, we used 10 times repeated 10-fold cross validation (i.e., 10 partitions with non-overlapping test sets from 10 permutations). All training and test set samples used for this simulation study are available from `http://www.ci.tuwien.ac.at/~meyer/benchdata/`. All methods were trained and tested on these preproduced data sets using an automated benchmark framework available from the authors which uses a flexible and easily extendible wrapper technique to handle the wide range of data sets and methods.

Hyperparameters were tuned for SVMs and neural networks on every run by using 1/3 of the training data for validation: For support vector machines, parameter tuning was performed by a grid search over the 2-dimensional parameter space (C, γ)—C ranges from 2^{-5} to 2^{12} and γ from 2^{-10} to 2^{5}—by training all corresponding SVMs on 2/3 of the training set and choosing the one minimizing the classification error/MSE on the remaining 1/3 (validation set). For neural networks, parameter tuning on the number of hidden units was performed by training 5 times all nets with the number of hidden units varying from 1 to $\log(n)$, n the number of observations, and choosing the value minimizing the mean classification error on the validation set. Because of the stochastic nature of the parameter optimization routine (gradient descent with randomly selected starting values), we trained 5 nets on each run and chose the best (with minimal training error) for the prediction on the test set. For bagging, double bagging and random forests, we tried different values for the parameter specifying the number of trees (25 to 150) and chose 100 for all three of them—higher values resulting in only minor increases in performance. The same procedure was applied to MART for which 1500 iterations turned out to be a reasonable choice. Finally, we tuned two hyperparameters of `rForst`: the number of variables randomly sampled as candidates at each split, and the minimum size of terminal nodes. We used ranges from 1 to $\lfloor \log_2 k + 1 \rfloor$ (k the number of variables), and 1 to 10, respectively.

5 Performance Measures

As performance measures we chose the mean squared error for regression and the classification error for classification. For artificial data, we also computed the corresponding decompositions into model bias, model variance and data variance, which is a sensible tool in the analysis of simulation results (see, e.g., Leisch, 1998, who assessed performance of ensemble methods.) For the classification case, various definitions exist (e.g., Breiman, 1998; Kong and Dietterich, 1995; Kohavi and Wolpert, 1996): among these three, we chose the simplest method defined by Kong and Dietterich, which does not take into account the data variance (which for our data sets is close to 0 anyway because the classes are almost non-overlapping).

6 Results

For each combination of method and dataset, we computed mean, median and interquartile range (IQR) of the test set error rates (the IQRs are computed using the means of the cross validation

results). We include the means to allow comparisons with existing results from other studies, but we base our analysis on the more robust median, complemented by the IQR as a dispersion measure. Note that in fact, the rankings induced by the medians sometimes differ from the ones based on means which are possibly biased by extreme values. The detailed figures are given in the Appendix: for classification, Tables 1 and 2 give the means of the classification test set errors, whereas for regression, Table 8 summarizes the means of the mean squared test set errors (MSE). Tables 3–6 contain the medians/IQRs for classification and Table 9 for regression. Top scores are underlined.

Some of the cells are empty, mostly due to implementation imperfections: `qda`, `mars`, `bruto` and `f.bruto` cannot handle categorical predictors, `lm` fails on the "autos" data set because the design matrix is not full rank (more variables than observations), `bruto` yielded meaningless results on "cpuSmall", `m.mars` could not deal with the solely categorical predictors of the "titanic" data set, and finally, `f.bruto` and `m.bruto` apparently had problems with the high dimensionality of the "credit" data set and did not terminate. The results for `nn` must be handled with care for data with categorical variables: as the Euclidean distance is used, the binary coding often makes them useless.

6.1 Classification

As a sanity check, we compared our SVM results to existing ones found in the literature, although the results must be handled with care due to the use of different software implementations and different partitioning of the input data into training and test sets. Also, authors typically do not include dispersion statistics. We indicate our mean values in parentheses.

- Onoda et al. (2001) found 23.5% (23.53%) for "P.I.Diabetes", 23.6% (23.65%) for "german credit", 16.0% (15.87%) for "heart1", 22.4% (21.16%) for "titanic", and 3.0% (3.58%) for "ringnorm".
- Gestel et al. (2001) list 29% (29.11%) for "liver", 23% (23.65%) for "german credit", 23% (23.53%) for "P.I.Diabetes", 24% (15.44%) for "sonar", 1.0 (0.14%) for "tictactoe", and 4% (3.14%) for "BreastCancer".
- Demirez and Bennett (2000) achieved 3.4% (3.14%) on "BreastCancer", 16% (15.87%) on "heart1", 10.6% (5.93%) on "ionosphere", 8.5% (5.98%) on "musk", 14.3% (15.44%) on "sonar", and 21.8% (23.53%) on "P.I.Diabetes".
- Herbrich et al. (1999) report 25.4% (15.87%) on "heart1", 33.1% (23.53%) on "P.I.Diabetes", 15.4% (15.44%) on "sonar", and 11.9% (5.93%) on "ionosphere".
- Anguita et al. (2000) found 3.9% (3.14%) for "BreastCancer", 15% (3.94%) for "hepatitis", 5.8% (5.93%) for "ionosphere", 12.6% (15.44%) for "sonar" and 0.51% (0.14)% for "tictactoe".
- Vincent and Bengio (2001) report 3.5% (3.14%) for "BreastCancer", 23.0% (23.53%) for "P.I.Diabetes", and 6.6% (5.93%) on "ionosphere".
- Auer et al. (2002) used SMO and found 3.13% (3.14%) for "BreastCancer", 0.57% (0.49%) for "chess", 24.55% (23.65%) for "german credit", 22.68% (23.53%) for "P.I.Diabetes", 19.22% (15.87%) for "heart1", 8.80% (5.93%) for "ionosphere", and 15.48% (15.44%) for "sonar".

The following analysis was done upon the median error statistics: in our simulations, support vector machines generally performed good on all data sets—they were always ranked in the top 3 (except for "heart1" and "cards"). However, they were outperformed on 10 out of 21 data sets (in the following, the value in parentheses after the data set name is the SVM result):

- On "BreastCancer" (2.92%) by `rForst` (1.49%) and `dbagg` (2.9%);
- On "heart1" (15.47%) by `dbagg` (13.12%), `lda` (13.33%), and `glm`/`multinom`/`rForst`/`bagg` (13.79%);

- On “cards” (12.4%) by `bagg` (6.3%), `dbagg` (6.35%), `rForst` (7.52%), `mart` (10.77%) and `nn` (10.94%);
- On “titanic” (21.27%) by `nnet`/`bagg`/`dbagg`/`mart` (20.81%);
- On “sonar” (15%) by `nn` (10%);
- On “P.I.Diabetes” (23.23%) by `lda`/`multi`/`f.mars`/`glm` (22.08%);
- On “liver” (28.57%) by `rForst` (27.02%);
- On “credit” (23.76%) by `multi`/`glm`/`dbagg`/`mart` (22.77%) and `bagg` (22.89%);
- On “spirals” (0.5%) by `nn` (0.1%); and
- On “threenorm” (15.45%) by `lvq` (13.7%).

In addition, SVMs share 5 top ranks with some other methods on “HouseVotes84” (almost all), “ionosphere” (`dbagg`), “hepatitis” (almost all), “monks3” (`rpart` and `mart`) and “promotergene” (`multi`, `glm`, `bagg`, `dbagg`).

Note the good performance of ensemble methods (`bagg`, `dbagg`, `rForst`, `mart`) on real data sets, and even of simple linear models (`glm` and `lda`) on some occasions[3]. In addition, we emphasize the fact that extensive hyperparameter tuning has only been performed for SVMs, neural networks and random forests—results of other methods can certainly still be improved!

As for the dispersion characteristics, the interquartile ranges show average rankings (SVM are on the top only for “monks3”): the results seem not exceptionally stable. Finally, when we compare the bias-variance decompositions of SVMs with the corresponding ones of the other methods, SVMs prove to be affected both by bias and variance (however more by bias on “twonorm”, “threenorm” and “ringnorm”).

6.2 Regression

In the literature, we found the following results from Tipping (2000): 2.92 (4.36) for “Friedman1”, 41.400 (18.130) for “Friedman2”, 20.2 (23.15) for “Friedman3”, and 8.04 (9.60) for “BostonHousing”. These results clearly differ from our results: for “BostonHousing”, the author trained 100 randomized samples with 481/25 training/test set splits, so he used 95% for training and 5% for testing compared to our choice of 90%/10%. Moreover, the author’s samples are likely to be more dependent than our set choice, because more samples do overlap. As to the “Friedman” data sets, the variance parameter of the three functions is not indicated, so we cannot be sure if the problems are really the same; especially for “Friedman2”, this is doubtful. In addition, the author used slightly bigger training sets (240 observations, compared to our 200).

Apart from that, the results on regression (again based on medians) are comparable to those for classification: again, SVMs are almost always ranked in the top 3 (except for “servo” and “Friedman3”). But they are first ranked on two data sets only (“BostonHousing” and “Friedman2”), and outperformed on all other occasions:

- On “ozone” (3.09) by `nnet` (0.14) and `ppr` (1.96);
- On “autos” (0.51) by `ppr` (0.21);
- On “cpu” (1.86) by `rForst` (1.27);
- On “SLID” (37.58) by `rForst` (33.33);
- On “autompg” (6.48) by `nnet` (6.37);
- On “servo” (0.26) by `rForst` (0.16), `nnet` (0.19) and `ppr` (0.20);

[3] Breiman et al. (1984) noted it were “. . . of surprise that [LDA] does as well as it does . . . ”. We only used the radial basis function kernel, though; some additional experiments showed that by the use of a linear kernel, similar results as with LDA could be achieved.

- On “abalone” (4.48) by `nnet` (4.29) and `ppr` (4.44);
- On “cpuSmall” (9.01) by `mart` (7.53);
- On “Friedman1” (4.31) by `bruto` (3.20) and `mars` (3.51); and
- On “Friedman3” (22.44) by `nnet` (6.25), `ppr` (21.15) and `rForst` (22.06).

Apparently, `nnet`, `ppr` and `rForst` seem to be good alternatives to support vector regression. Again—as for classification—the IQR values for the SVM results, compared to the other methods, do not reveal high stability, and the bias-variance decomposition also yielded that SVMs are influenced both by bias and variance.

7 Conclusion

Support vector machines yielded good performance, but were not top ranked on all data sets. For classification, simple statistical procedures and ensemble methods proved very competitive, mostly producing good results “out of the box” without the inconvenience of delicate and computationally expensive hyperparameter tuning. For regression tasks, neural networks, projection pursuit regression and random forests often yielded better results than SVMs. As to the dispersion characteristics, the results for SVMs compared to other methods showed average precision. However, there is no evidence that the SVM results were particularly affected by either model bias or model variance. In short, our results confirm the potential of SVMs to yield good results, especially for classification, but their overall superiority cannot be attested.

Acknowledgements

This research was supported by the Austrian Science Foundation (FWF) under grant SFB#010 (‘Adaptive Information Systems and Modeling in Economics and Management Science’).

We are particularly grateful to the reviewers for many helpful comments and suggestions. In addition, we thank Torsten Hothorn for his help regarding the use of the “`ipred`” package and his critical review of the corresponding results, and also Andy Liaw for his support in using the “`randomForest`” package.

References

Anguita, D., Boni, A., and Ridella, S. (2000). Evaluating the generalization ability of support vector machines through the bootstrap. *Neural Processing Letters*, 11(1):51–58.

Auer, P., Burgsteiner, H., and Maass, W. (2002). Reducing communication for distributed learning in neural networks. In *Artificial Neural Networks—ICANN 2001*. Springer-Verlag.

Ben-Hur, A., Horn, D., Siegelmann, H. T., and Vapnik, V. (2001). Support vector clustering. *Journal of Machine Learning Research*, 2:125–137.

Blake, C. and Merz, C. (1998). UCI repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences, `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Breiman, L. (1998). Arcing classifiers. *Annals of Statistics*, 26(3):801–849.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Wadsworth, California.

Chambers, J. M. (1998). *Programming with Data: a guide to the S Language*. Springer.

Chang, C.-C. and Lin, C.-J. (2001). Libsvm: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Demirez, A. and Bennett, K. (2000). Optimization approaches to semisupervised learning. In Ferris, M., Mangasarian, O., and Pang, J., editors, *Applications and Algorithms of Complementarity*. Kluwer Academic Publishers, Boston.

Friedman, J. (1991). Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1202.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378.

Gestel, T. V., Suykens, J., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., Moor, B. D., and Vandewalle, J. (2001). Benchmarking least squares support vector machine classifiers. *Machine Learning*. To appear.

Guermeur, Y., Eliseeff, A., and PaugamMoisy, H. (2000). A new multi-class svm based on a uniform convergence result. In Amari, S.-I., Giles, C., Gori, M., and Piuri, V., editors, *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks IJCNN 2000*, pages IV–183 – IV–188. Los Alamitos, IEEE Computer Society.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer.

Herbrich, R., Graepel, T., and Campbell, C. (1999). Bayes point machines: Estimating the bayes point in kernel space. In *Proceedings of IJCAI Workshop Support Vector Machines*, pages 23–27.

Hothorn, T. and Lausen, B. (2002). Double-bagging: Combining classifiers by bootstrap aggregation. *Pattern Recognition*. To appear.

Hsu, C.-W. and Lin, C.-J. (2001). A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*. To appear.

Ihaka, R. and Gentleman, R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314.

King, R., Feng, C., and Shutherland, A. (1995). Statlog: comparison of classification algorithms on large real-world problems. In *Applied Artificial Intelligence*, volume 9, pages 259–287.

Kohavi, R. and Wolpert, D. H. (1996). Bias plus variance decomposition for zero-one loss. In Saitta, L., editor, *Machine Learning: Proceedings of the 13th International Conference*, pages 275–283. Morgan Kaufmann.

Kong, E. B. and Dietterich, T. G. (1995). Error-correcting output coding corrects bias and variance. In *Machine Learning: Proceedings of the 12th International Conference*, pages 313–321. Morgan-Kaufmann.

Leisch, F. (1998). *Ensemble Methods for Neural Clustering and Classification*. PhD thesis. Technische Universität Wien. Available from: http://www.ci.tuwien.ac.at/~leisch/papers/Leisch:1998.ps.gz.

Leisch, F. and Dimitriadou, E. (2001). mlbench—a collection for artificial and real-world machine learning benchmarking problems. R package, Version 0.5-6. Available from http://cran.R-project.org.

Lim, T.-S., Loh, W.-Y., and Shih, Y.-S. (2000). A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning*, 40(3):203–228.

Mangasarian, O. L. and Musicant, D. R. (2000). Robust linear and support vector regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(9):950–955.

Mayoraz, E. N. (2001). Multiclass classification with pairwise coupled neural networks or support vector machines. In Dorffner, G., Bischoff, H., and Hornik, K., editors, *Artificial Neural Networks—ICANN 2001*, Vienna. Springer.

Meyer, D. (2001). Support vector machines. *R News*, 1(3):23–26. http://CRAN.R-project.org/doc/Rnews/.

Michie, D., Spiegelhalter, D. J., and Taylor, C. C. (1994). *Machine Learning, Neural and Statistical Classification*. Ellis Horwood.

Neal, R. (1998). Assessing relevance determination methods using delve generalization in neural networks and machine learning. In Bishop, C., editor, *Neural Networks and Machine Learning*. Springer.

Onoda, T., Rätsch, G., and Müller, K.-R. (2001). Soft margins for adaboost. *Machine Learning*, 42(3):287–320.

Prechelt, L. (1994). Proben1—a set of neural network benchmark problems and benchmarking rules. Fakultaet fuer Informatik, Universitaet Karlsruhe, Germany., available from ftp://ftp.ira.uka.de/pub/papers/techreports/1994/1994-21.ps.gz.

Prechelt, L. (1995). Some notes on neural learning algorithm benchmarking. *Neurocomputing*, 9(3):343–347.

Ralaivola, L. and d'Alché Buc, F. (2001). Incremental support vector machine learning: A local approach. In Dorffner, G., Bischoff, H., and Hornik, K., editors, *Artificial Neural Networks—ICANN 2001*, Vienna. Springer.

Ripley, B. D. (1996). *Pattern recognition and neural networks*. Cambridge University Press, Cambridge, UK.

Tipping, M. (2000). The relevance vector machine. In Morgan Kaufmann, San Mateo, C., editor, *Advances in Neural Information Processing Systems*.

Vapnik, V. (1998). *Statistical learning theory*. Wiley, New York.

Venables, W. and Ripley, B. D. (1996). *Modern Applied Statistics with S-Plus*. Springer, third edition. Software available at http://www.stats.ox.ac.uk/pub/MASS3/.

Vincent, P. and Bengio, Y. (2001). A neural support vector network architecture with adaptive kernels. In *Proceedings of the Neural Information Processing Systems (NIPS) Conference*.

Weston, J. and Watkins, C. (1998). Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, Egham, TW20 0EX, UK.

Appendix

	svm	lda	rpart	qda	nn	multi	f.mars	f.bruto	nnet
BreastCancer	3.14	3.56	5.51		24.27	5.86	4.09		4.49
HouseVotes84	2.58	2.62	2.62		13.75	0.00	2.62		0.00
heart1	15.87	13.67	18.52		19.58	14.70	15.98		14.50
ionosphere	5.93	12.99	12.80		13.62	11.77	9.66		12.13
cards	12.48	13.47	13.38		11.44	13.03	13.62		12.67
hepatitis	3.94	5.05	12.50		0.00	0.00	3.61		0.48
tictactoe	0.14	1.69	8.24		33.97	1.80	1.69		2.57
chess	0.49	5.73	3.20		39.73	2.43	5.88		1.14
monks3	1.07	3.57	1.07		28.65	1.97	3.57		1.74
promotergene	10.96	28.34	30.89		23.44	12.31	16.73		14.98
titanic	21.16	22.16	21.48		66.90	22.16	22.16		21.09
sonar	15.44	25.20	29.42	25.97	12.69	27.10	22.30	30.08	21.96
P.I.Diabetes	23.53	22.60	25.38	25.94	29.74	22.37	22.67	34.90	23.73
liver	29.11	32.03	32.77	39.45	37.12	31.75	29.56	57.96	34.25
musk	5.98	19.48	22.48	33.08	11.69	17.39	14.65	43.26	14.14
credit	23.65	23.12	25.12	28.45	33.13	23.24	24.10		27.59
circle	2.66	49.48	10.65	10.94	5.88	49.49	5.61	4.05	4.17
spirals	0.81	49.99	3.70	50.00	0.17	49.99	9.49	7.90	3.37
twonorm	2.82	3.16	25.98	5.32	7.27	6.95	6.34	4.13	5.32
threenorm	15.76	18.20	34.36	20.86	25.29	18.58	24.29	21.69	22.01
ringnorm	3.58	38.75	25.28	6.53	40.87	39.07	9.34	9.06	30.47

Table 1: Classification Test Set Errors (Mean) 1—the first part contains data with both categorical and metric variables, data in the second part have metric variables only, and the third part lists the synthetic data sets.

	svm	m.mars	m.bruto	glm	rForst	bagg	dbagg	mart	lvq
BreastCancer	3.14	4.79		5.32	2.28	3.21	2.67	3.86	4.89
HouseVotes84	2.58	2.61		0.00	2.36	0.00	0.00	2.49	5.94
heart1	15.87	18.20		14.66	14.15	14.93	13.10	16.55	20.62
ionosphere	5.93	9.32		11.91	7.32	8.32	7.09	8.84	14.38
cards	12.48	14.32		12.88	7.82	6.74	6.99	11.23	18.40
hepatitis	3.94	5.87		0.00	2.75	0.00	0.00	8.80	8.04
tictactoe	0.14	1.69		1.80	1.22	1.28	2.44	2.58	25.31
chess	0.49	8.33		2.46	1.38	0.40	0.71	0.99	17.04
monks3	1.07	3.55		1.90	1.72	1.96	1.92	1.07	10.06
promotergene	10.96	19.48		41.42	11.00	11.37	11.74	17.67	33.55
titanic	21.16			22.16	21.46	21.07	21.04	21.14	22.79
sonar	15.44	23.17	25.09	26.27	16.20	20.82	18.81	23.29	17.87
P.I.Diabetes	23.53	25.95	34.90	22.37	23.60	24.09	23.44	24.38	28.33
liver	29.11	31.42	42.04	31.75	*27.04	29.86	29.45	29.96	41.71
musk	5.98	11.63	43.26	19.53	10.61	12.96	12.12	11.82	13.67
credit	23.65	27.27		23.24	23.45	22.89	22.65	23.51	28.39
circle	2.66	9.87	8.93	49.49	6.73	7.47	6.59	8.49	44.30
spirals	0.81	2.18	1.46	49.99	2.43	2.71	2.21	6.45	46.28
twonorm	2.82	10.53	5.92	5.64	4.09	7.96	2.83	12.99	3.07
threenorm	15.76	24.80	19.30	18.58	*18.55	21.22	17.22	25.66	14.17
ringnorm	3.58	16.57	16.31	39.07	*5.92	11.93	11.57	15.86	38.07

*) Breiman (2001) found better results for “liver” (25.1%), “threenorm” (17.5%) and “ringnorm” (4.9%). For the two latter, a larger training set (300 examples) was used.

Table 2: Classification Test Set Errors (Mean) 2

	svm	lda	rpart	nn	multi	f.mars	nnet
BreastCancer	2.92	2.94	5.84	24.29	5.80	3.03	4.41
	(0.72)	(0.84)	(0.92)	(1.52)	(1.25)	(0.66)	(0.94)
HouseVotes84	0.00	0.00	0.00	15.38	0.00	0.00	0.00
	(2.60)	(3.92)	(3.92)	(5.82)	(0.00)	(3.92)	(0.00)
heart1	15.47	13.33	17.55	19.35	13.79	16.67	13.79
	(2.01)	(0.89)	(4.32)	(0.79)	(1.29)	(2.82)	(1.61)
ionosphere	5.71	11.43	14.09	11.43	11.43	8.57	11.43
	(1.10)	(0.49)	(0.78)	(0.31)	(1.28)	(0.92)	(1.86)
cards	12.40	13.85	13.18	10.94	13.33	13.85	12.40
	(3.06)	(2.07)	(1.74)	(0.97)	(2.08)	(2.52)	(2.45)
hepatitis	0.00	0.00	11.11	0.00	0.00	0.00	0.00
	(2.99)	(6.36)	(8.45)	(0.00)	(0.00)	(3.58)	(0.75)
tictactoe	0.00	2.06	8.33	34.02	2.06	2.06	2.08
	(0.18)	(0.08)	(0.95)	(0.22)	(0.08)	(0.08)	(0.52)
chess	0.31	5.61	3.12	39.88	2.49	5.62	1.25
	(0.19)	(0.20)	(0.06)	(0.20)	(0.10)	(0.12)	(0.19)
monks3	1.75	3.57	1.75	28.57	1.79	3.57	1.79
	(0.00)	(0.01)	(0.00)	(1.17)	(0.31)	(0.01)	(0.49)
promotergene	9.09	27.27	33.33	25.00	9.09	16.67	16.67
	(2.88)	(2.22)	(4.26)	(4.75)	(5.97)	(3.03)	(2.10)
titanic	21.27	22.17	21.27	66.52	22.17	22.17	20.81
	(0.45)	(0.08)	(0.49)	(0.34)	(0.08)	(0.08)	(0.24)

Table 3: Classification: Medians and IQRs of test set errors (1). This table lists data with both categorical and metric variables.

	svm	m.mars	glm	rForst	bagg	dbagg	mart	lvq
BreastCancer	2.92	4.41	4.48	1.49	2.94	2.90	2.99	4.41
	(0.72)	(1.02)	(0.51)	(0.16)	(0.27)	(0.31)	(0.81)	(0.98)
HouseVotes84	0.00	0.00	0.00	0.00	0.00	0.00	0.00	4.55
	(2.60)	(3.77)	(0.00)	(3.24)	(0.00)	(0.00)	(3.18)	(3.10)
heart1	15.47	17.24	13.79	13.79	13.79	13.12	16.40	19.35
	(2.01)	(2.74)	(1.29)	(1.97)	(1.27)	(1.69)	(1.50)	(3.18)
ionosphere	5.71	8.57	11.43	7.02	8.57	5.71	8.57	14.29
	(1.10)	(1.34)	(1.35)	(0.58)	(0.50)	(0.70)	(1.15)	(0.28)
cards	12.40	14.06	13.24	7.52	6.30	6.35	10.77	18.46
	(3.06)	(1.81)	(2.22)	(0.55)	(0.78)	(1.09)	(1.83)	(4.34)
hepatitis	0.00	0.00	0.00	0.00	0.00	0.00	9.09	9.09
	(2.99)	(7.85)	(0.00)	(4.43)	(0.00)	(0.00)	(5.92)	(5.02)
tictactoe	0.00	2.06	2.06	1.03	1.03	2.07	2.08	25.77
	(0.18)	(0.08)	(0.08)	(0.11)	(0.20)	(0.10)	(0.34)	(1.34)
chess	0.31	8.27	2.49	1.25	0.31	0.62	0.93	16.85
	(0.19)	(0.64)	(0.16)	(0.12)	(0.11)	(0.09)	(0.09)	(0.49)
monks3	1.75	3.57	1.79	1.79	1.79	1.79	1.75	8.93
	(0.00)	(0.01)	(0.18)	(0.40)	(0.18)	(0.27)	(0.00)	(1.56)
promotergene	9.09	16.67	41.67	9.09	9.09	9.09	16.67	36.36
	(2.88)	(4.64)	(11.80)	(3.75)	(2.61)	(1.46)	(1.89)	(2.97)
titanic	21.27		22.17	21.27	20.81	20.81	20.81	22.17
	(0.45)		(0.08)	(0.43)	(0.17)	(0.17)	(0.32)	(0.76)

Table 4: Classification: Medians and IQRs of test set errors (2). This table lists data with both categorical and metric variables.

	svm	lda	rpart	qda	nn	multi	f.mars	f.bruto	nnet
sonar	15.00	24.40	28.57	23.81	10.00	28.57	20.00	30.00	20.00
	(1.68)	(1.88)	(4.10)	(0.69)	(0.98)	(1.86)	(1.33)	(4.75)	(1.78)
P.I.Diabetes	23.23	22.08	24.68	25.97	28.95	22.08	22.08	35.06	23.38
	(0.52)	(0.58)	(0.75)	(0.41)	(0.48)	(0.26)	(0.93)	(0.02)	(0.72)
liver	28.57	32.38	33.33	40.00	37.14	31.43	28.57	58.33	34.29
	(1.74)	(1.34)	(3.14)	(0.81)	(1.11)	(1.20)	(1.63)	(0.47)	(0.49)
musk	6.12	20.41	22.45	32.65	12.24	16.67	14.58	43.30	14.29
	(1.19)	(0.84)	(1.71)	(1.76)	(0.89)	(0.64)	(1.11)	(0.37)	(2.89)
credit	23.76	22.77	25.25	28.36	33.17	22.77	24.26		26.87
	(0.60)	(0.42)	(0.87)	(1.19)	(0.59)	(0.20)	(0.60)		(1.46)
circle	2.50	49.60	10.25	10.50	5.90	49.60	5.55	3.70	3.75
	(0.49)	(2.40)	(0.46)	(1.89)	(0.60)	(2.40)	(0.74)	(0.51)	(0.98)
spirals	0.50	50.00	3.60	50.00	0.10	50.00	9.50	7.85	2.20
	(0.51)	(0.04)	(0.28)	(0.03)	(0.07)	(0.05)	(0.32)	(0.32)	(1.29)
twonorm	2.70	3.10	25.95	5.20	7.20	7.20	6.20	4.15	5.10
	(0.20)	(0.22)	(0.63)	(0.43)	(0.50)	(0.81)	(0.36)	(0.39)	(0.61)
threenorm	15.45	18.30	34.35	20.70	25.20	18.60	24.60	21.50	21.75
	(0.60)	(0.81)	(0.72)	(0.75)	(0.54)	(0.94)	(1.20)	(0.51)	(1.22)
ringnorm	2.90	38.75	25.05	6.50	40.90	39.00	9.25	8.90	30.25
	(0.70)	(0.97)	(1.27)	(0.51)	(0.58)	(0.71)	(1.08)	(0.70)	(0.75)

Table 5: Classification: Medians and IQRs of test set errors (3). The upper part shows data sets with metric variables only, the lower part contains the artificial data sets.

	svm	m.mars	m.bruto	glm	rForst	bagg	dbagg	mart	lvq
sonar	15.00	23.81	21.36	25.00	15.00	19.05	19.05	23.81	19.05
	(1.68)	(3.49)	(5.60)	(1.75)	(1.13)	(1.85)	(2.15)	(2.60)	(3.43)
P.I.Diabetes	23.23	25.81	35.06	22.08	23.38	23.68	23.38	23.68	27.27
	(0.52)	(1.24)	(0.02)	(0.26)	(1.21)	(0.79)	(1.07)	(0.55)	(1.70)
liver	28.57	30.56	41.67	31.43	27.02	30.56	28.57	28.57	40.00
	(1.74)	(3.15)	(0.47)	(1.20)	(2.15)	(1.40)	(2.18)	(2.03)	(2.34)
musk	6.12	12.24	43.30	20.41	10.36	12.50	12.24	10.42	12.50
	(1.19)	(1.48)	(0.37)	(1.52)	(1.40)	(0.88)	(0.62)	(1.44)	(1.18)
credit	23.76	26.73		22.77	23.76	22.89	22.77	22.77	28.71
	(0.60)	(0.97)		(0.20)	(0.64)	(1.31)	(0.44)	(1.09)	(1.07)
circle	2.50	9.80	8.70	49.60	6.60	7.40	6.50	8.55	41.60
	(0.49)	(0.64)	(1.46)	(2.40)	(0.50)	(0.40)	(0.56)	(0.66)	(3.19)
spirals	0.50	2.00	1.40	50.00	2.30	2.60	2.15	5.80	47.55
	(0.51)	(0.12)	(0.13)	(0.05)	(0.39)	(0.42)	(0.43)	(0.90)	(1.86)
twonorm	2.70	10.60	5.80	5.45	4.00	7.90	2.70	12.05	3.10
	(0.20)	(0.82)	(0.48)	(0.54)	(0.21)	(0.67)	(0.28)	(1.55)	(0.06)
threenorm	15.45	24.65	19.30	18.60	18.40	21.10	17.05	25.30	13.70
	(0.60)	(1.32)	(0.50)	(0.94)	(0.55)	(0.98)	(0.86)	(1.48)	(0.77)
ringnorm	2.90	16.55	16.25	39.00	5.65	11.30	11.10	14.40	37.05
	(0.70)	(0.90)	(1.10)	(0.71)	(0.33)	(0.76)	(0.82)	(2.30)	(0.90)

Table 6: Classification: Medians and IQRs of test set errors (4). The upper part shows data sets with metric variables only, the lower part contains the artificial data sets.

	circle			spirals			twonorm			threenorm			ringnorm		
	bi	va	to	bi	va	to	bi	va	to	bi	va	to	bi	va	to
svm	25	75	3	42	58	1	75	25	3	79	21	16	80	20	2
lda	100	0	53	100	0	50	81	19	3	89	11	19	87	13	38
rpart	79	21	10	37	63	4	14	86	26	42	58	35	21	79	25
qda	77	23	11	100	0	50	50	50	6	59	41	22	56	44	6
nn	20	80	6	14	86	0	32	68	7	49	51	25	100	0	41
multi	100	0	52	100	0	49	42	58	7	84	16	19	89	11	39
f.mars	27	73	5	72	28	11	44	56	6	61	39	25	39	61	10
f.bruto	28	72	4	73	27	9	59	41	4	74	26	21	48	52	9
nnet	7	93	5	8	92	4	65	35	3	80	20	20	57	43	31
m.mars	42	58	10	20	80	4	25	75	11	50	50	24	43	57	17
m.bruto	51	49	9	24	76	3	38	62	6	55	45	19	55	45	16
glm	85	15	49	100	0	50	42	58	5	85	15	19	88	12	39
rForst	76	24	7	60	40	3	47	53	5	72	28	20	35	65	9
bagg	58	42	7	84	16	4	39	61	9	52	48	23	42	58	15
dbagg	60	40	8	51	49	3	86	14	3	79	21	17	25	75	12
mart	61	39	7	79	21	7	31	69	11	49	51	22	43	57	14
lvq	100	0	48	100	0	50	75	25	4	73	27	14	100	0	39

Table 7: Classification Bias-Variance-Decomposition. Bias and variance are expressed as percentages of the total variance. Legend: bi=bias, va=variance, to=total variance (%).

	svm	lm	rpart	nnet	mars	bruto	ppr	rForst	bagg	mart
BostonHousing	9.60	23.66	23.18	15.52			13.46	11.14	16.16	11.19
ozone	4.89	11.31	11.87	2.22			2.16	6.10	8.35	6.73
autos ($\times 10^6$)	1.22		4.34	0.43			0.25	1.22	2.85	1.67
cpu ($\times 10^3$)	5.46	5.01	10.16	5.15			3.16	3.44	8.34	8.97
SLID	38.52	43.31	40.66	37.96			38.79	34.13	39.28	39.50
autompg	7.11	8.89	12.36	7.34			7.75	7.35	9.25	8.48
servo	0.46	1.24	0.76	0.27			0.27	0.25	0.51	0.56
abalone	4.51	4.92	5.86	4.31			4.42	4.57	5.12	4.64
cpuSmall	9.20	97.73	27.72	9.21	10.60		12.11	8.45	23.94	7.55
Friedman1	4.36	7.71	11.87	7.42	3.56	3.22	7.48	6.02	7.82	5.69
Friedman2 ($\times 10^3$)	18.13	36.15	34.40	19.61	38.09	36.06	19.89	21.50	23.03	23.17
Friedman3 ($\times 10^{-3}$)	23.15	45.42	42.35	18.12	25.09	23.19	21.84	22.21	28.14	25.16

Table 8: Regression Mean Squared Test Set Errors (Mean)—the upper table contains real, and the lower synthetic data.

	svm	lm	rpart	nnet	mars	bruto	ppr	rForst	bagg	mart
Boston-	8.01	21.75	22.07	13.81			11.22	10.06	14.14	8.68
Housing	(0.72)	(0.37)	(0.99)	(2.76)			(0.87)	(0.27)	(0.31)	(0.41)
Ozone	3.09	10.40	10.59	0.14			1.96	5.46	7.58	5.24
	(3.71)	(0.82)	(4.39)	(1.68)			(0.12)	(3.69)	(2.78)	(3.42)
autos	0.51		4.28	0.39			0.21	1.10	2.51	0.98
($\times 10^6$)	(0.75)		(2.50)	(0.20)			(0.06)	(0.74)	(1.14)	(0.72)
cpu	1.86	3.49	6.68	2.37			2.03	1.27	2.87	2.59
($\times 10^3$)	(2.71)	(0.48)	(0.98)	(2.34)			(0.58)	(0.65)	(1.58)	(1.80)
SLID	37.58	42.48	40.92	37.79			38.64	33.33	39.13	39.72
	(3.18)	(2.79)	(3.25)	(3.41)			(3.64)	(2.99)	(3.10)	(0.21)
autompg	6.48	8.66	11.34	6.37			6.75	6.64	8.52	8.08
	(0.71)	(0.76)	(1.53)	(0.85)			(0.52)	(0.23)	(0.99)	(0.72)
servo	0.26	1.11	0.47	0.19			0.20	0.16	0.27	0.43
	(0.05)	(0.05)	(0.08)	(0.05)			(0.11)	(0.05)	(0.09)	(0.17)
abalone	4.48	4.93	5.84	4.29			4.44	4.54	5.13	4.64
	(0.02)	(0.02)	(0.05)	(0.03)			(0.04)	(0.03)	(0.02)	(0.04)
cpuSmall	9.01	96.23	26.97	9.09	10.48		10.88	8.46	23.41	7.53
	(0.05)	(0.61)	(0.35)	(0.07)	(0.06)		(0.95)	(0.07)	(0.12)	(0.12)
Friedman1	4.31	7.72	11.78	7.56	3.51	3.20	7.21	6.05	7.69	5.66
	(0.29)	(0.26)	(0.30)	(0.44)	(0.20)	(0.14)	(0.29)	(0.17)	(0.14)	(0.23)
Friedman2	17.99	36.12	33.91	19.33	37.74	36.00	19.26	21.49	22.88	22.86
($\times 10^3$)	(0.76)	(0.52)	(1.50)	(1.20)	(1.22)	(0.54)	(0.33)	(0.74)	(0.95)	(1.27)
Friedman3	22.44	45.19	41.90	16.25	25.14	23.42	21.15	22.06	27.63	23.58
($\times 10^{-3}$)	(1.33)	(0.70)	(2.11)	(1.29)	(0.86)	(0.56)	(0.80)	(1.99)	(1.25)	(2.81)

Table 9: Regression Mean Squared Test Set Errors (Medians and IQRs)—the upper table contains real, and the lower synthetic data.

	Friedman 1				Friedman 2 ($\times 10^3$)				Friedman 3 ($\times 10^{-2}$)			
	d.v	m.b	m.v	t.v	d.v	m.b	m.v	t.v	d.v	m.b	m.v	t.v
svm	27.25	48.42	24.33	4.17	86.17	7.61	6.22	17.36	35.42	56.01	8.57	23.90
lm	9.90	86.34	3.76	6.93	44.46	53.82	1.72	36.20	23.29	74.39	2.32	45.38
rpart	7.72	55.12	37.16	11.11	47.11	17.06	35.83	32.64	22.09	41.53	36.38	45.26
nnet	17.69	78.04	4.27	7.29	72.88	25.10	2.02	18.94	41.57	21.25	37.18	19.58
mars	28.72	58.91	12.37	3.19	44.65	50.36	4.99	40.29	29.73	62.70	7.57	27.59
bruto	32.47	63.25	4.28	3.63	45.31	52.89	1.80	34.16	38.46	57.03	4.51	24.69
ppr	16.22	54.02	29.76	7.04	80.74	4.59	14.67	19.91	51.11	26.99	21.90	21.54
rForst	15.22	72.82	11.96	6.85	77.82	10.16	12.02	20.43	47.47	35.47	17.06	20.15
bagg	13.30	73.92	12.78	7.78	60.93	28.43	10.64	26.79	34.88	48.02	17.10	28.84
mart	12.61	77.27	10.12	4.16	53.19	37.59	9.22	23.07	31.31	55.48	13.21	22.22

Table 10: Regression Bias-Variance-Decomposition (d.v=data variance, m.b=model bias, m.v=model variance, t.v=total variance). Data variance, model bias and model variance are expressed as percentages of the total variance.

Method	R Function	R Package	Version
Support Vector Machine (SVM)	`svm`	**e1071**	1.3-3
Neural Network (NN)	`nnet`	**nnet**	7.0-7
Classification & Regression Trees	`rpart`	**rpart**	3.1-8
Linear Model (LM)	`lm`	**base**	1.6.1
Linear Discriminant Analysis (LDA)	`lda`	**MASS**	7.0-7
Multinomial Log-Linear Model	`multinom`	**nnet**	7.0-7
Generalized Linear Model (GLM) with binomial link	`glm`	**base**	1.6.1
Quadratic Discriminant Analysis (QDA)	`qda`	**MASS**	7.0-7
Nearest Neighbour (NN)	`knn`	**class**	7.0-7
Mixture Discriminant Analysis (MDA)	`mda`	**mda**	0.2-15
Flexible Discriminant Analysis (FDA)	`fda`	**mda**	0.2-15
Multiple Adaptive Regression Splines (MARS)	`mars`	**mda**	0.2-15
Adaptive Splines by Adaptive Backfitting (BRUTO)	`bruto`	**mda**	0.2-15
Projection Pursuit Regression (PPR)	`ppr`	**modreg**	1.6.1
Random Forests	`randomForest`	**randomForest**	3.3-6
Bagging	`bagging`	**ipred**	0.5-8
Double Bagging	`bagging`	**ipred**	0.5-8
Multiple Additive Regression Trees (MART)	`mart`	[a])	

[a]) The software is now commercially distributed: `http://www.salford-systems.com/`

Table 11: Classification and Regression methods in R (version used: 1.6.1)

Problem	#Examples	#Attributes				Class Distribution (%)
		b	c	m	tot.	
BreastCancer	699		9		9	34.48 / 65.52
HouseVotes84	435	16			16	38.62 / 61.38
heart1	303	3	5	5	13	45.54 / 54.46
ionosphere	351	1		32	33	35.90 / 64.10
cards	690	4	5	6	15	44.49 / 55.51
hepatitis	155	13		6	19	48.70 / 51.30
tictactoe	958		9		9	34.66 / 65.34
chess	3196	35	1		36	47.78 / 52.22
monks3	554	2	4		6	47.60 / 52.40
promotergene	106		57		57	50.00 / 50.00
titanic	2201	2	1		3	32.30 / 67.70
sonar	208			60	60	46.64 / 53.36
PimaIndianDiabetes	768			8	8	34.90 / 65.10
liver	345			6	6	42.58 / 57.42
musk	476			166	166	42.99 / 57.00
credit	690			24	24	29.89 / 70.11
circle in a square	200 / 1000			2	2	50.00 / 50.00
2 noisy spirals	200 / 1000			2	2	50.00 / 50.00
twonorm	200 / 1000			20	20	50.00 / 50.00
threeorm	200 / 1000			20	20	50.00 / 50.00
ringnorm	200 / 1000			20	20	50.00 / 50.00

Table 12: Benchmark data sets for classification—the first and second parts contain real data, the lower part synthetic data sets. Legend: b=binary, c=categorical, m=metric.

Problem	#Examples	#Attributes				
		b	c	m	n	tot.
BostonHousing	506	1		12		13
ozone	366	3		9		12
autos	205	4	6	15		25
cpu	209		0	6		6
SLID	7425		2	2		4
autompg	398		3	4		7
servo	167		2	2		4
abalone	4177		1	7		8
cpuSmall	8192			12		12
Friedman1	200 / 1000			5	5	10
Friedman2	200 / 1000			4		4
Friedman3	200 / 1000			4		4

Table 13: Benchmark data sets for regression—the upper part describes real data, the lower part synthetic data sets. Legend: b=binary, c=categorical, m=metric, n=noise