LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

INSTITUT FÜR STATISTIK

Jan Gertheiss & Gerhard Tutz

Feature Selection and Weighting by Nearest Neighbor Ensembles

Technical Report Number 033, 2008
Department of Statistics
University of Munich

http://www.stat.uni-muenchen.de

Feature Selection and Weighting by Nearest Neighbor Ensembles

Jan Gertheiss & Gerhard Tutz
Ludwig-Maximilians-Universität München
Akademiestraße 1, 80799 München
{jan.gertheiss,tutz}@stat.uni-muenchen.de

June 19, 2008

Abstract

In the field of statistical discrimination nearest neighbor methods are a well known, quite simple but successful nonparametric classification tool. In higher dimensions, however, predictive power normally deteriorates. In general, if some covariates are assumed to be noise variables, variable selection is a promising approach. The paper's main focus is on the development and evaluation of a nearest neighbor ensemble with implicit variable selection. In contrast to other nearest neighbor approaches we are not primarily interested in classification, but in estimating the (posterior) class probabilities. In simulation studies and for real world data the proposed nearest neighbor ensemble is compared to an extended forward / backward variable selection procedure for nearest neighbor classifiers, and some alternative well established classification tools (that offer probability estimates as well). Despite its simple structure, the proposed method's performance is quite good - especially if relevant covariates can be separated from noise variables. Another advantage of the presented ensemble is the easy identification of interactions that are usually hard to detect. So not simply variable selection but rather some kind of feature selection is performed.

Keywords: Nearest Neighbor Methods, Variable Selection, Ensemble Methods, Classification

1 Introduction

The rather old nearest neighbor method (Fix and Hodges, 1951) is one of the simplest and most intuitive techniques in the field of statistical discrimination. The method is nonparametric and memory based. A new observation with unknown class label is placed into the class of the observation from the training set that is closest to the new observation - with respect to some covariates. Despite (or because of) its simplicity nearest neighbor predictions are often accurate. Cover and Hart (1967) showed that (for any number of categories) the probability of error of the (one) nearest neighbor rule is bounded above by twice the Bayes error rate. For results when discrimination is based on k nearest neighbors see Ripley (1996).

Closeness or similarity, respectively distance d of two observations is derived from a certain metric in the predictor space. For a given training set $T = \{(y_i, x_i); i = 1, \ldots, n\}$, with y_i denoting the class membership and the vector $x_i = (x_{i1}, \ldots, x_{ip})^T$ representing the predictor values, the nearest neighbor classification of a new observation (y, x) is $\hat{y} = y_{[1]}$, with $(x_{[1]}, y_{[1]})$ denoting the nearest neighbor in the training set, i.e. $d(x, x_{[1]}) = \min_{1 \leq i \leq n}(d(x, x_i))$. A possible distance measure is the so-called Minkowski distance

$$d(x_i, x_r) = \left(\sum_{j=1}^{p} |x_{ij} - x_{rj}|^q \right)^{1/q}.$$

When the Euclidian metric is used one has the special case of $q = 2$. All nearest neighbor methods in the following are based on the Euclidian metric.

In this paper nearest neighbor approaches are applied in a more general way. One aspect of the proposed method is feature selection. Our aim is not only classifying (or estimation) but to find the most influential variables and interaction terms. In higher dimensions simple nearest neighborhood estimates tend to be unstable when noise variables are present. Therefore we aim at selecting relevant variables or interactions between them, and combine the nearest neighborhood predictions based on single or small groups of predictors. Beyond simple classification, we use nearest neighbor concepts as nonparametric estimators for posterior class probabilities - given classes $g = 1, \ldots, G$. If not only the first but the k nearest neighbors of observation i are used for classifying i, the relative frequency of predictions in favor of category g among these neighbors can be seen as an estimate of the probability of category g. These estimates $\hat{\pi}_{ig}$ can take values h/k, $h \in \{0, ..., k\}$, indicating how many times g is observed among the k nearest neighbors of observation i. If neighbors are weighted with respect to their distance to the observation of interest, $\hat{\pi}$ can in principle take all values in $[0, 1]$. Weighting neighbors is for example proposed by Silverman and Jones (1989) and further investigated and implemented by Hechenbichler and Schliep (2004). By

contrast, in the nearest neighbor ensemble proposed in this article probability estimates unequal to h/k, $h \in \{0, ..., k\}$, do not result from the weighting of neighbors but from combining and weighting different single nearest neighbor estimates.

The paper is organized as follows. In the next section the proposed nearest neighbor ensemble approach is presented. By means of simulations studies its behavior and performance is extensively studied in Section 3. Finally in Section 4 the introduced technique is visualized for some real world data

2 Nearest Neighbor Ensembles

Ensemble methods are a general group of methods. Several classifiers - or *learners* - are fitted, votes are counted, and final classification is given by the most popular class. Nearest neighbor ensembles have been proposed by some authors. Domeniconi and Yan (2004) investigated ensembles of nearest neighbor classifiers based on random subsets of predictors, while performing adaptive sampling. That means variable selection is not completely at random, but a probability distribution is employed in the sampling mechanism. The probability distribution is derived from some kind of relevance measure. Thus, the approach is a two-step procedure, because in a first step all features' relevance needs to be determined by an adequate technique. Yankov et al. (2006) dealt with forecasting time series and proposed ensembles of forecasts from different neighborhoods, namely k_1-NN and k_2-NN with small k_1 and larger k_2. The name ensemble, however, is misleading, since prediction for a given unit is finally based on either k_1-NN or k_2-NN. The question what prediction is more suitable needs to be answered by an adequate classifier. Our goal is to perform variable selection, i.e. selection in the feature space, using nearest neighbor methods.

2.1 Basic Concept

We will start with the simplest case when the learners in the ensemble are nearest neighbor estimates based on single variables. Thus, let $\hat{\pi}_{ig(j)}$ be the nearest neighbor estimate of probability that observation i falls in category g, if the distance measure is only based on predictor x_j. The final estimate $\hat{\pi}_{ig}$ is constructed as an ensemble, i.e. the weighted average

$$\hat{\pi}_{ig} = \sum_{j=1}^{p} c_j \hat{\pi}_{ig(j)}, \text{ with } c_j \geq 0 \; \forall j \text{ and } \sum_j c_j = 1. \tag{1}$$

We explicitly use weights c_j on the variables in order to obtain a selection of relevant predictors. These weights - or coefficients - are unknown and need to be determined in some way. They only have to be positive and sum up to one.

Before presenting our strategy for determining these weights, the question shall be answered if further flexibility is possible. Further flexibility would result from weights depending on category g, i.e.

$$\hat{\pi}_{ig} = \sum_j c_{gj} \hat{\pi}_{ig(j)}, \text{ with } c_{gj} \geq 0 \; \forall g, j \text{ and } \sum_j c_{gj} = 1 \; \forall g.$$

However, it can be shown that restriction $c_{1j} = \ldots = c_{Gj} = c_j$ is the only possibility to ensure that $\hat{\pi}_{ig} \geq 0 \; \forall g$ and $\sum_g \hat{\pi}_{ig} = 1$ for all possible future estimations $\{\hat{\pi}_{ig(j)}\}$ with $\hat{\pi}_{ig(j)} \geq 0 \; \forall g, j$ and $\sum_g \hat{\pi}_{ig(j)} = 1 \; \forall j$.

Since it is not known where a new observation i (that is to be classified) will be located in the predictor space, it is unknown what values single estimates $\hat{\pi}_{ig(j)}$ will have. Hence it must be ensured that estimates $\hat{\pi}_{ig}$ sum up to one given any estimates $\{\hat{\pi}_{ig(j)}\}$. The proof of the statement above is given in the appendix.

2.2 Determination of Weights

As mentioned before, in the ensemble formula (1) the single nearest neighbor estimates $\hat{\pi}_{ig(j)}$ are considered as fixed, and weights c_j need to be determined. For that purpose we primarily choose a loss function - or score - $L(y, \hat{\Pi})$, which quantifies how well the true class labels $y = (y_1, \ldots, y_n)^T$ are fitted by probability estimates $(\hat{\Pi})_{ig} = \hat{\pi}_{ig}$, $i = 1, \ldots, n$, $g = 1, \ldots, G$. As it is seen from the ensemble formula (1), given all single nearest neighbor estimates $\{\hat{\pi}_{ig(j)}\}$, the matrix $\hat{\Pi}$, consisting of final estimates $\hat{\pi}_{ig}$, is a function of $c = (c_1, \ldots, c_p)^T$. So for given training data $T = \{(y_i, x_i); i = 1, \ldots, n\}$ (and hence given estimates $\hat{\pi}_{ig(j)}$) our strategy is minimizing $L(y, \hat{\Pi})$ over all possible c.

Possible Loss Functions

Before introducing loss functions, let us represent the categorial response $y_i \in \{1, \ldots, G\}$ by a G-dimensional vector $z_i = (z_{i1}, \ldots, z_{iG})^T$ of indicator variables

$$z_{ig} = \begin{cases} 1, & \text{if } y_i = g \\ 0, & \text{otherwise.} \end{cases}$$

With convention "$0 \cdot \infty = 0$", a somewhat natural loss function is the *Logarithmic Score*

$$L(y, \hat{\Pi}) = \sum_i \sum_g z_{ig} \log(1/\hat{\pi}_{ig}),$$

which can be derived from the likelihood. For a given p-dimensional predictor vector x_i a single z_i is multinomially distributed with one draw:

$$z_i \sim \text{Mult}(1, \pi_i),$$

with the group probabilities depending on x_i and being merged into a vector $\pi_i = (\pi_{i1}, \ldots, \pi_{iG})^T$. So we have $P(y_i = g|x_i) = \pi_{i1}^{z_{i1}} \cdot \ldots \cdot \pi_{iG}^{z_{iG}}$. For a single observation y_i the negative log likelihood is $-l(y_i, \pi_i) = -l(z_i, \pi_i) = \sum_g z_{ig} \log(1/\pi_{ig})$. If a sample of n observations is given, we have $l(y, \Pi) = \sum_{i=1}^n \sum_g z_{ig} \log(\pi_{ig})$, with y denoting the n-dimensional vector of observed classes. The maximum $l(y, \Pi) = 0$ is obtained for $\pi_{ig} = 1$, if $z_{ig} = 1$, and zero otherwise. A single $r(\hat{\pi}_i) = \sum_g z_{ig} \log(1/\hat{\pi}_{ig})$ may be seen as the Kullback-Leibler-divergence between z_i and $\hat{\pi}_i$, also called *"minus log likelihood error"* (Le Cessie and van Houwelingen, 1992).

However, in spite of the close connection to the maximum likelihood principle, the logarithmic scoring rule is not really recommendable, because it is too sensitive with respect to differences between very small probabilities (cf. Selten, 1998). In the present case this *hypersensitivity* may be carried to an extreme. If the number of considered neighbors is only modest and the sample is large, it is not unlikely that there is at least one observation i with no neighbor from the same category. Then the estimated probability $\hat{\pi}_{iy_i}$ for the correct class y_i is zero and the logarithmic loss has value ∞.

The hypersensitivity can be removed via a second order Taylor approximation of the logarithmic score, expanded about $\pi_{ig} = 1$, if $z_{ig} = 1$, and $\pi_{ig} = 0$ otherwise. Since

$$\frac{\partial L(y, \Pi)}{\partial \pi_{ig}} = \frac{-z_{ig}}{\pi_{ig}}, \text{ if } z_{ig} = 1;\ 0 \text{ otherwise,}$$

and

$$\frac{\partial^2 L(y, \Pi)}{\partial \pi_{ig} \partial \pi_{jk}} = \frac{z_{ig}}{\pi_{ig}^2}, \text{ if } z_{ig} = 1, i = j, g = k;\ 0 \text{ otherwise,}$$

we obtain the loss function

$$V(y, \hat{\Pi}) = \sum_i \sum_g z_{ig} \left((1 - \hat{\pi}_{ig}) + \frac{1}{2}(1 - \hat{\pi}_{ig})^2 \right).$$

However, by removing the logarithmic score's hypersensitivity V suffers from a new theoretic disadvantage. By contrast to the logarithmic loss, V is not *"incentive compatible"* (Selten, 1998). Incentive compatibility means that the expected loss $E(V) = \sum_{y=1}^G \pi_y V(y, \hat{\pi}_y)$ for a new observation y with true class probabilities $\pi_1, \ldots, \pi_G$ is not minimized by $\hat{\pi}_g = \pi_g$, $g = 1, \ldots, G$. For a two-class response with $g \in \{1, 2\}$, for example, $E(V)$ is minimized by $\hat{\pi}_1 = 0$, if $\pi_1 < 1/3$; $\hat{\pi}_1 = 3\pi_1 - 1$, if $1/3 \leq \pi_1 \leq 2/3$; $\hat{\pi}_1 = 1$, if $\pi_1 > 2/3$.

An incentive compatible scoring rule that is not hypersensitive is the pure *quadratic* loss (see e.g. Selten, 1998)

$$Q(y, \Pi) = \sum_i \sum_g (z_{ig} - \pi_{ig})^2,$$

which was first introduced by Brier (1950) and is sometimes also called *Brier Score*. If the response has more than two categories, it should be kept in mind

that the logarithmic score, or its approximation, only looks at the estimated probability for the actual class y_i, whereas the quadratic loss Q also takes into account how the estimated probabilities are distributed on the false classes. For fixed estimated probability of the true class the loss is higher, if the true class is faced with a single but strong competitor than in the case when probability mass is equally distributed over the false classes.

Practical Implementation

For practical minimization of the Brier score the procedure is as follows. For each training observation i we create a matrix P_i of single estimates

$$(P_i)_{gj} = \hat{\pi}_{ig(j)}.$$

These matrices are merged into a big matrix $P = (P_1^T | \ldots | P_n^T)^T$. The same is done with dummy vectors z_i, i.e. we have $z = (z_1^T, \ldots, z_n^T)^T$. Now the Brier score as function of $c = (c_1, \ldots, c_p)^T$ can be written in matrix notation:

$$Q(c) = (z - Pc)^T (z - Pc).$$

Hence a quadratic optimization problem with restrictions $c_j \geq 0 \; \forall j$ and $\sum_j c_j = 1$ arises. So weights c_j can be determined using quadratic programming methods, e.g. using the `R` add-on package `quadprog` (Turlach, 2007). An alternative interpretation is in terms of regression: z is regressed (without intercept) on the estimated probabilities forming P.

In spite of the disadvantage of not being incentive compatible the approximate logarithmic loss V shall be tested as an alternative for determining the weights c_j. If rows from P are deleted where z has entry zero, and the resulting matrix is denoted by $\tilde{P}$, V as a function of c can be written is matrix notation as well ($\mathbb{1}$ denotes the vector of length n consisting of 1s):

$$\begin{aligned} V(c) &= \mathbb{1}^T(\mathbb{1} - \tilde{P}c) + \frac{1}{2}(\mathbb{1} - \tilde{P}c)^T(\mathbb{1} - \tilde{P}c) \\ &= \frac{3}{2}n - 2\mathbb{1}^T\tilde{P}c + \frac{1}{2}c^T\tilde{P}^T\tilde{P}c \\ &= -\frac{1}{2}n + \frac{1}{2}(2\mathbb{1} - \tilde{P}c)^T(2\mathbb{1} - \tilde{P}c). \end{aligned}$$

Since $\operatorname{argmin}_c V(c) = \operatorname{argmin}_c (2\mathbb{1} - \tilde{P}c)^T(2\mathbb{1} - \tilde{P}c)$, another regression problem with restrictions arises, which can be solved using quadratic programming methods, too.

2.3 Variable Selection and Interactions

Variable x_j is selected if $c_j \neq 0$. There are various ways to obtain predictors with zero weights. When determination of weights is completed, variable selection can, for example, be done via *hard thresholding*, i.e. coefficients less than a certain threshold t, for example $t = 0.25 \cdot \max_j\{c_j\}$, are set to zero. A coefficient less than $0.25 \cdot \max_j\{c_j\}$ means that the corresponding predictor does not even have 25% of the weight of the 'most important' predictor, resp. estimate. Of course, after eliminating some coefficients, the remaining weights need to be rescaled to sum up to one. The threshold value $0.25 \cdot \max_j\{c_j\}$ is taken as default in all investigations below. Alternatively *soft thresholding* may be applied. When coefficients - denoted by $\tilde{c}_j$ - have been determined, we set $c_j = (\tilde{c}_j - t)^+$ (and rescale); s^+ is the positive part of s, i.e. $s^+ = s$, if $s > 0$, and $s^+ = 0$ otherwise.

If thresholding is used, variables (resp. single estimates) are selected after initial determination of weights. Since coefficients fulfilling $\sum_j c_j = 1$ are anyway obtained by scaling, restrictions may be primarily disregarded when determining weights. If restrictions are replaced by $\sum_j |\tilde{c}_j| \leq s$, a *lasso* type problem (Tibshirani, 1996) arises. Lasso typical selection characteristics cause $\tilde{c}_j = 0$ for some j. With rescaling and $c_j = \tilde{c}_j^+$ a sensible ensemble results. If lasso estimation is done with additional restriction $\tilde{c}_j \geq 0$, the *positive lasso* (Efron et al., 2004) is obtained. Nevertheless we prefer estimation without these additional constrains, since computation of the original lasso solution is easier. In addition variable selection is intended and negative coefficients indicate poor predictive capability of the corresponding term.

Finally - and very importantly - the matrix of predictions P can be augmented by including interactions of predictors. That means adding all predictions $\hat{\pi}_{ig(jl)}$, resp. $\hat{\pi}_{ig(jlm)}$ based on two or even three predictors j, l and m. So the (initial) ensemble may consist of much more than p terms. Obviously this is feasible for small scale problems only, because including interactions P has $p + \binom{p}{2} + \ldots$ columns. However, if variable selection is applied, many terms will be excluded, and the final ensemble will contain only a modest number of estimates.

3 Simulation Studies

To investigate its behavior the presented nearest neighbor ensemble is studied in simulation studies taken from the literature. Particularly when prediction performance is evaluated, the proposed approach should be compared to some alternative procedures, in order to know if the observed performance can be called "good". So we first introduce our reference methods.

3.1 Reference Methods

Two important features of the proposed nearest neighbor ensemble are its selection property and the possible identification of interactions. So the ensemble should be compared to another nearest neighbor approach with suchlike features.

Nearest Neighbor Forward/Backward Variable Selection

The procedure is very simple. We just employ a slightly modified forward variable selection approach based on leaving one out cross validation. For categorial outcomes, p (potential) predictors and a given (small) S the algorithm is as follows.

Forward Variable Selection by Nearest Neighbor Methods

Step 1:

Select up to S predictors from the p predictors for nearest neighbor classification by LOO cross validation.

Step 2:

From the remaining predictors select another set of 1 to S predictors by LOO cross validation and add these predictors to the already chosen one(s).

Step 3:

If there is no improvement in prediction accuracy after Step 2, just take the predictor set from Step 1; else repeat Step 2 until there is no improvement in prediction accuracy anymore.

The tuning parameter S can be seen as the number of simple forward selection steps that are checked in one iteration. Furthermore S can be interpreted as the highest interaction that should be detected. The motivation is that the (marginal) discriminative capability of a certain predictor may be lower or completely absent, if other variables are disregarded. Hence relevant predictors may not be selected by simple forward selection procedures. An illustrating example is given in the next section.

Furthermore the given algorithm may be seen as a slightly modified version of a *GPTA(l,r)* (Kudo and Sklansky, 2000) procedure, i.e. go forward l stages by adding l predictors and go backward r stages by deleting r predictors and repeat this process. Since for small scale problems this approach gives quite good results (cf. Kudo and Sklansky, 2000), we use it as a kind of reference procedure for nearest neighbor based variable selection.

Wherever forward selection is considered backward selection may be seen as an alternative.

Backward Variable Selection by Nearest Neighbor Methods

Step 1:

Perform nearest neighbor classification based on the entire set of p predictors.

Step 2:

By LOO cross validation select a set of maximum S predictors to be excluded from the reference data for nearest neighbor classification.

Step 3:

If there is no improvement in prediction accuracy after Step 2, just take the original predictor set from Step 1; else exclude further sets of 1 to S predictors until there is no improvement in prediction accuracy anymore.

In analogy to the previous paragraph the tuning parameter S can be seen as the number of simple backward selection steps that are checked in one iteration. In the current implementation forward and backward variable selection aim at minimizing the Brier score on a given training set.

Some Alternative Classification Tools

The methods introduced above are compared to weighted 5 nearest neighbors as proposed in Hechenbichler and Schliep (2004), R package `kknn`) - an approach that can also be found in Silverman and Jones (1989), a commentary on the famous Fix and Hodges (1951) paper. Additionally we will report the performance of some other well established classification tools; namely linear discriminant analysis (LDA), CART (Breiman et al., 1984) and Random Forests (Breiman, 2001). For estimation we used the R packages `MASS` (Venables and Ripley, 2002), `rpart` (Therneau and Atkinson, 2007), `randomForest` (Liaw and Wiener, 2002), see R Development Core Team (2007) for further information.

For estimating the ensemble weights the corresponding optimization problem (with restrictions) was solved via quadratic programming methods from the R add-on package `quadprog`. The single estimates forming the ensemble are standard 3 nearest neighbor estimates. Variable selection is done via hard thresholding with $t = 0.25 \max_j c_j$.

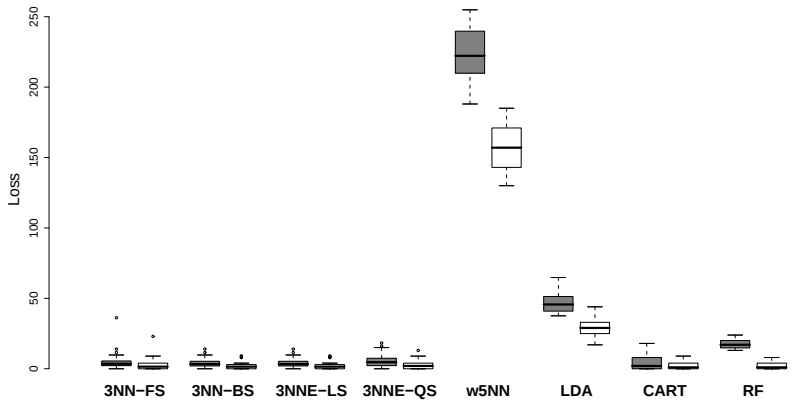

Figure 1: Boxplots of quadratic loss (dark-gray) and number of missclassified observations (light-colored) over 30 simulations for 3 nearest neighbors with forward / backward variable selection ($S = 4$), 3 nearest neighbors ensemble based on approx. log score / quadratic score, weighted 5 nearest neighbors, LDA, CART and Random Forests (RF), given the easy classification problem.

3.2 Two Classification Problems

We look at two simulated problems from Hastie et al. (2001). There are 10 independent features x_j, each uniformly distributed on $[0, 1]$. The two class 0/1 coded response y is defined as follows:

- as an *"easy"* problem: $y = I(x_1 > 0.5)$, and
- as a *"difficult"* problem: $y = I(\text{sign}(\prod_{j=1}^{3}(x_j - 0.5)) > 0)$.

Given the easy problem, dark-gray boxplots in Figure 1 summarize the results in terms of the quadratic loss (Brier score) for 3 nearest neighbors with forward, resp. backward variable selection ($S = 4$) and the developed nearest neighbor ensemble classification technique over 30 realizations of training ($m = 200$) and test ($n = 1000$) data. Due to the disadvantage of not being incentive compatible the approximate logarithmic loss is not appropriate for evaluating the performance of the considered methods. The logarithmic score itself is not applicable either, because with just one test observation falling a category with estimated probability (close to) zero it tends to infinity.

To make variable selection more complicated in the nearest neighbor ensemble, not only predictions based on single predictors are considered but also interaction up to order 3, i.e. predictions using sets of maximum 3 predictors. Ensemble weights are determined via minimizing the approximate log score or the Brier score. So the quadratic loss may be seen as a prejudiced measure of prediction accuracy. Hence as a more neutral quantity we additionally give the number of missclassified test observations (light-colored), if observations are classified as belonging to the category with highest posterior probability.

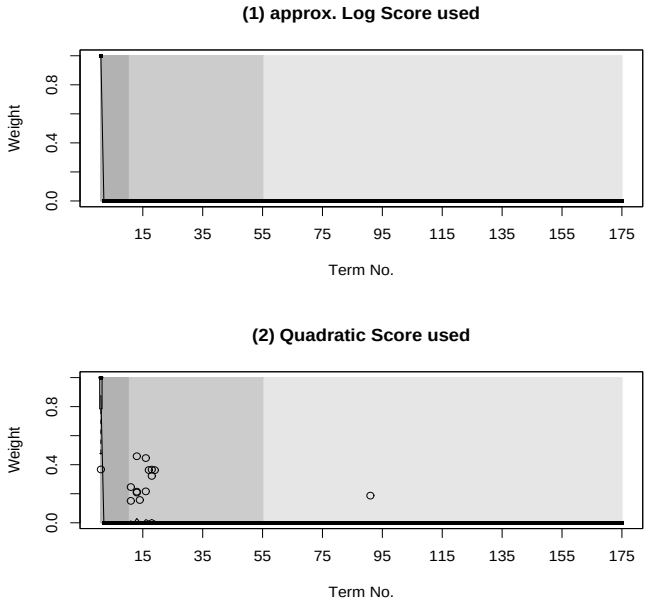

Figure 2: Boxplots of weights in the nearest neighbor ensemble given the easy classification problem and 30 realizations of training data, each term's average weight is marked by the solid line.

All considered methods with variable selection properties perform well. By contrast, particularly ordinary nearest neighbor approaches without variable selection get into difficulties caused by noise variables.

In Figure 2 weights of terms in the nearest neighbor ensemble are shown by means of (degenerated) boxplots. Estimation was based on either the approximate logarithmic loss or the Brier score. The background indicates estimates based on a single predictor only, estimates using two predictors, and triple interactions respectively. As requested, the nearest neighbor estimate using x_1 only has the far highest weight in the ensemble. But when the quadratic Brier score was used for estimation, in a few iterations also less relevant predictions got some weight.

In Figure 3 we show weights of the terms in the ensemble when the difficult classification problem is analyzed. Variable selection is perfect. The most important term no. 56 is the prediction based on x_1, x_2 and x_3. So the good performance (see Figure 4) is not very surprising.

A forward selection procedure with $S \geq 3$ can be expected to select the relevant independent variables. But due to the complicated interaction of x_1, x_2 and x_3, the forward variable selection with $S = 1$ would not be successful. We chose $S = 4$ again. Also greedy algorithms like CART and the methods based on the latter like Random Forests are not able to identify the right variables. LDA as well is completely inappropriate, and as before nearest neighbor methods

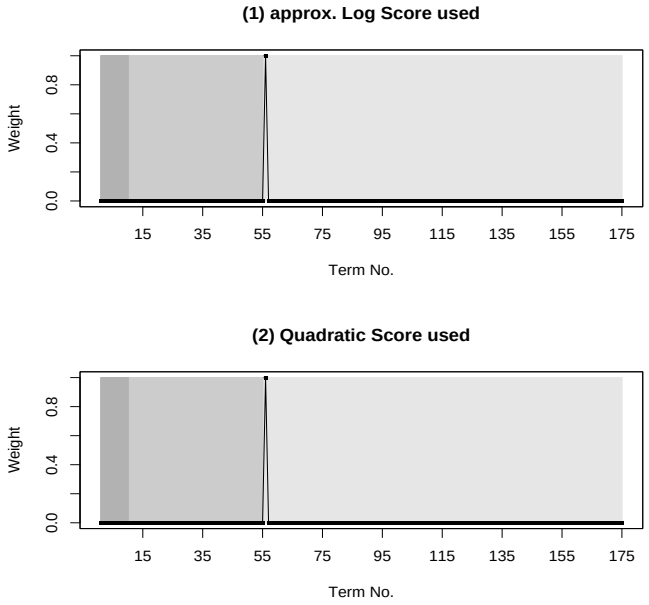

Figure 3: Boxplots of weights in the nearest neighbor ensemble given the difficult classification problem and 30 realizations of training data, each term's average weight is marked by the solid line.

without variable selection suffer from the presence of noise variables. In case of backward selection a large S seems to ensure that enough predictors are excluded to outperform nearest neighbor techniques based on the entire set of, partly noise, variables. As seen from Figure 3, when using ensemble classification in general the term based on x_1, x_2 and x_3 has the far highest weight; in every simulation run it is even the only prediction that is taken into account. So in the difficult classification problem on average nearest neighbor ensemble classification performs as well as forward or backward selection.

3.3 More Examples

Some further simulation scenarios are taken from Hastie and Tibshirani (1996). Each training set consists of $m = 200$ observations, but our test set size is $n = 1000$. Each class has the same number of observations. Exceptions from that rules are indicated. In detail the following classification problems are investigated, cf. Hastie and Tibshirani (1996):

1. **2 Dimensional Gaussian**: Two Gaussian classes in two dimensions (x_1, x_2) are separated by 2 units in x_1. The predictors have variance 1 and 2, and correlation 0.75.

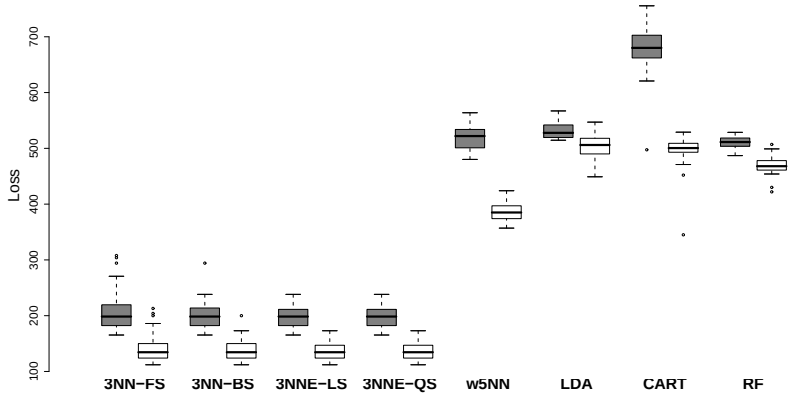

Figure 4: Boxplots of quadratic loss (dark-gray) and number of missclassified observations (light-colored) for 3 nearest neighbors with forward / backward variable selection ($S = 4$), 3 nearest neighbors ensemble based on approx. log score / quadratic score, weighted 5 nearest neighbors, LDA, CART and Random Forests (RF), given the difficult classification problem and 30 realizations of training and test set.

2. **2 Dimensional Gaussian with 14 Noise**: As before, but additionally 14 noise variables having independent standard Gaussian distributions are given.

3. **Unstructured**: In this example data with extremely disconnected class structure is simulated. There are 4 classes, each with 3 spherical bivariate normal subclasses, having standard deviation 0.25. The means of the resulting 12 subclasses are chosen at random (without replacement) from $\{1, \ldots, 5\} \times \{1, \ldots, 5\}$. Each training sample has 20, each test set 100 observations per subclass.

4. **Unstructured with 8 Noise**: As above, but augmented with 8 independent standard normal predictors.

5. **4 Dimensional Spheres with 6 Noise**: 10 predictors and 2 classes are given. The last 6 predictors are noise variables, with standard Gaussian distributions, independent of each other and the class label. The first 4 predictors in class 1 are independent standard normal, but conditioned on the radius being greater than 3, whereas the first 4 predictors in class 2 are independent standard normal without restrictions. The first class almost completely surrounds the second class in the 4 dimensional subspace of the first 4 predictors.

6. **10 Dimensional Spheres**: The situation is similar to the previous example. All 10 predictors in class 1 are independent standard normal, but now conditioned on the squared radius being between 22.4 and 40, while

the predictors in class 2 are again independent standard normal without restrictions. Now there are no pure noise variables. The second class is almost completely surrounded by the the first class in the full 10 dimensional feature space.

7. **Constant Class Probabilities**: A 4 class problem is given with class probabilities $(0.1, 0.2, 0.2, 0.5)$, but independent of the predictors. The latter are independent standard normal in 6 dimensions. Here the observed class frequencies may - of course - vary from one realization to another and are far away from being uniformly distributed.

8. **Friedman's example**: An example like this can be originally found in Friedman (1994). There are 2 classes in 10 dimensions. In class 1 the predictors are independent standard normal, in class 2 independent normal with mean and variance proportional to $\sqrt{j}$ and $1/\sqrt{j}$ respectively, $j = 1, \ldots, 10$. That means, all predictors are important, but those with higher index j are more so.

We compare the same classification methods as before, i.e. the proposed nearest neighbor ensemble, nearest neighbor based forward, resp. backward variable selection procedures, weighted 5 nearest neighbors, LDA, CART, and Random Forests. $S = 4$ is chosen (if possible) for forward as well as backward selection. In general the highest term in the nearest neighbor ensemble refers to an interaction of order 3 (if possible). To save some computational time in case of scenario 2 we only choose $S = 3$ and do not consider triple interactions. Figures 5 and 6 show the quadratic loss and the number of missclassified observations from the test set over 30 simulations.

Except the first and the last scenario the nearest neighbor ensemble based on the quadratic Brier score is among the best performing methods. Its performance is particularly high if some noise variables are given. Generally the Brier score gives better results than the approximate log score.

3.4 The Problem of Standardizing

In the previous sections we did not generally scale input data. Of course, some methods as LDA perform implicit scaling. The (weighted) k nearest neighbor algorithm from Hechenbichler and Schliep (2004) first by default standardizes all predictors to have unit variance over the training data, and the test set predictors are standardized by the corresponding training variance. The data preprocessing in Hastie and Tibshirani (1996) is similar. The motivation for scaling predictors in nearest neighbor procedures consists of two (related) points: (a) If a predictors's variance is very high, it may happen quite often that two observations show high differences in the corresponding direction. So the distance is strongly influenced by the corresponding term and the importance of the variable may be overrated.

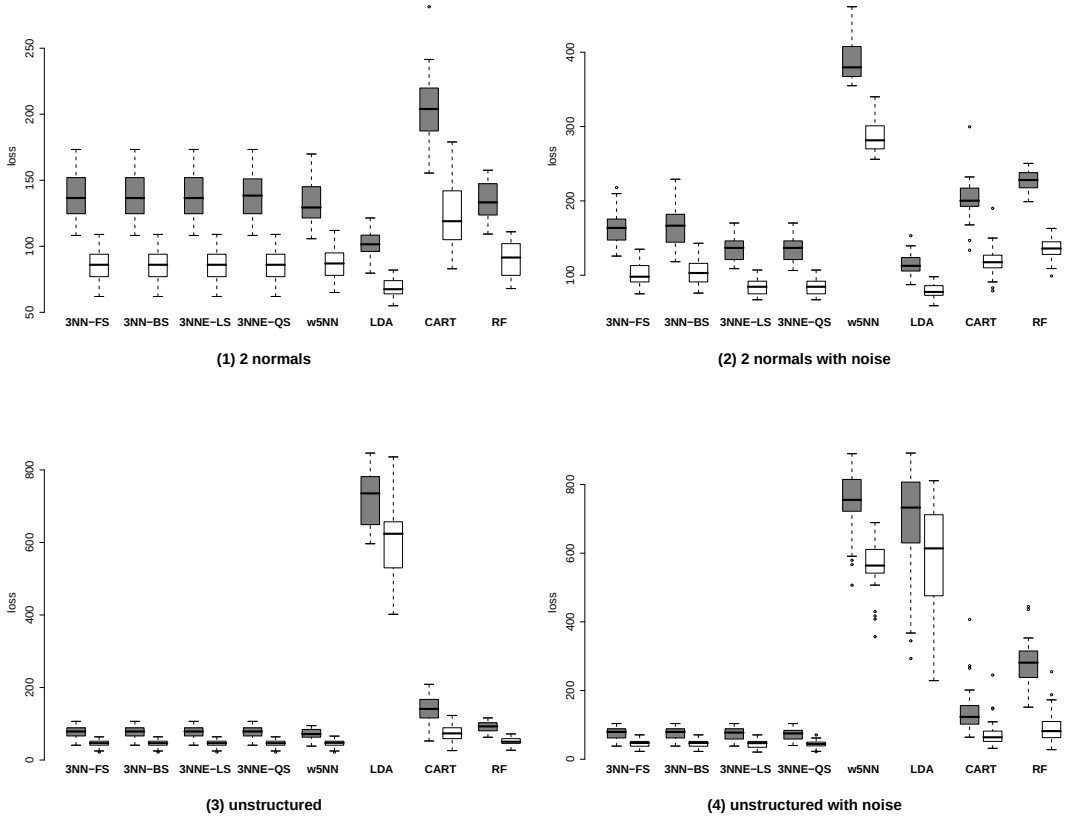

Figure 5: Boxplots of quadratic loss (dark-gray) and number of missclassified observations (light-colored) for 3 nearest neighbors with forward / backward variable selection, 3 nearest neighbor ensembles based on approx. log score / quadratic score, weighted 5 nearest neighbors, LDA, CART and Random Forests (RF); classification problems 1 to 4.

Since (b) the variance can be changed by simply changing the unit, e.g. from m to cm, standardizing protects against suchlike mistakes or manipulations.

Todeschini (1989) investigated the effect of different types of data transformations on nearest neighbor methods. We only consider autoscaling, i.e. scaling with respect to standard deviations, but focus on how these standard deviations are computed. To simply compute predictors' variances using the entire training data set without consideration of the class labels seems to be common practice. But it can be inappropriate to use these *overall* variances. For illustration consider the following situation: In class 1 the predictors x_1 and x_2 are independent normal with means -2 and -0.5 and variances 1 and 4. In Class 2 the variances are the same but means are 2 and 0.5. The training set consists of 50 observations in each class. In Figure 8 this situation is visualized. The two classes are almost perfectly separated by predictor x_1. Hence the squared distance

$$d^2(x_i, x_l) = (x_{i1} - x_{l1})^2 + (x_{i2} - x_{l2})^2$$

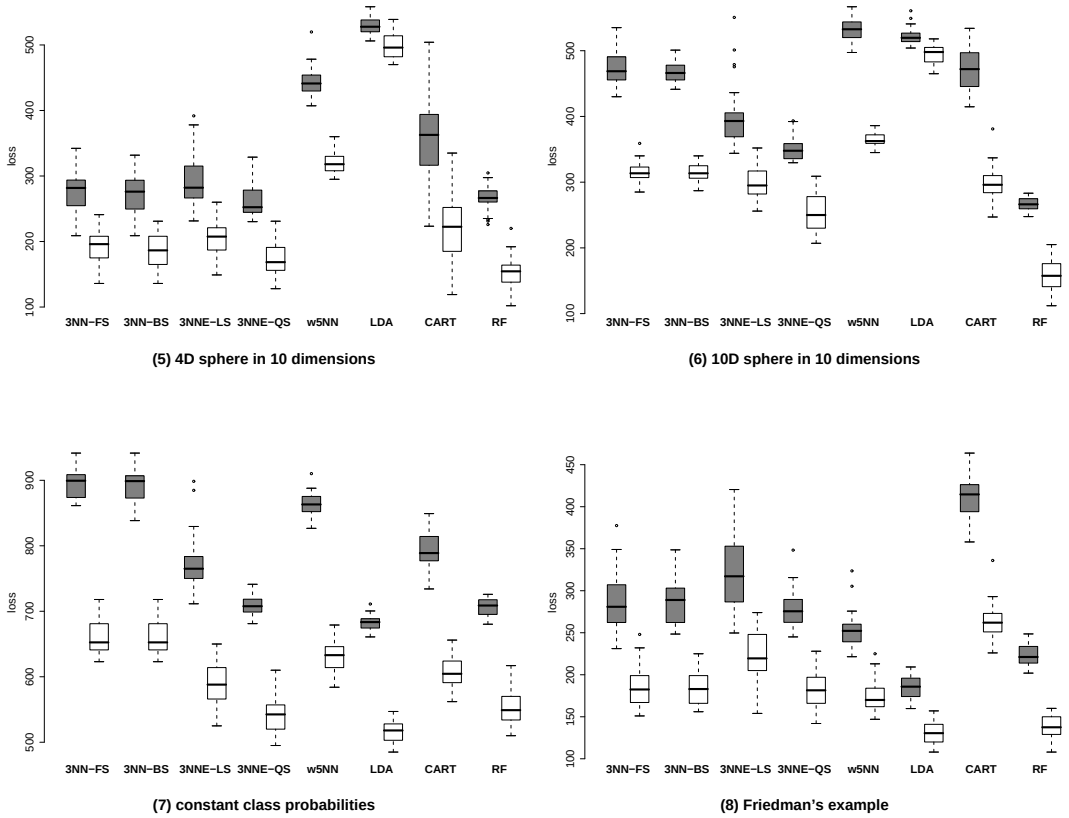

Figure 6: Boxplots of quadratic loss (dark-gray) and number of missclassified observations (light-colored) for 3 nearest neighbors with forward / backward variable selection, 3 nearest neighbor ensembles based on approx. log score / quadratic score, weighted 5 nearest neighbors, LDA, CART and Random Forests (RF); classification problems 5 to 8.

between two observations x_i and x_j should mainly depend on the difference in the x_1 direction. But dividing the observed x_1 values by the overall standard deviation from the mixture distribution causes a lower weight of x_1 - instead of the higher one desired. Our proposal is to use *pooled* variances instead of overall ones. The observed pooled, or *within*, variance $\hat{\sigma}^2_{(p)}$ of a predictor x given classes $g = 1, \ldots, G$ is defined as follows:

$$\hat{\sigma}^2_{(p)} = \frac{1}{n-G} \sum_{g=1}^{G} \sum_{i:y_i=g} (x_i - \bar{x}_{(g)})^2,$$

with y_i denoting the class label of observation i and $\bar{x}_{(g)}$ the mean of x in class g. In case of two classes having the same number of observations we simply have

$$\hat{\sigma}^2_{(p)} = (\hat{\sigma}^2_{(1)} + \hat{\sigma}^2_{(2)})/2,$$

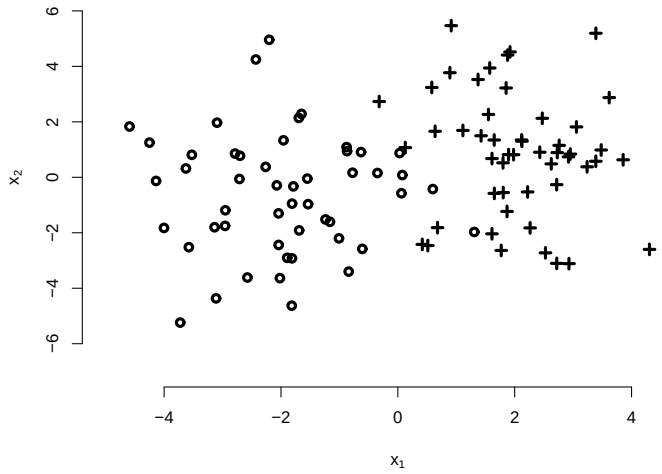

Figure 7: Illustration of a simple classification problem, observations from class 1 (∘) are separated very well from class 2 (+).

with $\hat{\sigma}^2_{(g)}$ denoting the inner class variance of class g. If the mean of x does not differ from one class to another, the pooled variance should be almost the same as the overall variance. But if differences do exist, which indicates one-dimensional discriminative capability, the pooled variance is the smaller one. Given the other predictors' means do not change between classes, the weight of x is increased compared to not scaling or to the use of overall variances. The danger of unit changing is avoided by employing $\hat{\sigma}^2_{(p)}$ as well.

To evaluate the consequences of different scaling approaches in the situation described above we generate a test set of 1000 observations and look at prediction accuracies. Figure 8 summarizes the results over 50 simulation runs. Since we are mainly interested in differences between scaling by overall variances and scaling by pooled variances we give relative values of quadratic loss (Brier score; left) and missclassification (right). Since for quadratic loss and missclassification the box is clearly below one, it can be stated that pooled variances are superior to overall ones. On average the former produces about 10% error less. Additionally values for no scaling are displayed on the left hand side of each plot in Figure 8. In the considered situation even no scaling seems to be better than scaling by overall variances.

A step further would be standardizing by a whole estimated pooled covariance matrix. If there are equal covariance matrices in the different groups, as assumed in Fisher's linear discriminant analysis, in each group the new predictors should approximately have unit variance while being independent of each other. But

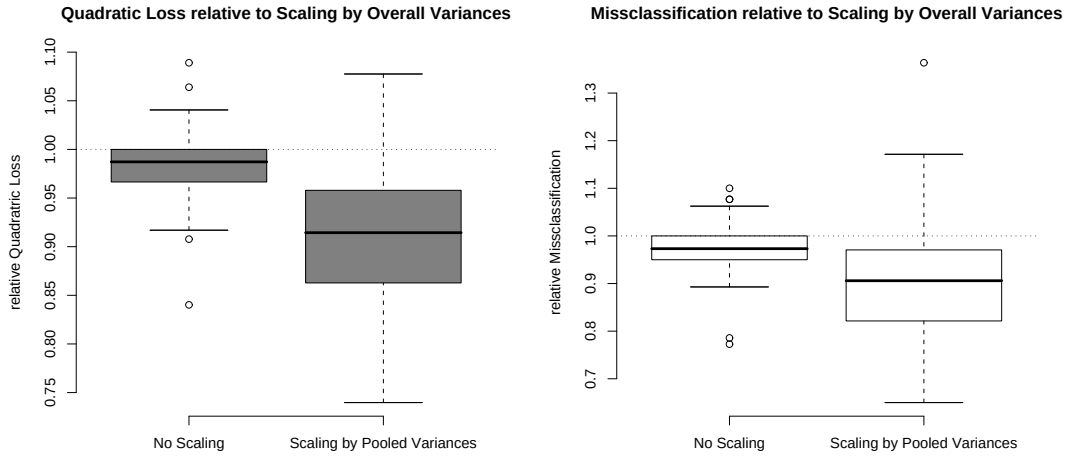

Figure 8: Quadratic loss (left) and missclassification (right) for no scaling and scaling by pooled variances; values relative to scaling by overall variances.

since in nearest neighbor procedures standardizing is only data preprocessing and each predictor normally has its own meaning, creating new predictors by linear combinations should be avoided. So the more restrictive standardizing by a diagonal matrix seems to be appropriate. For nearest neighbor ensembles scaling by pooled variances is used in the evaluation of real world data below.

4 Evaluation of Real World Data

When classification tools are visualized and compared this should not be done by simulations only, but also based on real world data sets. For that purpose we use the famous machine learning benchmark glass data set and data from the analysis of Italian olive oils.

4.1 Glass Data

The data can be obtained, for example, from the R add-on package `mlbench`: 214 observations are given containing examples of the chemical analysis of 6 different types of glass. The problem is to forecast the type of glass on the basis of the chemical analysis. The latter is given in form of 9 metric predictors: (1) refractive index, plus content of (2) Sodium, (3) Magnesium, (4) Aluminum, (5) Silicon, (6) Potassium, (7) Calcium, (8) Barium and (9) Iron, each measured in weight percent in the corresponding oxide. The possible types of glass are: (I) building windows (float processed), (II) building windows (non-float-processed), (III) vehicle windows, (IV) containers, (V) tableware, (VI) headlamps. The

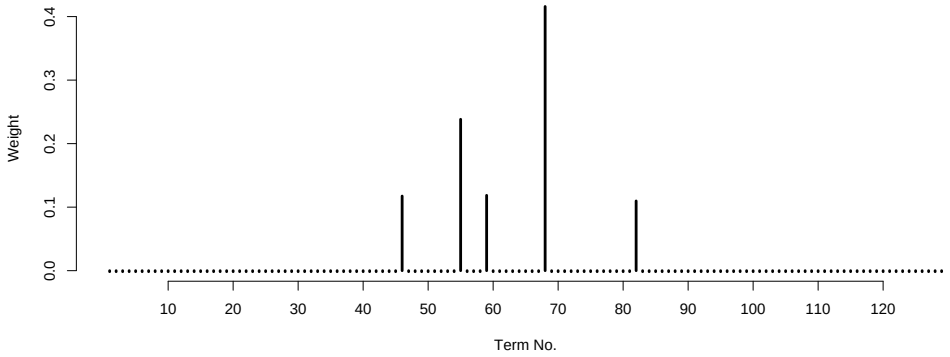

Figure 9: Composition of the nearest neighbor ensemble minimizing the Brier score and trained on the whole glass data set, terms no. 46 – 129 correspond to triple interactions.

data has also been considered by Breiman (2001), for example. It has originally been taken from the UCI Repository of Machine Learning Databases at `http://www.ics.uci.edu/~mlearn/MLRepository.html`. Such kind of study was motivated by criminological investigation; because at the scene of the crime, the glass left can be used as evidence, if it is correctly identified.

If all observations are used for training our nearest neighbor ensemble with hard-thresholding and $t = 0.25 \max_j\{c_j\}$, five terms are selected. The computed weights are seen in Figure 9. The minimized loss function is the Brier score. All selected terms correspond to triple interactions, but predictors number (8) and (9) are never included. That means Barium and Iron are not used for classification and may be excluded from discriminant analysis.

For evaluating the performance of the investigated classification tools (randomly chosen) 20% of the data is set aside. On the remaining data the methods are trained. The methods are those already compared in the simulation studies above: Nearest neighbor with forward and backward variable selection, nearest neighbor ensembles, weighted 5 nearest neighbors without variable selection, LDA, CART and Random Forests. We have $S = 3$, and the highest interaction in the nearest neighbor ensemble is set equal to S. Not explicitly mentioned tuning parameters have default values. Figure 10 shows the quadratic loss and number of missclassified observations on the test set over 50 random splits of the data at hand.

If the proposed nearest neighbor ensemble is based on the Brier score it is competitive to Random Forests, which have been shown to perform very well on this data set (see Breiman, 2001). Furthermore the ensemble is superior to the (weighted) standard nearest neighbor approach. Maybe, because there are some noise variables without discriminative power. For the better performance of the

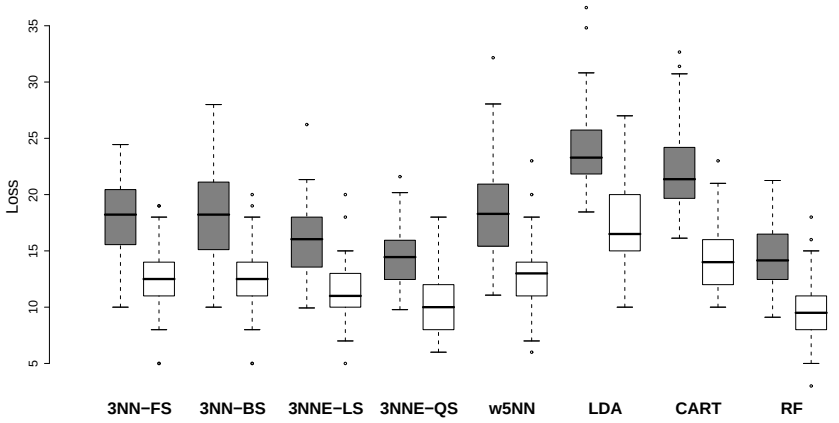

Figure 10: Boxplots of quadratic loss (dark-gray) and number of missclassified observations (light-colored) for 3 nearest neighbors with forward / backward variable selection, 3 nearest neighbor ensembles based on approx. log score / quadratic score, weighted 5 nearest neighbors, LDA, CART and Random Forests (RF), given the glass data and 50 random splits into training and test data.

ensemble, however, this is unlikely the only reason, since backward or forward selection does not improve nearest neighbor prediction.

4.2 Olives Data

The data is from Forina et al. (1982). The task is recognizing the geographical origin of Italian olives oils from their fatty acid composition. All in all the different oils are from nine regions, resp. classes: Calabria, Sicily, Umbria, Coast-Sardinia, Inland-Sardinia, North-Apulia, South-Apulia, East-Liguria, West-Liguria. Only eight predictors (i.e. fatty acids) are given; so in the nearest neighbor ensemble even estimates based on four covariates can be allowed. Since now more data points (572) than in the previous example are available, (randomly selected) 40% of the data are used as test set. The same classification tools as before are trained on the remaining data, and serve for discriminant analysis of the test data. As before, this procedure is repeated 50 times. The results in terms of the Brier score (dark-gray) and number of missclassified test observations are shown in Figure 11. The overall winner is the (weighted) standard nearest neighbor method taking into account all predictors at hand. Apparently all predictors have some predictive power. So no improvement can be expected, if some kind of variable selection is applied. Indeed when the nearest neighbor ensemble is trained on the whole data set (see Figure 12), only terms of order 4 are selected - covering all 8 predictors. Again the Brier score gives better results than the approximate logarithmic loss. So the former is used for the determination of weights in Figure 12.

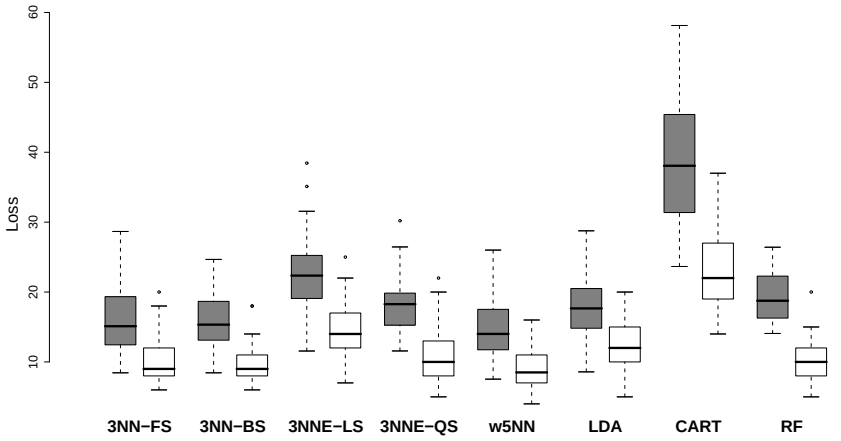

Figure 11: Boxplots of quadratic loss (dark-gray) and number of missclassified observations (light-colored) for 3 nearest neighbors with forward / backward variable selection, 3 nearest neighbor ensembles based on approx. log score / quadratic score, weighted 5 nearest neighbors, LDA, CART and Random Forests (RF), given the olives data and 50 random splits into training and test data.

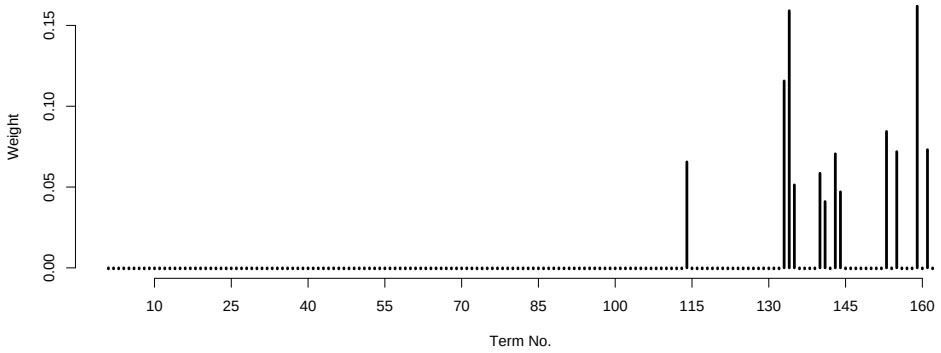

Figure 12: Composition of the nearest neighbor ensemble minimizing the Brier score and trained on the whole olives data set, terms no. 93 – 162 correspond to interactions of four predictors.

5 Summary and Discussion

We propose nonparametric probability estimation by an ensemble - i.e. weighted average - of nearest neighbor estimates. Each single estimate is based on a single or a very small subset of predictors. In contrast to many other ensemble approaches no randomness is included in the proposed method, e.g. by randomly selecting predictor sets. Instead all possible predictor combinations up to a previously chosen maximum number of predictors are taken as candidates, and weights are explicitly determined via minimization of a loss function - preferably the Brier score. By enforcing many zero weights the final ensemble only consists of a modest number of terms. As a result our ensemble is not a black box (by contrast to many other ensemble methods), but it is directly comprehensible how estimation is carried out. Covariates which are not contained in selected predictor sets do not serve for classification, or estimation of posterior probabilities. That means variable selection is explicitly done.

The proposed ensemble approach shows good performance for small scale problems, particularly if pure noise variables can be separated from relevant covariates. Easy identification of interactions is another advantage of the presented method. If the largest set of covariates is adequately chosen, even interactions that are usually hard to detect should be identified. So even if classification shall be done by another method, the ensemble may be used for variable selection.

Direct application of the proposed technique to high dimensional problems with interactions, however, is not recommended. If the number of potential predictors is high, interactions cannot be taken into account (because the number becomes too high). But given microarrays, in genetics for example, the presented ensemble might be useful as nonparametric gene preselection tool. Genes may be ranked according to their weight in the ensemble, and further analysis can be based on the *"best genes"* only.

Furthermore ensemble methodology may be applied for automatic choice of the most appropriate metrics, or semi-metrics (along the lines of Ferraty and Vieu, 2006); or the right neighborhood(s). For that purpose terms in the ensemble are nearest neighbor estimates based on different (semi-)metrics or different neighborhoods.

Finally, application to regression problems is possible as well. The concept of minimizing loss functions can be directly adopted, since the quadratic loss is the somewhat natural choice in regression problems.

Appendix

Proposition 1 *Given the following ensemble formula for computing the probability that observation i falls in category g:*

$$\hat{\pi}_{ig} = \sum_j c_{gj} \hat{\pi}_{ig(j)}, \text{ with } c_{gj} \geq 0 \; \forall g, j \text{ and } \sum_j c_{gj} = 1 \; \forall g;$$

restriction $c_{1j} = \ldots = c_{Gj} = c_j$ is the only possibility to ensure that $\hat{\pi}_{ig} \geq 0 \; \forall g$ and $\sum_g \hat{\pi}_{ig} = 1$ for all possible future estimations $\{\hat{\pi}_{ig(j)}\}$ with $\hat{\pi}_{ig(j)} \geq 0 \; \forall g, j$ and $\sum_g \hat{\pi}_{ig(j)} = 1 \; \forall j$.

Proof: We first show that the given restriction has the desired effect. Since $\hat{\pi}_{ig(j)} \geq 0 \; \forall g, j$, trivially follows $\sum_j c_j \hat{\pi}_{ig(j)} \geq 0$; furthermore

$$\sum_g \sum_j c_j \hat{\pi}_{ig(j)} = \sum_j c_j \sum_g \hat{\pi}_{ig(j)} = \sum_j c_j = 1.$$

In the next step we assume that c_{gj} vary over categories g for at least one j, i.e. coefficients c_{gj} can be ordered in terms of $c_{[1]j} \leq \ldots \leq c_{[G]j}$ with at least one $\leq$ being a $<$. Now we create groups $J_r = \{j | c_{rj} = \max_g c_{gj}\}$, $r = 1, \ldots, G$. If $\max_g c_{gj}$ is not unique for j, arbitrarily choose maximum c_{gj} to have disjoint J_r but covering all j.

Since $\sum_j c_{gj} = 1 \; \forall g$, there must be at least two j with coefficients differing over categories and $\max_g c_{gj}$ in different categories. Hence there are at least two nonempty sets J_{r_1} and J_{r_2}. Let the nonempty sets be denoted by $J_{r_1}, \ldots, J_{r_l}$. If now $\hat{\pi}_{ir_t(j)} = 1 \; \forall j \in J_{r_t}$ (which is not completely unlikely in case of nearest neighbor predictions), due to the $<$ above, it follows:

$$\sum_g \hat{\pi}_{ig} = \sum_g \sum_j c_{gj} \hat{\pi}_{ig(j)} = \sum_{j \in J_{r_1}} c_{r_1 j} + \ldots + \sum_{j \in J_{r_l}} c_{r_l j} > 1.$$

□

References

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

Breiman, L., J. H. Friedman, R. A. Olshen, and J. C. Stone (1984). *Classification and Regression Trees*. Monterey, CA: Wadsworth.

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review 78*, 1–3.

Cover, T. M. and P. E. Hart (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory 13*, 21–27.

Domeniconi, C. and B. Yan (2004). Nearest neighbor ensemble. In *Proceedings of the 17th International Conference on Pattern Recognition, Cambridge, UK*.

Efron, B., T. Hastie, I. Johnstone, and R. Tibshirani (2004). Least angle regression. *The Annals of Statistics 32*, 407–499.

Ferraty, F. and P. Vieu (2006). *Nonparametric Functional Data Analysis*. New York: Springer.

Fix, E. and J. L. Hodges (1951). Discriminatory analysis - nonparametric discrimination: consistency properties. US air force school of aviation medicine, Randolph Field Texas.

Forina, M., C. Armanino, S. Lanteri, and E. Tiscornia (1982). Classifiaction of olive oils from their fatty acid composition. In H. Martens and H. Russwurm (Eds.), *Food Research and Data Analysis, Proceedings from the IUFoST Symposium*, pp. 189–214. London/New York: Applied Science Publishers LTD.

Friedman, J. H. (1994). Flexible metric nearest neighbor classification. Technical report, Stanford University.

Hastie, T. and R. Tibshirani (1996). Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence 18*, 607–616.

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

Hechenbichler, K. and K. Schliep (2004). Weighted k-nearest-neighbor techniques and ordinal classification. SFB 386 Discussion Paper 399, Ludwig-Maximilians-Universität München.

Kudo, M. and J. Sklansky (2000). Comparison of algorithms that selected features for pattern classifiers. *Pattern Recognition 33*, 25–41.

Le Cessie, S. and J. C. van Houwelingen (1992). Ridge estimators in logistic regression. *Applied Statistics 41*, 191–201.

Liaw, A. and M. Wiener (2002). Classification and regression by randomforest. *R News 2*(3), 18–22.

R Development Core Team (2007). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. ISBN 3-900051-07-0.

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge: University Press.

Selten, R. (1998). Axiomatic characterization of the quadratic scoring rule. *Experimental Economics 1*, 43–62.

Silverman, B. W. and M. C. Jones (1989). Commentary on Fix and Hodges (1951): An important contribution to nonparametric discriminant analysis and density estimation. *International Statistical Review 57*, 233–238.

Therneau, T. M. and B. Atkinson (2007). *rpart: Recursive Partitioning*. R package version 3.1-36, R port by Brian Ripley.

Todeschini, R. (1989). k-nearest neighbor method: the influence of data transformations and metrics. *Chemometrics and Intelligent Laboratory Systems 6*, 213–220.

Turlach, B. A. (2007). *quadprog: Functions to solve Quadratic Programming Problems*. R package version 1.4-11, S original by Berwin A. Turlach, R port by Andreas Weingessel.

Venables, W. N. and B. D. Ripley (2002). *Modern Applied Statistics with S* (Fourth ed.). New York: Springer.

Yankov, D., D. DeCoste, and E. Keogh (2006). Ensembles of nearest neighbor forecasts. In J. Fürnkranz, T. Scheffer, and M. Spiliopoulou (Eds.), *Lecture Notes in Computer Science, Proceedings of the 17th European Conference on Machine Learning*, pp. 545–556. Berlin/Heidelberg: Springer-Verlag.