Hindawi Publishing Corporation
Journal of Biomedicine and Biotechnology
Volume 2008, Article ID 218097, 7 pages
doi:10.1155/2008/218097

Research Article

Classification Models for Early Detection of Prostate Cancer

Joerg D. Wichard,[1, 2] Henning Cammann,[1] Carsten Stephan,[3] and Thomas Tolxdorff[1]

[1] *Institute of Medical Informatics, Charité - Universitätsmedizin, Hindenburgdamm 30, 12200 Berlin, Germany*
[2] *Molecular Modelling Group, Institut für Molekulare Pharmakologie, Robert Rössle Straße 10, 13125 Berlin, Germany*
[3] *Department of Urology, Charité - Universitätsmedizin, Charitéplatz 1, 10098 Berlin, Germany*

Correspondence should be addressed to Joerg D. Wichard, joergwichard@web.de

Received 12 September 2007; Accepted 2 January 2008

Recommended by Daniel Howard

We investigate the performance of different classification models and their ability to recognize prostate cancer in an early stage. We build ensembles of classification models in order to increase the classification performance. We measure the performance of our models in an extensive cross-validation procedure and compare different classification models. The datasets come from clinical examinations and some of the classification models are already in use to support the urologists in their clinical work.

1. INTRODUCTION

Prostate cancer is one of the most common types of cancer among male patients in the western world. The number of expected new cases in the USA for the year 2006 was 235,000 with 27,000 expected deaths [1]. Early detection of prostate cancer improves the chances of a curative treatment and a lot of progress has been made in this field during the last decade. The early detection is considerably enhanced by the measurement of prostate-specific antigen (PSA) in conjunction with other clinically available data like age, digital rectal examination (DRE), and transrectal ultrasonography (TRUS) variables like prostate volume. We compared several classification models and analyzed their performance on the clinical dataset with an extended cross-validation procedure. The models were linear discriminant analysis (LDA), penalized discriminant analysis (PDA) [2], logistic regression [3], classification and regression trees (CARTs) [4], multilayer perceptron (MLP) [5], support vector machines (SVMs) [6, 7], and nearest neighbour classifiers [8]. All these models are implemented in an open-source Matlab-toolbox that is available on the internet [9].

This study will help to improve the software package *ProstataClass* [10] which was developed at Charité and currently uses an artificial neural network as classification engine. This program is successfully used in clinical practice for several years.

2. DATA

We had access to the clinically available data of 506 patients with 313 cases of prostate cancer (PCa) and 193 non-PCa. The data were selected from a group of 780 patients randomly. The data entry for each patient included age, PSA, the ratio of free to total prostate-specific antigen (PSA-Ratio), TRUS, and the diagnostic finding from the DRE which was a binary variable (suspicious or nonsuspicious). Blood sampling and handling were performed as described in Stephan et al. [11]. The samples were taken before any diagnostic or therapeutic procedures, and sera were stored at 80°C until analyzed. After thawing at room temperature, samples were processed within 3 hours. Prostate volume was determined by transrectal ultrasound using the prolate ellipse formula. The scatter plot of the variables under investigation is shown in Figure 1. PCa and non-PCa patients were histologically confirmed by 6–8 core prostate biopsy.

3. ENSEMBLES

The average output of several different models $f_i(x)$ is called an ensemble model:

$$\hat{f}(x) = \sum_{i=1}^{K} \omega_i f_i(x), \qquad (1)$$

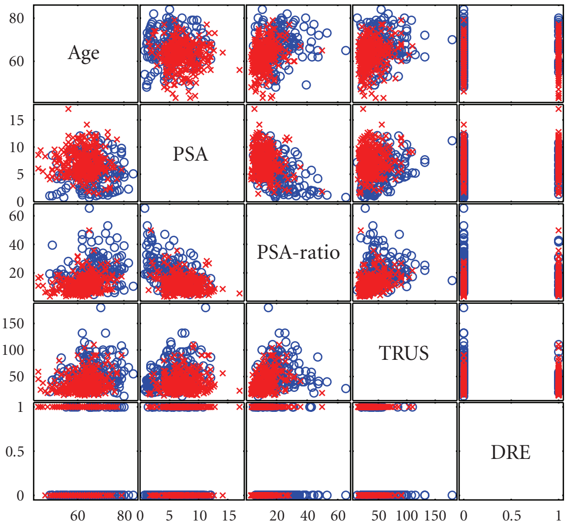

Figure 1: A scatterplot matrix of the data. Each box shows a pair of variables and the cases are color-coded, a red cross marks PCa, and a blue circle non-PCa. The DRE is a binary variable (suspicious or nonsuspicious).

where we assume that the model weights ω_i sum to one $\sum_{i=1}^{K}\omega_i = 1$. There are several suggestions concerning the choice of the model weights (see Perrone and Cooper [12]) but we decided to use uniform weights with $\omega_i = 1/K$ for the sake of simplicity and not to run into overfitting problems as reported by Krogh and Sollich [13].

The central feature of the ensemble approach is the generalization ability of the resulting model. In the case of regression models (with continuous output values), it was shown that the generalization error of the ensemble is in the average case lower than the mean of the generalization error of the single-ensemble members (see Krogh and Vedelsby 1995 [14]). This holds in general, independent of the model class, as long as the models constituting the ensemble are diverse with respect to the hypothesis of the unknown function. In the case of (binary) classification models, the situation was not so clear because the classical bias-variance decomposition of the squared error loss in regression problems (Geman et al. [15]) had to be extended to the zero-one loss function. There are several approaches dealing with this problem, see Kong and Dietterich [16], Kohavi and Wolpert [17], or Domingos [18].

The zero-one loss function is not the only possible choice for classification problems. If we are interested in a likelihood whether a sample belongs to one class or not, we can use the error loss from regression and consider the binary classification problem as a regression problem that works on two possible outcomes. In practice, many classifiers are trained in that way.

Our ensemble approach is based on the observation that the generalization error of an ensemble model could be improved if the models on which averaging is done disagree and if their residual errors are uncorrelated [13]. To see this, we have to investigate the contribution of the single model in the ensemble to the generalization error. We consider the case where we have a given dataset $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ and we want to find a function $f(\mathbf{x})$ that approximates y at new observations of $\mathbf{x}$. These observations are assumed to come from the same source that generated the training set D, that is, from the same (unknown) probability distribution P. It should be noted that f depends also on D. The expected generalization error $\mathrm{Err}(\mathbf{x}, D)$ given a particular $\mathbf{x}$ and a training set D is

$$\mathrm{Err}(\mathbf{x}, D) = E\Big[\big(y - f(\mathbf{x})\big)^2 \mid \mathbf{x}, D\Big], \tag{2}$$

where the expectation $E[\cdot]$ is taken with respect to the probability distribution P. We are now interested in

$$\mathrm{Err}(\mathbf{x}) = E_D[\mathrm{Err}(\mathbf{x}, D)], \tag{3}$$

where the expectation $E_D[\cdot]$ is taken with respect to all possible realizations of training sets D with fixed sample size N. According to Geman et al. [15], the bias/variance decomposition of $\mathrm{Err}(\mathbf{x})$ is

$$\begin{aligned}\mathrm{Err}(\mathbf{x}) &= \sigma^2 + \left(E_D[f(\mathbf{x})] - E[y \mid \mathbf{x}]\right)^2 \\ &\quad + E_D\Big[\big(f(\mathbf{x}) - E_D[f(\mathbf{x})]\big)^2\Big] \\ &= \sigma^2 + \mathrm{Bias}\big(f(x)\big)^2 + \mathrm{Var}\big(f(x)\big),\end{aligned} \tag{4}$$

where $E[y \mid \mathbf{x}]$ is the deterministic part of the data and σ^2 is the variance of y given $\mathbf{x}$. Balancing between the bias and the variance terms is a crucial problem in model building. If we try to decrease the bias term on a specific training set, we usually increase the variance term and vice versa. We now consider the case of an ensemble average $\hat{f}(\mathbf{x})$, consisting of K individual models as defined in (1). If we put this into (4), we get

$$\mathrm{Err}(\mathbf{x}) = \sigma^2 + \mathrm{Bias}\big(\hat{f}(x)\big)^2 + \mathrm{Var}\big(\hat{f}(x)\big), \tag{5}$$

and we can have a look at the effects concerning bias and variance. The bias term in (5) is just the average of the biases of the individual models in the ensemble. So we should not expect a reduction in the bias term compared to single models. According to Naftaly et al. [19], the variance term of the ensemble could be decomposed in the following way:

$$\begin{aligned}\mathrm{Var}\big(\hat{f}\big) &= E\Big[\big(\hat{f} - E\big[\hat{f}\big]\big)^2\Big] \\ &= E\left[\left(\sum_{i=1}^{K}\omega_i f_i\right)^2\right] - \left(E\left[\sum_{i=1}^{K}\omega_i f_i\right]\right)^2 \\ &= \sum_{i=1}^{K}\omega_i^2\left(E[f_i^2] - E^2[f_i]\right) \\ &\quad + 2\sum_{i<j}\omega_i\omega_j\left(E[f_i f_j] - E[f_i]E[f_j]\right),\end{aligned} \tag{6}$$

where the expectation is taken with respect to D. The first sum in (6) marks the lower bound of the ensemble variance and is the weighted mean of the variances of the ensemble members. The second sum contains the cross terms of the ensemble members and disappears if the models are completely uncorrelated [13]. So the reduction in the variance of the ensemble is related to the degree of independence of the single models [19].

4. CROSS-VALIDATION AND MODEL SELECTION

Our model selection scheme is a mixture of bagging [20] and cross-validation. *Bagging* or *Bootstrap aggregating* was proposed by Breiman [20] in order to improve the classification by combining classifiers trained on randomly generated subsets of the entire training sets. We extended this approach by applying a cross-validation scheme for model selection on each subset and after that we combine the selected models to an ensemble. In contrast to classical cross-validation, we use random subsets as cross-validation folds. In K-fold cross-validation, the dataset is partitioned into K subsets. Of these K subsets, a single subset is retained as the validation data for testing the model, and the remaining $K - 1$ subsets are used for model training. The cross-validation process is then repeated K times with each of the K subsets used only once as the validation data. The K results from the folds then can be averaged to produce a single estimation.

If we lack relevant problem-specific knowledge, cross-validation methods could be used to select a classification method empirically [21]. This is a common approach because it seems to be obvious that no classification method is uniformly superior, see, for example, Quinlan [22] for a detailed study. It is also a common approach to select the model parameters with cross-validation [23]. The idea to combine the models from the K cross-validation folds (stacking) was described by Wolpert [24].

We suggest to train several models on each CV-fold, to select the best performing model on the validation set, and to combine the selected models from the K-folds. If we train models of one type but with different initial conditions (e.g., neural networks with different numbers of hidden neurons), then we find proper values for the free parameters of the model. We could extend that by combining models from different classes in order to increase the model diversity. We call this a *heterogeneous ensemble* or *mixed ensemble* and applied this method effectively to regression problems [25] and classification tasks [26].

Our model selection scheme works as follows: for the K-fold CV, the data is divided K-times into a *training set* and a *test set*, both sets containing randomly drawn subsets of the data without replications. The size of each test set was 25% of the entire dataset.

In every CV-fold, we train several different models with a variety of model parameters (see Section 5 for an overview of the models and the related model parameters). In each fold, we select only one model to become a member of the final ensemble (namely, the best model with respect to the test set). This means that all models have to compete with each other in a fair tournament because they are trained and validated on the same dataset. The models with the lowest classification error in each CV-fold are taken out and added to the final ensemble, receiving the weight $\omega_i = 1/k$ (see (1)). All other models in this CV-fold are deleted.

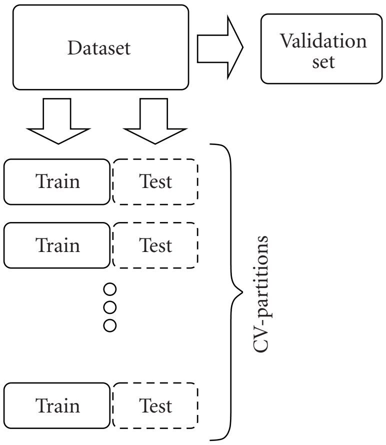

FIGURE 2: For every partition of the cross-validation, the data is divided in a training and a test set. The performance of each ensemble model was assessed on validation set which was initially removed and never included in model training.

We can use this model selection scheme in two ways. If we have no idea or prior knowledge, where classification or regression method should be used to cope with a specific problem, we could use this scheme in order to look for an empirical answer and to compare the performance of the different model classes. The other way is the estimation of model parameters for the different model classes described in Section 5.

5. CLASSIFICATION MODELS

In this section, we give a short overview of the model classes that we used for ensemble building. All models belong to the well-established collection of machine-learning algorithms for classification and regression tasks, so details can be found in the textbooks like, for instance, Hastie et al. [2]. The implementation of these models in an open-source toolbox together with a more detailed description can be found in [9]. The toolbox is an open-source MATLAB Toolbox which allows the integration of existing implementations of classification algorithms and it contains more then the few model classes described in the text.

5.1. *Linear discriminant analysis*

The LDA is a simple but useful classifier. If we assume that the two classes $k = \{0, 1\}$ have a Gaussian distribution with mean μ_k and they share the same covariance matrix Σ, then the *linear discriminant function* $\delta_k(\mathbf{x})$, $k = \{0, 1\}$ is given by

$$\delta_k(\mathbf{x}) = \mathbf{x}^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \log(\pi_k), \qquad (7)$$

where π_k denotes the frequency of occurrence of the class labels. The predicted class labels are given by

$$f(\mathbf{x}) = \arg\max_{k=(0,1)}\{\delta_k(\mathbf{x})\}. \qquad (8)$$

We also implemented two modifications: the quadratic discriminant analysis (QDA) and the PDA, as described in detail in Hastie et al. [2]. Linear method are usually conceptually simple, robust, fast, and, in particular in high-dimensional problems, they could be very powerful.

5.2. Logistic regression model

Logistic regression (Log.Reg.) is a model for binomial distributed dependent variables and is used extensively in the medical and social sciences. Hastie et al. [2] pointed out that the Logistic Regression model has the same form as the LDA, the only difference lies in the way, the linear coefficients are estimated. See Hosmer and Lemeshow for a detailed introduction [27]. We used the binary Log.Reg. to compute the probability of the dichotomic variable y (PCa or non-PCa) from the m independent variables $\mathbf{x}$:

$$y = \frac{1}{1 + \exp(\mathbf{z})} \tag{9}$$

with

$$\mathbf{z} = a_0 + \sum_{i=1}^{m} a_i x_i, \tag{10}$$

wherein the model coefficients are estimated with a second-order gradient decent (quadratic approximation to likelihood function). This could be a critical issue in high-dimensional problems because these calculations are time and memory consuming.

5.3. Multilayer perceptron

We train a multilayer feed-forward neural network "MLP" with a sigmoid activation function. The weights are initialized with Gaussian-distributed random numbers having zero mean and scaled variances. The weights are trained with a gradient descend based on the Rprop algorithm [28] with the improvements given in [29]. The MLP works with a common weight decay with the penalty term

$$P(\vec{w}) = \lambda \sum_{i=1}^{N} - \frac{w_i^2}{1 + w_i^2}\,, \tag{11}$$

where $\vec{w}$ denotes the N-dimensional weight vector of the MLP and a small regularization parameter λ. The number of hidden layers, the number of neurons, and the number of regularization parameter are adjusted during the CV-training. We further applied the concept of an ϵ-insensitive error loss that we introduced in the context of cellular neural networks (CNNs) [30].

5.4. Support vector machines

Over the last decade, SVMs have become very powerful tools in machine learning. An SVM creates a hyperplane in a feature space that separates the data into two classes with the maximum margin. The feature space can be a mapping of the original features $(\mathbf{x}, \mathbf{x}')$ into a higher-dimensional space using a positive semidefinite function:

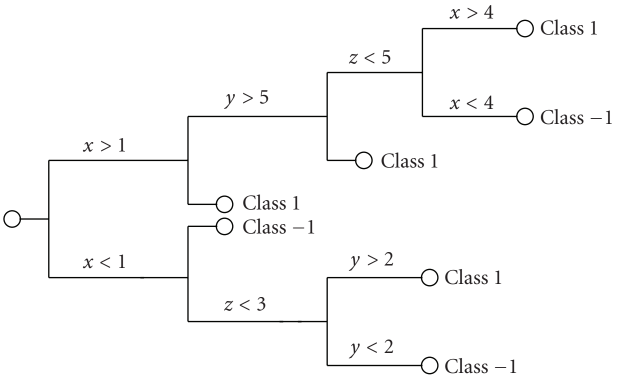

Figure 3: A sketch of a classification tree, wherein the leaves represent classes and the branches represent conjunctions of features that lead to those classes.

$$(\mathbf{x}, \mathbf{x}') \longmapsto k(\mathbf{x}, \mathbf{x}'). \tag{12}$$

The function $k(\cdot, \cdot)$ is called the *kernel function* and the so-called *kernel trick* uses Mercer's condition, which states that any positive semidefinite kernel $k(\mathbf{x}, \mathbf{x}')$ can be expressed as a dot product in a highdimensional space (see [31] for a detailed introduction). The theoretical foundations of this approach were given by Vapnik's *statistical learning theory* [6, 32] and later extended to the nonlinear case [7]. We use an implementation of SVMs that is based on the libsvm provided by Chang and Lin [33] with the standard kernels:

$$\begin{aligned} k(\mathbf{x}, \mathbf{x}') &= (\mathbf{x} \cdot \mathbf{x}') \text{ linear} \\ &= (\mathbf{x} \cdot \mathbf{x}' + 1)^d \text{ polynomial} \\ &= \exp\left(- \frac{\|\mathbf{x} - \mathbf{x}'\|}{\sigma^2} \right) \text{ rbf} \end{aligned} \tag{13}$$

The parameters of the model are, with respect to the kernel-type, the polynomial degree d, the width of the rbf σ^2 and the value concerning the cost of constrain violation during the SVM training.

5.5. Trees

Trees are conceptually simple but powerful tools for classification and regression. For our purpose, we use the *classification and regression trees* (CARTs) as described in Breiman et al. [4]. The main feature of the CART algorithm is the binary decision role that is introduced at each tree node with respect to the information content of the split. In this way, the most discriminating binary splits are near the tree root building an hierarchical decision scheme. A sketch of a decision tree is shown in Figure 3. It is known that trees have a high variance, so they benefit from the ensemble approach [20]. These trees ensembles are also know as *random forests*. The parameters of the tree models are related to splitting the tree nodes (the impurity measure and the split criterion, see [2] for a detailed description).

5.6. Nearest-neighbor classifier

A K-nearest-neighbor classifier (KNN) takes a weighted average over the labels z_i of those observations $\mathbf{z}_i$ in the training set that are closest to the query point $\mathbf{x}$. This denotes as

$$f(\mathbf{x}) = \frac{1}{\sum w_i} \sum_{\mathbf{z}_i \in N_k(\mathbf{x})} w_i z_i, \tag{14}$$

where $N_k(\mathbf{x})$ denotes the k-element neighborhood of $\mathbf{x}$, defined in a given metric, and w_i is the related distance. Common choices are the L_1, L_2, and the L_∞ metrics. The parameters of the model are the number of neighbors and the choice of the metric. KNNs offer a very intuitive approach to classification problems because they are based on the concept of similarity. This works fine in lower dimensions but leads to problems in higher dimensions, known as the *curse of dimensionality* [34].

6. APPLICATION TO THE CLINICAL DATA

We compared the model classes described above in a unified framework under fair conditions. Thus, we trained an ensemble of each model class consisting of 11 ensembles members (11 CV-folds in the training scheme described in Section 4). The performance of each ensemble model was assessed on the 20% of data (validation set), which was initially removed and never included in model training (see Figure 2). This procedure was independently repeated 20 times. This means that all model-building processes, that is, the random removal of 20% of the data, the construction of a classification model ensemble on the remaining 80% of the data as outlined in Section 4, and the final test on the unseen validation data were performed each time. Finally, the mean average prediction values with respect to the validation set were calculated and are listed in Table 2. In some cases, it is useful to apply a kind of data preprocessing like balancing. If the distribution of the two classes differ in the sense, that one class is only represented with a small number of examples, we can balance the data in the training set. This can improve the convergence of several training algorithms and has also an impact to the classification error [35]. We apply balancing in the way that we reduce the number of samples in the one class until we have an balanced ratio of the class labels. The ratio of the class labels in the validation set was never changed because it reflects the real data distribution. Balancing was only applied to the training data. We used three different performance measures in order to compare the different classification models. Therefore, we have to define the four possible outcomes of a classification that can be formulated in a 2 × 2 confusion matrix, as shown in Table 1. The accuracy,

$$\text{Accuracy} = \frac{\text{tp} + \text{tn}}{\text{tp} + \text{tn} + \text{fp} + \text{fn}}, \tag{15}$$

seems to be the canonical error measure for almost all classification problems if the dataset is balanced. Other important measures are the specificity that quantifies how well a binary classification model correctly identifies the negative cases (non-PCa patients),

Table 1: The confusion matrix for a binary classification problem.

	predicted class + 1	predicted class − 1
Real class + 1	True positive (tp)	False negative (fn)
Real class − 1	False positive (fp)	True negative (tn)

$$\text{Specificity} = \frac{\text{tn}}{\text{tn} + \text{fp}}, \tag{16}$$

and the sensitivity, which is the proportion of true positives of all diseased cases (PCa patients) in the population,

$$\text{Sensitivity} = \frac{\text{tp}}{\text{tp} + \text{fn}}. \tag{17}$$

A high sensitivity is required when early diagnosis and treatment are beneficial, which is the case in PCa.

The precision or positive predictive value (PPV) is given by

$$\text{PPV} = \frac{\text{tp}}{\text{tp} + \text{fp}}, \tag{18}$$

and is the proportion of patients with positive test results who are correctly diagnosed. The F-Score is the harmonic mean of precision and sensitivity,

$$\text{F-Score} = 2 \cdot \frac{\text{Sensitivity} \cdot \text{PPV}}{\text{Sensitivity} + \text{PPV}}, \tag{19}$$

and it is useful if the classes in the classification problem are not equally distributed. Another measure is the area under curve (AUC) wherein the curve is the receiver operating characteristic (ROC-curve)curve. The ROC-curve is the graphical plot of the sensitivity versus the (1-specificity) for a binary classifier as its discrimination threshold is varied.

The ROC-curve offers the opportunity to calculate the specificity at a fixed sensitivity level and vice versa. This is important because, from the clinical point of view, a high sensitivity 95% is wanted to detect all patients with PCa first. To avoid a high false-positive rate, we computed the specificity at the level of 95% sensitivity (SPS95) from the ROC-curve as another important performance measure.

To have an impression about the correct classified non-PCa patients in this case, we computed the specificity at the level of 95% sensitivity (SPS95) from the ROC-curve. If we compare the outcome of the statistical analysis of the model performance as listed in Table 1 for the unbalanced case and in Table 2 for the balanced case, we can state that the differences between the different classifiers are marginal. Even the more sophisticated classification models (SVMs or Mixed Ensembles) could not outperform the robust linear candidates (LDA/PDA).

Tables 2 and 3 present the main results with only small differences between the classifiers. The standard deviations of the performance measures are given except for the ROC-curve-based measures (AUC and SPS95). Most papers in

Table 2: The average performance of several classifier ensembles with respect to the validation set which was initially removed and never included in model training. We show the mean and the standard deviation values from 20 independent validation runs, no preprocessing was used.

	Accuracy	F-score	AUC	SPS95
PDA	0.776 ± 0.026	0.823 ± 0.026	0.863	0.454
Log.Reg.	0.778 ± 0.038	0.823 ± 0.036	0.868	0.484
MLP	0.791 ± 0.045	0.823 ± 0.04	0.863	0.453
SVM	0.795 ± 0.023	0.833 ± 0.02	0.825	0.142
CART	0.757 ± 0.03	0.809 ± 0.026	0.843	0.394
KNN	0.756 ± 0.036	0.813 ± 0.032	0.809	0.309
Mixed	0.783 ± 0.03	0.828 ± 0.026	0.860	0.457

Table 3: The average performance of several classifier ensembles with respect to the validation set which was initially removed and never included in model training. We show the mean and the standard deviation values from 20 independent validation runs wherein the training data was balanced.

	Accuracy	F-score	AUC	SPS95
PDA	0.772 ± 0.034	0.809 ± 0.035	0.861	0.414
Log.Reg.	0.792 ± 0.03	0.834 ± 0.027	0.868	0.458
MLP	0.766 ± 0.027	0.787 ± 0.029	0.858	0.451
SVM	0.786 ± 0.038	0.816 ± 0.042	0.821	0.051
CART	0.755 ± 0.031	0.792 ± 0.029	0.841	0.376
KNN	0.726 ± 0.032	0.766 ± 0.034	0.801	0.297
Mixed	0.789 ± 0.033	0.830 ± 0.026	0.867	0.445

this field do not discuss this really complex problem and it cannot be solved in the scope of this paper, but it should be mentioned. As an example of a special solution of this problem, see the paper of Hilgers [36].

7. CONCLUSIONS

We compared several classification models with respect to their ability to recognize prostate cancer in an early stage. This was done in an ensemble framework in order to estimate proper model parameters and to increase classification performance. It turned out that all models under investigation are performing very well with only marginal differences and are compareable with similar studies, like, for example, Finne et al. [37], Remzi et al. [38], or Zlotta et al. [39]. In future research, it should be investigated whether these results are valid for other populations of patients (e.g., screening data) and other PSA test assays and whether the performance of classification could be increased by including new variables or by splitting the groups of patients into different PSA ranges.

REFERENCES

[1] A. Jemal, R. Siegel, E. M. Ward, and M. J. Thun, “Cancer facts & figures,” Tech. Rep., Department of Epidemiology and Surveillance Research, American Cancer Society, Atlanta, Ga, USA, 2006.

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer Series in Statistics, Springer, New York, NY, USA, 2001.

[3] Y. Bard, *Nonlinear Parameter Estimation*, Academic Press, New York, NY, USA, 1974.

[4] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*, Wadsworth and Brooks, Monterey, Calif, USA, 1993.

[5] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, UK, 1995.

[6] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, USA, 1999.

[7] B. E. Boser, I. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the 5th Annual ACM Conference on Computational Learning Theory (COLT ’92)*, pp. 144–152, Pittsburgh, Pa, USA, July 1992.

[8] C. Merkwirth, U. Parlitz, and W. Lauterborn, “Fast nearest-neighbor searching for nonlinear signal processing,” *Physical Review E*, vol. 62, no. 2, pp. 2089–2097, 2000.

[9] J. D. Wichard and C. Merkwirth, “ENTOOL—A Matlab toolbox for ensemble modeling,” http://www.j-wichard.de/entool/, 2007.

[10] C. Stephan, H. Cammann, A. Semjonow, et al., “Multicenter evaluation of an artificial neural network to increase the prostate cancer detection rate and reduce unnecessary biopsies,” *Clinical Chemistry*, vol. 48, no. 8, pp. 1279–1287, 2002.

[11] C. Stephan, M. Klaas, C. Müller, D. Schnorr, S. A. Loening, and K. Jung, “Interchangeability of measurements of total and free prostate-specific antigen in serum with 5 frequently used assay combinations: an update,” *Clinical Chemistry*, vol. 52, no. 1, pp. 59–64, 2006.

[12] M. P. Perrone and L. N. Cooper, “When networks disagree: ensemble methods for hybrid neural networks,” in *Neural Networks for Speech and Image Processing*, R. J. Mammone, Ed., pp. 126–142, Chapman-Hall, New York, NY, USA, 1993.

[13] A. Krogh and P. Sollich, “Statistical mechanics of ensemble learning,” *Physical Review E*, vol. 55, no. 1, pp. 811–825, 1997.

[14] A. Krogh and J. Vedelsby, “Neural network ensembles, cross validation, and active learning,” in *Advances in Neural Information Processing Systems*, G. Tesauro, D. Touretzky, and T. Leen, Eds., vol. 7, pp. 231–238, MIT Press, Cambridge, Mass, USA, 1995.

[15] S. Geman, E. Bienenstock, and R. Doursat, “Neural networks and the bias/variance dilemma,” *Neural Computation*, vol. 4, no. 1, pp. 1–58, 1992.

[16] E. B. Kong and T. G. Dietterich, “Error-correcting output coding corrects bias and variance,” in *Proceedings of the 12th International Conference on Machine Learning (ICML ’95)*, pp. 313–321, Tahoe City, Calif, USA, July 1995.

[17] R. Kohavi and D. Wolpert, “Bias plus variance decomposition for zero-one loss functions,” in *Proceedings of the 13th International Conference on Machine Learning (ICML ’96)*, L. Saitta, Ed., pp. 275–283, Morgan Kaufmann, Bari, Italy, July 1996.

[18] P. Domingos, “A unified bias-variance decomposition for zero-one and squared loss,” in *Proceedings of the 17th National Conference on Artificial Intelligence*, pp. 564–569, Austin, Tex, USA, July-August 2000.

[19] U. Naftaly, N. Intrator, and D. Horn, “Optimal ensemble averaging of neural networks,” *Network: Computation in Neural Systems*, vol. 8, no. 3, pp. 283–296, 1997.

[20] L. Breiman, “Bagging predictors,” *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[21] C. Schaffer, “Selecting a classification method by cross-validation,” in *Proceedings of the 4th International Workshop on Artificial Intelligence and Statistics*, pp. 15–25, Fort Lauderdale, Fla, USA, January 1993.

[22] J. R. Quinlan, “Comparing connectionist and symbolic learning methods,” in *Computational Learning Theory and Natural Learning Systems*, vol. 1, pp. 445–456, MIT Press, Cambridge, Mass, USA, 1994.

[23] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[24] D. H. Wolpert, “Stacked generalization,” *Neural Networks*, vol. 5, pp. 241–259, 1992.

[25] J. D. Wichard, C. Merkwirth, and M. Ogorzałek, “Detecting correlation in stockmarkets,” *Physica A*, vol. 344, no. 1-2, pp. 308–311, 2004.

[26] A. Rothfuss, T. Steger-Hartmann, N. Heinrich, and J. D. Wichard, “Computational prediction of the chromosome-damaging potential of chemicals,” *Chemical Research in Toxicology*, vol. 19, no. 10, pp. 1313–1319, 2006.

[27] D. W. Hosmer and S. Lemeshow, *Applied Logistic Regression*, John Wiley & Sons, New York, NY, USA, 1989.

[28] M. Riedmiller and H. Braun, “A direct adaptive method for faster backpropagation learning: the RPROP algorithm ,” in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 1, pp. 586–591, San Francisco, Calif, USA, March-April 1993.

[29] C. Igel and M. Hüsken, “Improving the Rprop learning algorithm,” in *Proceedings of the 2nd International ICSC Symposium on Neural Computation (NC ’02)*, H. Bothe and R. Rojas, Eds., pp. 115–121, Academic Press, Berlin, Germany, May 2000.

[30] C. Merkwirth, J. D. Wichard, and M. Ogorzałek, “Stochastic gradient descent training of ensembles of DT-CNN classifiers for digit recognition,” in *Proceedings of the 16th European Conference on Circuit Theory and Design (ECCTD ’03)*, vol. 2, pp. 337–341, Kraków, Poland, September 2003.

[31] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, Cambridge, UK, 2000.

[32] V. N. Vapnik and A. J. Tscherwonenkis, *Theorie der Zeichenerkennung*, Akademie, Berlin, 1979.

[33] C. C. Chang and C. J. Lin, “Libsvm—Alibrary for support vector machines,” 2001.

[34] R. E. Bellman, *Adaptive Control Processes*, Princeton University Press, Princeton, NJ, USA, 1961.

[35] C. Merkwirth, M. Ogorzałek, and J. D. Wichard, “Stochastic gradient descent training of ensembles of DT-CNN classifiers for digit recognition,” in *Proceedings of the 16th European Conference on Circuit Theory and Design (ECCTD ’03)*, vol. 2, pp. 337–341, Kraków, Poland, September 2003.

[36] R. A. Hilgers, “Distribution-free confidence bounds for ROC curves,” *Methods of Information in Medicine*, vol. 30, no. 2, pp. 96–101, 1991.

[37] P. Finne, R. Finne, A. Auvinen, et al., “Predicting the outcome of prostate biopsy in screen-positive men by a multilayer perceptron network,” *Urology*, vol. 56, no. 3, pp. 418–422, 2000.

[38] M. Remzi, T. Anagnostou, V. Ravery, et al., “An artificial neural network to predict the outcome of repeat prostate biopsies,” *Urology*, vol. 62, no. 3, pp. 456–460, 2003.

[39] A. R. Zlotta, M. Remzi, P. B. Snow, C. C. Schulman, M. Marberger, and B. Djavan, “An artificial neural network for prostate cancer staging when serum prostate specific antigen is 10 ng./ml. or less,” *Journal of Urology*, vol. 169, no. 5, pp. 1724–1728, 2003.