universität
wien

# MASTERARBEIT

Titel der Masterarbeit

"Tumor Classification
Based on Gene Expression Profiles"

Verfasserin
BSc Melanie Angela Görner

angestrebter akademischer Grad
Master of Science in Mathematics (MSc)

Amsterdam, April 2010

| | |
|---|---|
| Studienkennzahl lt. Studienblatt: | A 066 821 |
| Studienrichtung lt. Studienblatt: | Masterstudium Mathematik |
| Betreuer: | Ao. Univ. Prof. Dr. Andreas Futschik |

**Acknowledgments**

For his kind support and suggestions during this study, I would like to thank my supervisor Prof. Dr. Andreas Futschik of the department of statistics at the University of Vienna.

Furthermore, I very much appreciate that Agendia BV (Amsterdam, the Netherlands) provided me with the possibility to write my thesis during an internship and also with the necessary data for the analysis. I want to thank the whole bioinformatics and medical affairs team (Annuska Glas, Paul Roepman, Femke de Snoo, Leonie Delahaye, Sun Tian and Sonia Salamanca) for their help and support especially concerning medical and technical questions. Special thanks go to Dr. Paul Roepman, who supervised the internship project, for the great comments, discussions and reviewing of the analysis.

Finally, I would also like to thank my dear friend Christian Geist for content-related and moral support, especially during the end of the writing process.

# Contents

# Chapter 1

# Introduction

In the last years, cancer has become a more and more frequent disease with approximately 3.2 million incidences per year just in Europe [23]. However, there are huge differences in the characteristics of individual cancer tumors, which lead to a different assessment of their aggressiveness. Here, we focus on breast cancer tumors, one of the most common kinds of tumors [44], and the question of whether a breast cancer patient might benefit from adjuvant therapy such as chemotherapy. For clinical diagnosis, criteria such as tumor size, histological grade and age (of the patient) are used to decide whether adjuvant therapy will be applied [20, 27]. However, this method causes overtreatment of many patients that are not in need for this therapy [10, 59, 60].

Cancer cells forming a tumor are characterized by their uncontrolled growth compared to normal tissue. Since this is caused by gene abnormalities, accessing the genetic code of these cells can reveal valuable information about the special characteristics of an individual tumor. This is done by measuring the amount of messenger RNA that enables gene expression, i.e., the process of transforming DNA code into proteins, defining the actual behavior of the cell.

For many years, measuring the expression of genes was elaborate and time consuming, until the development of microarray technology in the 1990s, which allows for the efficient extraction of thousands of gene expressions at a time [30]. Therefore, it has been possible to record large data sets suitable for disease classification on the basis of statistical analysis and machine learning techniques.

One of the first commercially available tests for tumor outcome prediction that takes advantage of the microarray technology is the *MammaPrint* test that has been developed in 2002 by van't Veer et al [60]. It allows for the prediction of distant metastasis development of breast cancer tumors by classifying their gene expression data. The test was developed analyzing tumor tissue from 78 breast cancer patients younger than 55 years for which it was known whether they developed distant metastases within 5 years after diagnosis. The advantage of MammaPrint compared to clinical characteristics defined in the NIH [27] and St. Gallen [20] guidelines is, that it produces a much lower rate of type II

errors[1] while keeping the type I errors[2] comparably low ($< 10\%$). This is a huge improvement because using the test can therefore save a lot of people from unnecessary chemotherapy that is meant to reduce the risk of metastasis development [60].

Generally, since microarrays became available for the extraction of gene data there has been a vast amount of research studies concerning disease classification based on microarray gene expression data (see e.g., [39, 52]). Topics that are addressed in these studies mainly concern the selection of genes that are able to distinguish between different classes (feature selection) and the definition of appropriate classification algorithms as well as their validation.

Especially the problem of feature selection deserves particular attention: Although a lot of studies defining classifiers for the prediction of breast cancer outcome have been published, the overlap of the selected features within these studies is remarkably low, even though performances are comparable [21]. This might lead to the conclusion that there is not just one unique set of optimal features, but rather a lot of non-overlapping feature sets with approximately the same predication power that cover similar biological functionalities involved in disease progression. On the other hand, taking into account the low number of samples in theses studies ($\sim 100$) compared to the huge number of variables (genes) ($\sim 25{,}000$), feature selection might be strongly dependent on the given set of training samples [41].

For this study we have access to a total of about 1000 samples (see Appendix C) for the analysis of classification and feature selection methods. All samples are taken from breast cancer patients, who depending on the data (sub-)set are lymph node negative or have a maximum of three lymph nodes that are positive, i.e., metastatic cells are found in maximal three lymph nodes adjacent to the primary tumor. The classification task is to predict the development of distant metastases on the basis of gene expression profiles obtained by microarray measurements. A distant metastasis or metastatic disease is the spread of cancer cells from the primary tumor to distant lymph nodes or other organs of the body [42]. In contrast to that, the spread to regional lymph nodes is called lymph node involvement or regional disease.

As the basis of our analysis we take the MammaPrint algorithm, which classifies tumor samples based on a set of 70 genes. We will first concentrate on classification using these genes rather than considering the whole genome. For comparing the quality of classification results we will – similar to the developers of MammaPrint – focus on restricting the maximal number of type II errors (false prediction of remaining metastasis free) and only if this is provided regard the minimization of type I errors. This defines a special kind of classification problem which leads to a strong focus on nearest centroid classifiers (such as MammaPrint) that allow for an easy incorporation of this restriction in the classification assignment (see chapter 4). After determining the classifier that

[1] Falsely predicting distant metastasis development (for patients that remain free of distant metastasis for at least 5 years).

[2] Falsely predicting that no distant metastasis will develop within 5 years.

performed best on our test data (using the set of 70 genes as features), we will then turn to analyzing the influence of different feature selection methods on its performance (see chapter 5). Ideally, we would want to compare each of these feature selection methods in conjunction with each of the classifiers, but in the interest of efficiency we decided to compare feature selection methods using the previously best performing classifier only.

In chapter 2 we describe the methods of data extraction, data preprocessing and classification analysis used within this study. Next, we give an overview of commonly used classification and feature selection methods in the area of the analysis of microarray gene expression data (chapter 3). In this chapter we will also discuss why the nearest centroid method is particularly suitable for the special classification problem considered here. This motivates the detailed analysis and comparison of different nearest centroid classifiers in chapter 4. Thereafter, we will study the influence of various feature selection methods on the performance of nearest centroid classification (chapter 5). Finally, in chapter 6, we conclude the results obtained in chapters 4 and 5 and discuss possible future work.

# Chapter 2

# Methods

DNA microarray technology allows for a fast analysis of thousands of genes in parallel. Data collected by this technique is the basis for many classification methods developed in the area of predicting disease outcome [32, 39, 52]. Filtering the large amount of data and defining a meaningful classifier to obtain a reliable prediction of the outcome is the main goal of microarray gene expression classification.

In this study we will consider the problem setting of predicting distant tumor metastases development based on gene expression levels measured on microarrays. Before looking at the classification problem itself, we first explain how data can be extracted from microarrays (section 2.1) and what preprocessing steps are conducted (section 2.2). Then we turn to the definition of the specific classification task and error measures that are used throughout the study (section 2.3). In section 2.4 we finally introduce methods for the evaluation of classification performance that are necessary for the optimization and comparison of classifiers presented in chapters 4 and 5.

## 2.1 Measuring Gene Expressions Using Microarray Technology

DNA microarrays are a powerful tool to measure expressions of almost all human genes very fast and on very limited space [32, 39, 52]. A microarray consists of a large variety of spots each of them being less than $250\mu m$ in diameter and capable of binding a specific DNA molecule. There are various microarray platforms [30] which we do not explain in detail here, but concentrate on those that use *complementary-RNA (cRNA)* as input probes, such as the Hu25K, 44K, HD44K and LD8pack arrays (by Agilent Technologies [1]), that are used for obtaining the data considered in this study. Whereas the first microarray comprises the whole human genome, the others are user-specified arrays that are designed to measure a set of 231 predefined genes more accurately since these are measured multiple times on one array. The Hu25K microarray containing

25,000 60-polymer oligonucleotide probes has been used during the development of the MammaPrint algorithm in [60]. Afterwards, for the diagnostic use of MammaPrint, it has been replaced by the low-density 8pack (LD8pack) array, that measures less genes, i.e. contains only 1,900 probes, where each of the 231 genes of interest is measured three times [26]. Furthermore on just one array slide, there are eight identical sub-arrays allowing for eight simultaneous analyses. More recently the Agilent 44K and HD44K arrays have been developed which measure more than 41,000 probes, where for the HD44K array four identical 44K sub-arrays are present on one slide. In these (sub-)arrays each of the 231 genes is measured five times.

For this study we had access to multiple datasets, each of them being measured on one of the aforementioned microarrays. An overview of all datasets and the corresponding microarrays used can be found in Appendix C.

In the human body, DNA is transcribed to the so called *messenger-RNA (mRNA)*, which then translates into amino acid sequences forming a particular protein. This process is called *gene expression*. Measuring gene expressions gives valuable insights into the genetic code of cells and in particular cancer cells and can therefore be used to reveal information about cancer behavior such as developing metastases. Microarray technology is capable of measuring thousands of gene expressions simultaneously which is why it gained so much attention in the area of cancer classification.

For microarray analysis mRNA is isolated from a tissue sample containing cancer cells. The extracted mRNA is amplified (replicated multiple times) and labeled with fluorescent markers (*dyes*) to obtain labeled cRNA.[1] Because of the nature of DNA, which usually is connected to its complementary part in a double helix, the labeled cRNA can be bound to small parts of DNA which are attached to the microarray. This process is called *hybridization*. For comparison, a second batch of cRNA is built from mRNA of a so called *reference pool*, which in our case consists of an equal amount of RNA from various breast cancer tumors. The two batches are labeled differently using different dyes, red-fluorescent Cy5 and green-fluorescent Cy3, and thereafter hybridized on the same microarray. A scanner is used to measure the amount of fluorescent dyes on each spot via imaging the microarray slide. It locates the spots and calculates the fluorescence intensities (Int(Cy3) and Int(Cy5)) for all pixels within each spot and outside of the spot (in the so called *background*). The gene expression level of each gene $\text{expr}(g)$ is given by the $\log_{10}$ ratio[2] of the fluorescent dye intensities:

$$\text{expr}(g) = \log_{10}\left(\frac{\text{Int(Cy5)}}{\text{Int(Cy3)}}\right) \tag{2.1}$$

which is calculated after preprocessing via subtracting background noise and normalization of the intensities with respect to so called *housekeeping genes* or *normalization genes*.

[1] See [59, 60] for more information on RNA isolation and labeling.

[2] Other common choices are $\log_2$ or the natural logarithm ln.

## 2.2 Data Preprocessing

The measurement of gene expressions with microarrays underlies a variety of sources for errors and variability, e.g., due to changing experimental conditions. Therefore, various preprocessing steps have to be implemented before a final gene expression measure is obtained.

Given the scanned image of the fluorescent intensities, feature extraction tools such as the one provided by Agilent Technologies [2], first search for the actual spots where the probes are supposed to bind to. Then the intensities at each spot as well as in the surrounding background are calculated and subtracted from the spot intensity. Such background noise can for example be caused by non-specific binding, i.e., adhering to sites on the surrounding (glas) area of the spots [39]. Usually background noise is supposed to be additive and independent of the true concentration of the gene in the sample given. Probes with negative background subtracted intensities are excluded from further calculations [26].

Since the fluorescent markers Cy3 and Cy5 have different fluorescent intensities due to different dye properties and light sensitivities, it is also necessary to correct for noise and potential bias introduced by the different dye measurements separately. There are multiple ways of addressing this problem (see e.g. [39]); in our case for measurements on LD8pack a set of 465 normalization genes are used that were found to have stable intensity levels in this specific test setting of tumor tissue analysis. For the other array types, all genes are used for normalization. Based on these genes a normalization factor is calculated which uses a combination of linear and lowess[3] correction (see Agilent Feature selection software manual [2]). The product of the background subtracted signal intensity and this normalization factor is then used to determine the log ratio of the gene expression level in (2.1).

Each gene is measured in quintuple (for 44K arrays) or triplicate (for the LD8pack array) on the single arrays and furthermore in some cases (e.g. for all measurements on the Hu25K and the LD8pack array) a second *dye-swap* hybridization is performed where the sample and the reference are labeled with the reverse dye compared to the first hybridization. To combine replicate measurements (first those within a single microarray and second those obtained form the two hybridizations) of the Hu25K array the *xdev* approach [13, 63] has been applied:

$$\text{xdev} = \frac{I_2 - I_1}{\sqrt{\sigma_2^2 + \sigma_1^2}}, \tag{2.2}$$

where $I_2$ and $I_1$ are the intensity of the two dyes and $\sigma_2^2$ and $\sigma_1^2$ are the corresponding variances of the estimated measurement errors.

However, this method of combining intensity measurements showed undesirable artifacts [26] and has therefore been replaced by error-weighted averaging

[3]Locally weighted scatterplot smoothing (lowess) is a regression technique for fitting smooth curves to a data set via local fitting of polynomials [11].

[63] for the more recent measurements performed on 44K and LD8pack arrays:

$$\bar{x} = \frac{\sum_{i=1}^{N} w(i) \cdot x(i)}{\sum_{i=1}^{N} w(i)}, \tag{2.3}$$

where the $x(i)$s are the log ratio intensities of the $i = 1, \ldots, N$ measurements ($N \in \{3, 5\}$ for combining measurements within the array and $N = 2$ for combining two hybridizations). The scalar weights $w(i)$ are inversely proportional to the approximated log ratio errors:

$$\sigma_{x(i)} \approx \log_{10}(e) \cdot \sqrt{\frac{\sigma_1^2}{I_1^2} + \frac{\sigma_2^2}{I_2^2}}, \quad w(i) = \frac{1}{\sigma_{x(i)}^2}.$$

## 2.3 Classification for the Prediction of Distant Metastasis Development

We are now going to present a detailed definition of the classification problem considered in this study. First, we define binary classification of tumor outcome and misclassification errors that can occur in this context. Thereafter, we focus on the special case of predicting distant metastasis development and the severity of different kinds of misclassification errors, which will lead to a particular performance assessment of classifiers defined in chapters 4 and 5.

Consider classification of tumor outcome where we distinguish only two classes of possible outcome (binary classification):

**positive:** The disease is observed / progresses (the tumor develops metastases).

**negative:** The disease is not observed / does not progress (the tumor does not develop metastases).

A tumor is represented by the expression of its genes measured on a microarray. For classification we will, however, not consider all genes, but only a relatively small subset, whose elements are called *features*. A classifier is a mapping defined on the *feature space* $\mathcal{X} \subset \mathbb{R}^n$ assigning to each feature vector in $\mathcal{X}$, i.e., the vector of gene expressions of the features, a *class*, an element of the space {positive, negative}.

The class assigned to the feature vector does not necessarily correspond to the tumor's true class which leads to four possible results of the classification with respect to the true classes:

**Definition 1** (diagnostic table)

| | | *TRUE class* | |
|---|---|---|---|
| | | *positive* | *negative* |
| *TEST outcome* | *positive* | *true positive (TP)* | *false positive (FP)* |
| | *negative* | *false negative (FN)* | *true negative (TN)* |

Thus, a binary classifier can produce two kinds of errors: FP (type I error) and FN (type II error). For measuring a classifier's accuracy, we will use the quantities *sensitivity* and *specificity*:

**Definition 2** (sensitivity)
*The* false negative rate (FNR) *is the relative amount of* false negatives *with respect to all positive samples:*

$$FNR = \frac{FN}{TP + FN}.$$

*Analogously, the* true positive rate (TPR) *or* sensitivity *is the relative amount of* true positives *over all positive samples:*

$$sensitivity = \frac{TP}{TP + FN} = 1 - FNR.$$

Thus, sensitivity measures a classifiers ability of recognizing positive samples (sensitivity = 1 means that all positive samples are correctly classified).

**Definition 3** (specificity)
*The* false positive rate (FPR) *is the relative amount of* false positives *with respect to all negative samples:*

$$FPR = \frac{FP}{FP + TN}.$$

*Analogously, the* true negative rate (FPR) *or* specificity *is the relative amount of* true negatives *over all negative samples:*

$$specificity = \frac{TN}{FP + TN} = 1 - FPR.$$

Thus, specificity measures a classifiers ability of recognizing negative samples (specificity = 1 means that all negative samples are correctly classified).

Please note that sensitivity = specificity = 1 represents perfect classification, since in this case all samples (positive and negative ones) are correctly classified. Therefore, we are aiming at maximizing these measures, which is similar to minimizing the (overall) misclassification error.

**Definition 4** (misclassification error)
*The* misclassification error (rate) *e is the relative amount of misclassified samples with respect to all tested samples:*

$$e = \frac{FP + FN}{TP + FP + FN + TN} = \frac{FP + FN}{n},$$

*where n is the number of samples in the test set.*

When using this measure as an estimation of the true error rate of a classifier, both error types are considered equally important, whereas separately measuring both kinds of errors by sensitivity and specificity allows for a more flexible error access.

Consider now the special case of predicting whether a tumor will develop distant metastases. The spread of tumor cells via the bloodstream or lymphatic system to lymph nodes distant from the primary tumor or other organs is called metastatic disease or distant metastasis [42]. A tumor formed by those cells is called metastatic tumor, which in the case of breast cancer is usually found in bones, lungs, liver or brain. The distinction of metastases in regional or distant lymph nodes is less obvious than for the spread to other organs. For example, (upper) arm lymph nodes are still considered regional lymph nodes to a primary breast cancer tumor. In order to define the (most difficult) upper line for lymph node distances, we call supraclavicular lymph nodes distant and those below the clavicula regional. By the term 'metastasis' we will in the following always refer to distant metastasis, although it might not be explicitly stated.

In order to formulate the classification problem as a binary problem, we define metastases development within 5 years as the positive and remaining metastases free for at least 5 years as the negative outcome. Equivalently, we refer to these two groups as high risk or low risk patients respectively, since a positive outcome indicates a high risk of developing distant metastases, whereas a negative outcome indicates a low risk thereof.

In this context, we consider type II errors worse than type I errors, i.e., misclassification of high risk patients is less accepted than misclassification of low risk patients. This assumption arises from the clinical background we are dealing with: overtreatment of low risk patients is stressful and costly, but undertreatment of high risk patients might lead to considerable worse health or even death. One possibility of addressing this problem is via assigning asymmetric costs to the two kinds of errors [3, 36, 37, 61] and minimizing the total costs caused by misclassifications. Another solution is to restrict the type II error rate (FNR) to some predefined maximum during training. We will use the latter method since we would like to control sensitivity, i.e., the frequency of type II errors. This is not necessarily achieved by cost assignment since it might be difficult to define costs appropriately, i.e., such that we can make sure to keep the type II error low.

## 2.4 Assessing Classifier Performance

In order to address the aforementioned problem, a classifier is *trained* on a set of tumor samples with known true classes, a *training set*, optimizing specificity and, more importantly, sensitivity on this set. The classifier is than *validated*, i.e., it's performance is measured on a second data set,which also consists of samples and their corresponding (true) classes, but is not used for training (*validation set*). This is done to avoid *overfitting*, a typical problem in supervised learning

(classification using training data sets for the classifier's definition) [43, 50]. Overfitting means that the classifier is over adjusted to the training data, i.e., its parameters are optimized in such a way that it predicts the classes of the training samples very well, but performs poorly on an independent data set. This happens when the classifier does not capture the underlying relationship of the samples to their classes, but rather random errors which occur, e.g., during measurements (noise). Since microarray gene data usually consists of only very few samples compared to the number of features (genes), one should put special emphasize on avoiding overfitting [58]. One way of addressing this problem is to use feature selection (see section 3.2), i.e., to reduce the feature space $\mathcal{X}$ to only those genes that are most suitable for classification. Furthermore, the evaluation of the classifier's performance should not be done on the same set the classifier has been trained on. This can either be achieved by validation on an independent data set or cross-validation.

### 2.4.1 Cross-Validation

Cross-validation (CV) [6, 33, 52] is a procedure frequently used for classification problems especially if there is only little data with known true classes available. In this case one might want to use all samples for training and cannot exclude some for subsequent validation. In order to still prevent overfitting, one possibility is to repeatedly train the classifier on a (randomly chosen) subset of the samples and use the remaining samples for validation. Thereby one can use all samples for training and (cross-)validation at the same time.

The most common CV-method is $k$-fold-cross-validation which randomly divides a given data set $S$ into $k$ subsets $S_1, \ldots, S_k$ of approximately the same size. In each CV-step $i = 1, \ldots, k$ the training set $S \setminus S_i$ is used to define a classifier which is subsequently tested on the validation set $S_i$. CV performance is defined as the average performance over all CV-steps, i.e., the average over the individual error measurements on the sets $S_1, \ldots, S_k$ of left out samples.

The special case of a $k$-fold-cross-validation, where $k = |S|$, the number of samples in the set, is called *leave-one-out cross-validation* (LOOCV).

*Stratified cross-validation* refers to cross-validation where in each CV-step the training and validation set contain approximately the same relative frequency of samples of each class. This method brings the advantage of generally being less biased than non-stratified cross-validation [33].

### 2.4.2 ROC Curve Analysis

*Receiver operating characteristics (ROC) curves* are a very useful tool to visualize and analyze the performance of binary classifiers [22, 35]. As discussed before, a classifier's performance is given by the measures sensitivity and specificity (each of which has a range of $[0, 1]$). The *ROC space* is defined as $[0, 1] \times [0, 1]$, where a *ROC point* in this space represents a classifier's false positive rate ($\text{FPR} = 1 - \text{specificity}$) on the first dimension and its true positive

rate (TPR = sensitivity) on the second. The point $(0, 1)$ in this space corresponds to perfect classification, i.e. all samples are classified correctly as there are no misclassified negative samples (specificity = 1) and all positive samples are correctly classified (sensitivity = 1). The diagonal line from $(0, 0)$ to $(1, 1)$, on the contrary, represents classification by random guessing: Consider classification by coin tossing where samples are classified into the positive class with probability $p$ and into the negative class with probability $1 - p$, $p \in [0, 1]$. The expected performance of this classifier results in a ROC point of $(p, p)$, since on average $p$ of the negative samples are incorrectly classified and $1 - p$ of the positive samples. In particular, a classifier classifying all samples as positive corresponds to the ROC point $(1, 1)$. Informally speaking, we can say that a classifier gets worse the closer it gets to the diagonal and better the more it approaches $(0, 1)$. ROC points below the diagonal do practically not occur, because for a classification worse than chance we could then just invert the class assignment.

If we now consider a classifier which does not assign a class to each sample but rather a score representing the degree to which it belongs to the positive class, we can calculate several ROC points for this classifier: Consider a (finite) training set of samples $x_1, \ldots, x_n$ (with known true classes) and its corresponding scores $s_1, \ldots, s_n$. Without loss of generality, we can assume that the scores are in ascending order, i.e., $s_1 < s_2 < \cdots < s_n$. Consider now a cutoff value $c$, such that a sample is classified into the positive class if and only if its score is greater than $c$. In theory for each possible value of $c \in \mathbb{R}$ a classification result could be obtained which is then compared to the true outcome to calculate sensitivity and specificity values that define a ROC point. For practical reasons, only those values are considered as cutoff values which do occur as a score of an element of the training set. For $c = s_i$ this implies that the samples corresponding to $s_{i+1}, \ldots, s_n$ are classified positive and those corresponding to $s_1, \ldots, s_i$ negative. Hence, $n + 1$ classifications can be obtained by considering the following cutoff values: $s_1 - \varepsilon, s_1, s_2, \ldots, s_n$ for some $\varepsilon > 0$. The smallest and the largest values result in the ROC points $(0, 0)$ and $(1, 1)$ respectively. Since we consider only finite training sets we obtain a 'wrinkled' graph as the classifier's ROC curve (figure 2.1) on the training set.

Besides the obvious visual analysis and comparison of classifiers offered by this technique, a common measure to assess classifier performance via ROC-curve analysis is to calculate the *area under the curve (AUC)*. Since this measure is a portion of the unit square, it's value lies between 0 and 1. However, as classification below the diagonal (representing random classification) does practically not occur, the AUC value can be expected to range between 0.5 (area under the diagonal) and 1. From a statistical point of view, the AUC is equivalent to the probability that the classifier ranks a randomly chosen positive sample higher than a randomly chosen negative sample. This is equivalent to the Wilcoxon test of ranks [22].

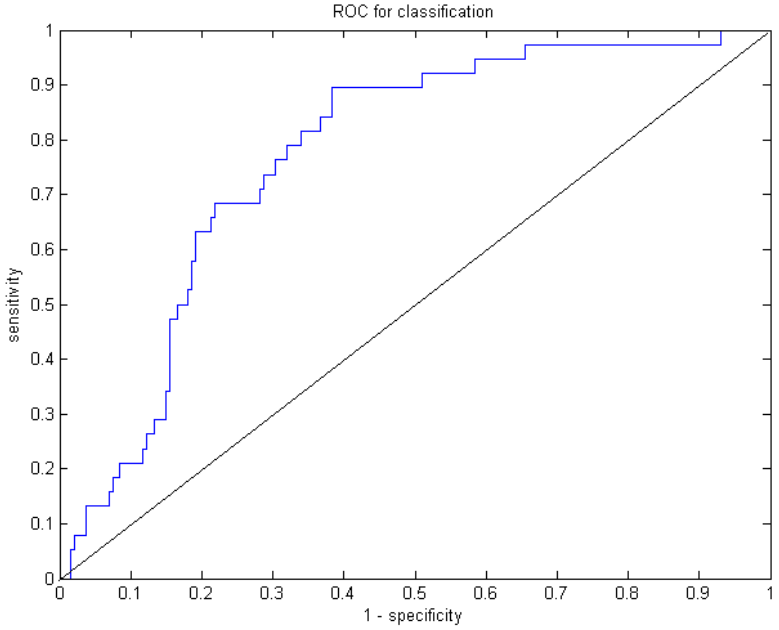



Figure 2.1: Example ROC curve. The diagonal line represents the expected performance of random classification, the blue graph above the diagonal a classifier's performance depending on different cutoff values.

# Chapter 3

# Classification and Feature Selection on Microarray Gene Expression Data

## 3.1 Overview of Classifiers

Many different classification methods have been developed and extensively tested and studied in the area of microarray gene expression data analysis [39, 52]. Since this specific type of data usually consists of only relatively few samples and a large amount of features, choosing an appropriate classification method is important to obtain reliable predictions on the tumor outcome.

The term 'classification' is equivalent to the term 'supervised learning' [52], where a data set with known true classes (learning set) is used as reference for the assignment of class labels to the objects of interest (samples). The task is to understand the underlying relation between the samples and the true class outcome. In unsupervised learning, no predefined reference labels are used which therefore have to be discovered from the data by the learning method. This technique is especially useful if the possible outcome classes are not know a priori. In the following we focus on supervised learning (classification) describing some of the most common methods and discuss their properties and especially their suitability for the classification problem defined in section 2.3:
Given the binary classification problem of predicting tumor outcome (positive or negative), we are aiming at minimizing misclassifications measured by the classifier's sensitivity and specificity values. In particular, we control the sensitivity, in that we set an upper bound for the maximum false positive rate (Definition 2).

Throughout this section we will use the following notation: For $i = 1, 2, \ldots, m$ genes and $j = 1, 2, \ldots, n$ samples, we denote the expression of gene $i$ in sample $j$ by $x_{ij}$, where for each $j$ the vector $x_j = (x_{1j}, \ldots, x_{nj})^T$ is an element of the

feature space $\mathcal{X}$. For $n = 1$, we will also simply refer to the feature vector $x_1$ by $x$. The outcome class vector will be denoted by $y = (y_1, \ldots, y_n)$, where each element $y_j$ represents a class, an element of $\{1, \ldots, N\}$. In the binary case, the classes will also be denoted by $+1$ representing the positive class and $-1$ representing the negative class. The matrix of gene expressions per feature $i$ and sample $j$ is given by $X = (x_{ij})_{i,j}$. Although here we use a matrix representation, we will also denote the set of samples $\{x_j \mid j = 1, \ldots, n\}$ by $X$ which will sometimes be more convenient than the matrix representation. The pair of the set $X$ and the corresponding outcome class vector $y$ is called the *learning set* $\mathcal{L} = (X, y)$.

A classifier is defined by a mapping $\mathcal{C}$ assigning to each feature vector $x \in X$ a class $k = 1, \ldots, N$ (or in the binary case $k \in \{-1, +1\}$ instead of $k \in \{1, 2\}$). The performance of a classifier (given by sensitivity and specificity) can be measured on (subsets of) the learning set $\mathcal{L}$, since it provides the true outcome class for each sample $x$ for comparison to the classification outcome $\mathcal{C}(x)$.

### 3.1.1 Naive Bayes

The basis of the naive Bayes classification method, is the *Bayes' Theorem*: For each class $k \in \{1, \ldots, N\}$ and each feature vector $x \in \mathcal{X}$ the conditional probability $\mathbb{P}(k|x)$ of a sample with feature vector $x$ belonging to class $k$ is given by:

$$\mathbb{P}(k|x) = \frac{\mathbb{P}(k) \cdot \mathbb{P}(x|k)}{\mathbb{P}(x)},$$

where $\mathbb{P}(k)$ is the probability of observing class $k$ which is approximated by the so called class prior $p(k)$, e.g., the relative size of class $k$ in the population of interest or in the training data. $\mathbb{P}(x|k)$ is the probability of observing the feature vector $x$ within the elements of class $k$ approximated by the class conditional density $p(x|k)$ and $\mathbb{P}(x)$ is the probability for observing $x$ approximated by $\sum_{l=1}^{N} p(l) \cdot p(x|l)$. In the case of naive Bayes classification, the features are assumed to be independent and therefore the class conditional density $p(x|k)$ is equal to the product of the conditional densities for each feature: $p(x|k) = \prod_{i=1}^{m} p(x_i|k)$, where $x_i$ is the gene expression level of gene $i$ in the feature vector $x$.

The *Bayes classification rule* is then given by assigning to $x$ the class which maximizes the so called posterior probability $p(k|x)$, an approximation of $\mathbb{P}(k|x)$:

$$\mathcal{C}_B(x) = \text{argmax}_k \; p(k|x) = \text{argmax}_k \; \frac{p(k) \cdot p(x|k)}{\sum_{l=1}^{N} p(l) \cdot p(x|l)}.$$

Many classifiers can be view as variations of this simple classification rule with respect to the estimation of the class conditional probabilities $\mathbb{P}(x|k)$ and the assumption on independence of the features. Examples include discriminant analysis, logistic regression, neural networks and classification trees [52].

Although the naive Bayes classifier is considered a rather simple classifier, it nevertheless showed good empirical results in previous studies [16, 18, 52]. However, we did not employ this classifier on our classification problem because we could not easily incorporate the requirement of restricting the amount of falsely positive classified samples. One possibility to do this would be to use cost asymmetry, but as discussed earlier, we restricted ourselves to the approach of directly controlling the type II error instead of considering cost assignment. Furthermore, assumptions on the prior probabilities and the underlying distributions (depending on the true class labels) for obtaining the data variable $X$ have to be made. If those assumptions are not justified, estimating the probability measures yields a huge source of errors.

### 3.1.2 Nearest Centroid

For nearest centroid classification (NC) one uses a representative of each class, a so called centroid $c_k \in \mathcal{X},\ k+1,\ldots,N$, and classifies according to the distance of a feature vector $x$ to each of the centroids. A classic nearest centroid classifier searches for the nearest (defined by a given distance measure) centroid $\hat{c}$ to the feature vector $x$ and classifies it to the same class as $\hat{c}$. The choice of an appropriate distance measure should be considered depending on the classification problem given. Possible choices are euclidean distance, city block distance ($l_1$ - norm) or $1 -$ correlation. Centroids can for example be constructed by averaging over feature vectors of samples belonging to one class, but they do not necessarily have to be derived from real data.

The nearest centroid classifier is also a rather simple classifier which is for instance used in [5, 59, 60] for the classification of tumor samples. We are going to present an analysis of this classifier for the prediction of tumor outome in chapter 4, where we modify the classification rule in order to control the type II error and constructed centroids by averaging over samples belonging to either group.

### 3.1.3 $k$-Nearest Neighbor

The principle of $k$-nearest neighbor classifiers ($k$NN) is similar to the one of nearest centroid classifiers. Given a feature vector $x$, instead of searching for the nearest centroid, $k$NN determines those $k$ samples of a set $\mathcal{L}_1 \subseteq \mathcal{L}$ (with known true outcome) that are closest to $x$ with respect to some similarity measure (the $k$ nearest neighbors). In the classical context $k$ is usually odd and $x$ is assigned to the class where the majority of these neighbors belongs to (majority voting). Hence, $\mathcal{L}_1$ must contain at least one sample of each class.

Considering the fact that in some studies $k = 1$ neighbors have been determined optimal for classification [16, 18], the $k$-nearest neighbor classifier is well comparable to the nearest centroid classifier: for $|\mathcal{L}_1| = N$, where $\mathcal{L}_1$ consists of

exactly one element of each class, the $k$NN classification rule is even identical to NC with the elements of $\mathcal{L}_1$ defining the centroids.

In order to obtain a more type II error sensitive classifier (for our binary classification problem of predicting tumor outcome), we modified the $k$NN classification rule from majority voting to requiring that for the sample to be classified negative, $l$ of the $k$ neighbors have to belong to the negative class. Both numbers, $l$ and $k$, are optimized during a training procedure. This resulted in a choice of $k$ around $20-25$ (for $l$ being greater than $0.9 \cdot k$) during various trainings (results not shown). For such large numbers of $k$ compared to standard values of $k < 7$ used in the literature [16, 52], considerably higher computational costs compared to the nearest centroid classifier have to be taken into account. Furthermore, since $k$ is very large, the classification can be considered more similar to the nearest centroid rule (with centroids taken as average over multiple samples) than to classical $k$-nearest neighbor rule with small values for $k$. Hence, we decided not to report results on this classifier in this study as it does not give any further insights (compared to NC).

### 3.1.4 Support Vector Machines

Support vector machines (SVMs), first introduced by Vapnik in 1979 (see [9, 52] for detailed information), were designed for binary classification problems with a positive and a negative class, i.e., $y_j \in \{-1, +1\},\ j = 1, \ldots, n$. If the data is linearly separable, then there is a hyperplane $H = \{x \in \mathbb{R}^n \mid w \cdot x + b = 0\}$ separating the samples such that for all feature vectors $x_j,\ j = 1, \ldots, n$

$$w \cdot x_j + b \geq 0 \ \Leftrightarrow \ y_j = +1. \tag{3.1}$$

$w$ is the normal to the hyperplane $H$ and $\frac{|b|}{||w||}$ is the perpendicular distance from $H$ to the origin. The aim of support vector classification is to maximize the *margin* $(d_+ + d_-)$, the sum of the distances of the hyperplane to the closest positive $(d_+)$ and negative $(d_-)$ sample. By reformulating (3.1), we can define two parallel hyperplanes $H_1$ and $H_2$ such that for all $j = 1, \ldots, n$

$$\begin{array}{ll} H_1: & w \cdot x_j + b \geq +1, \quad \text{if } y_j = +1 \\ H_2: & w \cdot x_j + b \leq -1, \quad \text{if } y_j = -1 \end{array} \tag{3.2}$$

or equivalently

$$y_j(w \cdot x_j + b) - 1 \geq 0. \tag{3.3}$$

Since for $H_1$ and $H_2$ the distance to the origin is given by $\frac{|1-b|}{||w||}$ and $\frac{|-1-b|}{||w||}$, respectively, the margin, i.e., the distance between the hyperplanes $H_1$ and $H_2$ is $\frac{2}{||w||}$. Thus, one seeks to minimize the norm of $w$ subject to the constraint (3.3). Those sample points that lie on one of the two hyperplanes, i.e., for which equality holds in (3.3) are called *support vectors*, because their removal would change the solution of the minimization problem. By using the method of Lagrangian multipliers it can be reformulated to the dual problem:

$$\begin{array}{c} \max_\alpha \sum_j \alpha_j - \frac{1}{2} \sum_{l,j} \alpha_l \alpha_j y_l y_j x_l \cdot x_j \\ \text{s.t.} \ \ \alpha_j \geq 0, \ \sum_j \alpha_j y_j = 0, \quad j = 1, \ldots, n, \end{array} \tag{3.4}$$

where $x_l \cdot x_j$ denotes the dot product of the two feature vectors $x_l$ and $x_j$. The solution of (3.4) is given by $w = \sum_j \alpha_j y_j x_j$, where only those samples $j$ that satisfy $\alpha_j > 0$ are support vectors.

In the non-separable case, one could either introduce slack variables into the optimization problem or use *kernel functions* which implicitly map the sample data into a (usually higher dimensional) euclidean space $\mathcal{H}$: Given $\Phi : \mathbb{R}^n \to \mathcal{H}$, the kernel function $K$ is defined by

$$K(x_l, x_j) = \Phi(x_l) \cdot \Phi(x_j). \tag{3.5}$$

Here we need not explicitly know the mapping $\Phi$ to the possible infinite dimensional space $\mathcal{H}$. Commonly used kernels are:

| | |
|---|---|
| Polynomial of degree p | $K(x_l, x_j) = (x_l \cdot x_j + 1)^p$ |
| Gaussian radial basis function (RBF) | $K(x_l, x_j) = e^{-\|\|x_l - x_j\|\|^2/2\sigma^2}$ |
| Two-layer sigmoid neural network | $K(x_l, x_j) = \arctan(\kappa x_l \cdot x_j - \delta)$, |

for which it has been shown that there exists a mapping $\Phi$ fulfilling (3.5).

Support Vector Machines are more sophisticated classifiers arising from the area of machine learning, where the approach can be considered a black-box algorithm. This is due to the use of Kernel functions which implicitly define a mapping of the sample data into a (higher dimensional) euclidean space. By varying the kernel function or the parameters to the kernel functions, the SVM classifier often proved to be well adjustable to specific classification tasks [5, 61]. However, for the data sets that we used, samples seemed to be hardly separable even in the infinite dimensional spaces used by RBF kernels with varying parameter $\sigma$. Almost all samples were chosen to be support vectors; in particular all positive samples which form a considerable smaller class than the negative samples (data not shown). Even introducing soft margins (see [52, 61]), i.e., slack variables, that allow for misclassification errors in the area close to the separating hyperplane, could not solve this problem.

### 3.1.5 Classification Trees

Tree structured classifiers are constructed by repeatedly splitting the learning set $\mathcal{L} = \{(x_j, y_j) \mid j = 1, \ldots, n\}$ into subsets, which define the tree nodes, and assigning a class to each terminal node. Classification trees differ by the splitting rules they use, the stop-splitting criterion and the assignment of terminal nodes to the outcome class. We will focus here on the *CART* (classification and regression tree) classifier which is commonly used for microarray data classification [52]. The splitting rule is defined by maximizing a so called *impurity function*: For any N-tuple of relative class frequencies $p = (p_1, p_2, \ldots, p_N)$[1] with $p_k \geq 0$ $k = 1, \ldots, N$ and $\sum_{k=1}^{N} p_k = 1$ the impurity function $f$ fulfills:

[1] Please keep in mind, that here again (as for the general description of naive bayes classifiers) $p$ denotes an approximation to the underlying probability measure $\mathbb{P}$.

1. $f$ is maximal if all classes are equally frequent: $p = (\frac{1}{N}, \ldots, \frac{1}{N})$.
2. $f$ is minimal if there is just one class present in the learning set: $\exists k \in \{0, \ldots, N\} \;\; p_k = 1$.
3. $f$ is symmetric in $p$: $f(p) = f(p')$ if $p$ and $p'$ differ only by the ordering of their elements.

CART uses the gini index, defined by $g(p) = \sum_{k \neq l} p_k p_l = 1 - \sum_k p_k^2$, as impurity function. Another common choice of an impurity function in tree classification is e.g. the information gain criterion (section 3.1.6). Based on this function, we can define an impurity measure $im$ for each tree node (represented by $T \subseteq \mathcal{L}$): $im(T) = f(p_1(T), \ldots, p_N(T))$, where $p_k(T)$ is the estimated conditional probability (relative frequency) of observing class $k$ in node $T$. Formally, $p_k(T) = \pi_k \frac{|\{(x,y) \in T \mid y=k\}|}{|T|}$, where $\pi_k$ is the class prior probability. Each non terminal node $T$ is subdivided by a splitting rule $s$ into $T_R$ and $T_L$ consisting of the samples that are assigned to the right and left daughter node respectively. The goodness of the split is measured by the *decrease in impurity*:

$$\Delta im(s, T) = im(T) - \frac{|T_R|}{|T|} \cdot im(T_R) - \frac{|T_L|}{|T|} \cdot im(T_L).$$

For each node $T$ the splitting rule with biggest decrease in impurity is chosen to determine the daughter nodes $T_R$ and $T_L$. This procedures is repeated until the full classification tree is grown (i.e. some stopping criterion is met). The terminal nodes of this tree are assigned to a class, e.g., by majority voting of the true class labels of the learning set.

Decision trees such as CART can be used as standalone classification methods, but their main benefit for the classification of gene expression data is in using them as a base classifier for aggregation algorithms such as boosting [7, 16]. In that context especially so called decision stumps, i.e., trees that only perform one split, are very popular since they have much less computational complexity and have even been shown to outperform fully grown trees as the basis a boosting algorithm [8].

### 3.1.6 Boosting

Boosting belongs to the class of aggregation predictors[2], i.e., it aims at combining a number of classification hypothesis $h_1, h_2, \ldots, h_T$ from a simple classifier (*weak learner*). Thereby a the combined hypothesis

$$f(x) = \sum_{i=t}^{T} \alpha_t h_t(x)$$

is obtained, where the weights $\alpha_i$ are assigned to each hypothesis during the learning process [7, 40]. In each boosting iteration $t = 1, \ldots, T$ the training

[2] See [52] for an overview of aggregation predictors

data is reweighted, which is been done by defining a new sample distribution $D_t$ depending on $D_{t-1}$ and the accuracy of $h_t$. The performance of a rather simple and therefore probably less accurate classifier can be improved by means of combining the weak hypotheses obtained by the weak learner based on the distributions $D_t,\ t = 1, \ldots, T$. The most common boosting algorithm, first introduced by Freund and Schapire [24], is *AdaBoost* (see [48] for a brief introduction).

The algorithm is designed for binary classification problems where the learning set $\mathcal{L}$ is given by pairs $(x_1, y_1), \ldots, (x_n, y_n)$, $x_j$ being an element of the feature space $\mathcal{X}$ for the $j$th sample and $y_j \in \{-1, +1\}$ the corresponding class. Starting with an initially uniform distribution $D_1(j) = \frac{1}{n},\ j = 1, \ldots, n$ the following steps are preformed for each iteration $t = 1, \ldots, T$:

**Algorithm 1** The AdaBoost algorithm

1. Training of the weak learner using the distribution $D_t$ and obtaining the weak hypothesis $h_t$
2. Calculation of the training error: $\epsilon_t = \mathbb{P}_{D_t}(h_t \neq y) = \sum_{\{j:h_t(x_j)\neq y_j\}} D_t(j)$
3. Determination of the weight $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$
4. Update of the distribution $D_t$ by $D_{t+1}(j) = \frac{D_t(j) exp(-\alpha_t y_j h_t(x_j))}{\sum_l D_t(l) exp(-\alpha_t y_l h_t(x_l))}$, $j = 1 \ldots, n$

Common choices for the weak classifiers are classifications trees or decision stumps (one dimensional trees).

The concept of the AdaBoost classifier has already been introduced in 1995 (by Freund and Schapire [24]) and since then constantly been improved and modified (see e.g. [49]). We will, however, consider the basic version defined here, which suffices for our purposes since we will use this classifier for feature selection only (see section 3.2.3). It can be shown that the training error of the AdaBoost algorithm (and also that of other boosting algorithms) converges to zero if the weighted empirical error of the weak learner can be guaranteed to be smaller than $\frac{1}{2} - \gamma,\ \ \gamma > 0$ [40]. Furthermore, there are studies that empirically show slow over-fitting behavior, although this cannot be proved theoretically [7, 38].

### 3.1.7 Conclusion

The classifiers presented here are commonly used and compared against each other within the area of classifying microarray gene expression data. It has been shown that each of them is – in general – useful to produce a meaningful classification of this type of data, although slight differences in performances can

be observed depending on the (learning) data set used for classification [5, 15, 16, 30, 52]. However, not all of them are suitable for the particular classification problem studied here, where we want to control the classifier's sensitivity (and therefore type II errors) directly. This is the case for Bayes classifiers as well as for $k$NN classifiers, that require an unusually large number of neighbors in order to meet the requirement of controlling sensitivity. Another problem is that the samples of the different classes in the learning data we used seem to be strongly overlapping and thus the SVM approach is not applicable since the problem could not be solved by using various kernels that (implicitely) map the data into a higher dimensional space. Furthermore, Hand [29] pointed out that for example in the area of bioinformatics, sophisticated classification methods often provide only little improvement compared to more simple approaches. Therefore, we decided to consider only one kind of classifier: the nearest centroid classifier and focus on the comparison of feature selection methods instead. In chapter 4 various modifications of the NC classification method are presented, based on that MammaPrint algorithm developed in [60]. Their performances on a learning set are compared against each other to determine an overall superior NC-classifier.

We then use this classifier to analyze the influence of different feature selection methods on its performance (see chapter 5).

## 3.2 Overview of Feature Selection Methods

The extraction of microarray gene expression data usually provides a huge amount of features (genes) compared to a very low number of samples. Since different genes encode different functionalities in the human DNA, not all of them influence the outcome one would like to predict. A feature is considered a good predictor if it either has a high predictive value by its own or in conjunction with some other subset of features. In order to filter for those genes and disregard the ones with low predictive value, many different feature selection methods have been established in the literature. They are considered very valuable since they do not only reduce the dimension of the classification problem but also filter out noise introduced by non predictive genes. Classification on a subset of selected features therefore becomes faster and – most importantly – more accurate (see e.g. [14, 15, 28, 34, 52]). Considering the prediction of breast cancer outcome on the basis of microarray gene expression levels, different sets of predictive features have been presented in the literature with a remarkably low overlap of almost zero genes [41]. This might be caused by the fact that gene expressions vary a lot subject to the training data the selection is based on. It is therefore very important to choose an appropriate feature selection method which operates prior or parallel to classification.

We will now give a short overview of some common feature selection procedures that might be useful for our classification problem. Generally, feature selection methods can be subdivided into three main classes: *filter*, *wrapper* and *embedded methods* [28, 52]. Filters select features in a preprocessing step, inde-

pendent of the classifier of interest, whereas wrappers and embedded methods use the classifier for measuring the predictive power of the features selected.

Simple filter methods include rank ordering of features according to correlation coefficients (section 3.2.1) or by construction of test statistics that identify features whose gene expression values differ significantly for the classes in the training set. For binary classification problems, it is easy to define test statistics for determining features whose feature vectors come from a different distribution in the two classes. Commonly used tests are, among others, the T-test, the nonparametric Wilcoxon test or the Kolmogorov-Smirnov test [34, 52].

Wrapper methods on the other hand are usually based on sequential forward or backward selection (SFS or SBS) respectively, i.e., they start with an empty (or complete respectively) set of features and subsequently add (or remove respectively) one feature that results in the greatest improve of performance.

Embedded methods are similar to wrappers except for the fact that they incorporate feature selection during training of the classifier, i.e., they perform implicit feature selection.

Compared to filters wrapper and embedded feature selection methods are sometimes critized to be more greedy and computationally complex [28]. Their advantage lies in the connection to the classifier of interest whose performance can therefore be directly influenced (improved), which is not guaranteed when using filters. On the other hand the involvement of the classifier is always accompanied by the risk of overfitting (see section 2.4). Good wrapper and embedded methods should therefore address both problems: computational complexity and overfitting.

In the following we describe some popular embedded feature selection methods, that appear to be suitable for improving nearest centroid classification: nearest shrunken centroid (section 3.2.2), adaboost (section 3.2.3) and the lasso (sections 3.2.4 and 3.2.5). For our purposes, however, we use some of these methods as the basis of a sequential forward selection algorithm only (see chapter 5).

### 3.2.1 Correlation to Disease Outcome

A (computationally) fast approach for selecting features is to consider the correlation of each gene to disease outcome [15, 28, 52], represented by a vector $s$ of $n$ binary components, where $n$ is the number of samples in the learning set. Each sample $x_1, \ldots, x_n$ is associated with its true outcome: Typically $+1$ represents positive and $-1$ negative outcome. For each gene the (Pearson) correlation (see Appendix B for definition) of its expression levels on the $n$ samples to $s$ is calculated and only those genes with high correlation (in absolute value) to the outcome are selected. For example, one could choose those genes whose correlation to disease outcome is unlikely to be observed in randomized data or simply the first $k$ features accoring to the rank order.

The approach of using correlation to disease outcome for feature selection has, among others, been used in [60] and is the basis of the MammaPrint test.

In this case disease outcome is represented by the binary survival vector $s$ whose entries are 1 for good prognosis (i.e. low risk sample) and 0 for poor prognosis (i.e. high risk sample). By using Monte Carlo methods to generate randomized data (from the training data set), the authors decided to use a cut-off value of 0.3 in absolute value to select for features that are highly correlated (or anti-correlated) with disease outcome. By selecting those features whose correlation to disease outcome was either less than $-0.3$ or greater than 0.3 they obtained a set of 231 genes which was rank ordered according to their correlation to $s$.

### 3.2.2 Nearest Shrunken Centroid

The nearest shrunken centroid method (NSC) was first introduced in 2002 by Tibshirani et al. [56], applied to DNA microarray data in 2003 [57], and thereafter modified by Wang et al. [62]. The main idea of this method is to 'shrink' the centroids of the different classes to the overall centroid (a centroid built over all samples independent of their classes) and thereby filtering out those genes that have a low predictive value, i.e., have similar values in all classes. For $i = 1, \ldots, m$ genes and $j = 1, \ldots, n$ samples let $C_k$ be the indices of the $n_k$ samples in class $k \in \{1, \ldots, N\}$. The class centroid for class $k$ at the $i$-th entry representing gene $i$ is defined by:

$$\bar{x}_{ik} = \sum_{j \in C_k} \frac{x_{ij}}{n_k}$$

and the overall centroid by

$$\bar{x}_i = \sum_{j=1}^{n} \frac{x_{ij}}{n}.$$

Let

$$d_{ik} = \frac{\bar{x}_{ik} - \bar{x}_i}{m_k \cdot s_i},$$

where $s_i = \sqrt{\frac{1}{n-2} \sum_{k=1}^{N} \sum_{j \in C_k} (x_{ij} - \bar{x}_{ik})^2}$ is the within-class standard deviation and $m_k$ is given by $m_k = \sqrt{\frac{1}{n_k} - \frac{1}{n}}$. $d_{ik}$ is shrunken by a number $\Delta$ towards zero: $d'_{ik} = \text{sign}(d_{ik})(|d_{ik}| - \Delta)_+$, which causes the $k$-th centroid to be shrunken towards the overall centroid:

$$\bar{x}'_{ik} = \bar{x}_i + m_k \cdot s_i \cdot d'_{ik}.$$

The authors called this shrinkage method *soft thresholding*: Each $d_{ik}$ is reduced by $\Delta$ in absolute value and set to zero, if its absolute value is less than $\Delta$. Since this parameter also determines the shrinkage of the class centroids, the choice of $\Delta$ is crucial for the feature selection procedure: If it is too large, a lot of information is lost, since the centroids are set equal to the overall centroid in many components, which do therefore not contribute to classification any more. If it is too small, the centroids remain almost unchanged and unpredictive features are not filtered out. The optimal $\Delta$ is determined via minimization of

training, test or cross-validation error, where the classification rule for a sample $x$ is defined by:

$$\mathcal{C}(x) = l, \quad \text{s.t.} \quad \delta_l(x) = \min_k \delta_k(x), \quad l \in \{1, \ldots, N\}, \tag{3.6}$$

where $\delta_k(x) = \sum_{i=1}^{p} \frac{(x_i - \bar{x}'_{ik})^2}{s_i^2} - 2 \cdot \log \pi_k$ ($\pi_k$ being the class prior probabilities).

### 3.2.3 Feature Selection via Boosting

The beneficial characteristic of the AdaBoost algorithm (section 3.1.6) is its ability to iteratively adapt a classification model to 'outlier' samples, i.e., samples that would often be misclassified by the weak learner without distribution adjustment. When using this algorithm for feature selection, it is presumed that it iteratively selects features that contain qualitatively new information, i.e., that would be appropriate to pick up those 'outlier' samples that have been misclassified by the combined classification rule defined on the set of currently selected features. One approach of using AdaBoost for feature selection has been introduced in [14]. The author proposed to use decision stumps based on the information gain criterion to select for a new feature to be added to the set of (already) selected features in each boosting iteration step. A learning classifier is used to find the optimal set of features which is defined by the set which maximizes the classifier's (training or validation) performance. The same learning classifier is also used to identify 'outlier' samples, in that the weak hypotheses are given by it's classification results. Therefore, for each iteration step $t = 1 \ldots, T$ the decision stump is only used to select the feature with biggest information gain on the sample distribution $D_t$ out of the set of previously non-selected features. The selection of features due to the information gain criterion is comparable to that of the gini index [46], also it originally evolved in the field of information theory. For the definition of information gain we need first need to declare the following quantities:

**Definition 5** (information entropy)
*The* entropy $H$ *of a discrete random variable* $Z$ *with values* $V_1, \ldots, V_N$ *is defined by*

$$H(Z) = -\sum_{k=1}^{N} \mathbb{P}(Z = V_k) \log_2(\mathbb{P}(Z = V_k)).$$

*Analogously, the* entropy of the set $\mathcal{L} = (X, y)$ *with class labels* $k = 1, \ldots, N$ *is given by*

$$H(\mathcal{L}) = -\sum_{k=1}^{N} p(k) \log_2(p(k)),$$

*where* $p(k)$ *is an estimate of the probability that a sample belongs to class* $k$*.*

A commun estimate of $p(k)$ is given by the relative amount of samples $j$ whose true class label is $k$: $p(k) = \frac{|\{j \in \{1,\ldots,n\} | y_j = k\}|}{n},\ k = 1, \ldots, N$, where $n$ is the number of samples in $\mathcal{L}$.

The definition of entropy of our learning set $\mathcal{L}$ is deducted from the general definition (for random variables) by considering the class variable $y$ of $\mathcal{L}$ as the realization of a random variable. Similarly, we define the conditional entropy for random variables and for the realization given by $y$ and the classification outcome $\mathcal{C}(X)$:

**Definition 6** (conditional information entropy)
*For two discrete random variables $Z_1$ with values $V_1, \ldots, V_N$ and $Z_2$ with values $W_1, \ldots, W_M$, the* conditional entropy $H(Z_2 \mid Z_1)$ *is given by*

$$H(Z_2 \mid Z_1) = \sum_{k=1}^{N} \mathbb{P}(Z_1 = V_k) H(Z_2 \mid Z_1 = V_k),$$

*where $H(Z_2 \mid Z_1 = V_k) = \sum_{l=1}^{M} \mathbb{P}(Z_1 = W_l \mid Z_2 = V_k) \log_2 \mathbb{P}(Z_1 = W_l \mid Z_2 = V_k),\ k = 1, \ldots, N$.*

*Analogously, the* conditional entropy of $\mathcal{L}$ *given the test outcome $\mathcal{C}(X)$ (of the classification $\mathcal{C}$ applied to $X$) is defined by*

$$\begin{aligned} H(\mathcal{L} \mid \mathcal{C}(X)) &= \sum_{k=1}^{N} p(\mathcal{C}(X) = k) H(\mathcal{L} \mid \mathcal{C}(X) = k) \\ &= -\sum_{k=1}^{N} p(\mathcal{C}(X) = k) \sum_{l=1}^{N} p(l \mid \mathcal{C}(X) = k) \log_2(p(l \mid \mathcal{C}(X) = k)), \end{aligned}$$

*where $p(\mathcal{C}(X) = k)$ and $p(l \mid \mathcal{C}(X) = k)$ are estimates of the probabilities for the classification outcome $k$ and the true outcome $l$ provided that the classification outcome is $k$, respectively.*

$p(\mathcal{C}(X) = k)$ and $p(l \mid \mathcal{C}(X) = k)$ are usually estimated by the observations in the learning set $\mathcal{L}$: $p(\mathcal{C}(X) = k) = \frac{|\{j \in \{1,\ldots,n\} | \mathcal{C}(x_j) = k\}|}{n},\ k = 1, \ldots, N$ and $p(l \mid \mathcal{C} = k) = \frac{|\{j \in \{1,\ldots,n\} | \mathcal{C}(x_j) = k,\ y_j = l\}|}{|\{j \in \{1,\ldots,n\} | \mathcal{C}(x_j) = k\}|},\ k = 1, \ldots, N,\ l = 1, \ldots, N$.

**Definition 7**
*For two discrete random variables $Z_1$ with values $V_1, \ldots, V_n$ and $Z_2$, the* information gain $\mathrm{IG}(Z_2 \mid Z_1)$ *of $Z_1$ to $Z_2$ is given by*

$$\mathrm{IG}(Z_2 \mid Z_1) = H(Z_2) - H(Z_2 \mid Z_1).$$

*Analogously, the* information gain *for the learning set $\mathcal{L}$ when applying classification rule $\mathcal{C}$ is given by*

$$\mathrm{IG}(\mathcal{L} \mid \mathcal{C}(X)) = H(\mathcal{L}) - H(\mathcal{L} \mid \mathcal{C}(X)).$$

For concrete realizations ($y$ and $\mathcal{C}(X)$) of the random variables this quantity can be interpreted as a measure of reduction of the expected entropy caused by partitioning of the samples in $\mathcal{L}$ according to $\mathcal{C}$.

Given a training set consisting of a matrix $X$ of measurements for $n$ samples and $m$ genes and a corresponding class vector $y = (y_1, \ldots, y_n) \in \{-1, 1\}^n$, maximizing the information gain using a classification tree is equal to minimizing the conditional entropy

$$\begin{aligned} H(\mathcal{L} \mid \mathcal{C}_i(X)) &= \mathbb{P}(\mathcal{C}_i(X) = -1) \cdot H(\mathcal{L} \mid \mathcal{C}_i(X) = -1) \\ &\quad \mathbb{P}(\mathcal{C}_i(X) = +1) \cdot H(\mathcal{L} \mid \mathcal{C}_i(X) = +1) \end{aligned}$$

over all features $i$ (that have not been selected yet). $\mathcal{C}_i$ denotes the classification according to the value of expression of gene $i$. This is equivalent to the splitting rule described in 3.1.5 since for decision stumps the first (and only) split defines the final classification (both children nodes are terminal).

Here a simple decision rule is used: Given a threshold value $\tilde{x}_i$ and a sample $x_j = (x_{1j}, \ldots, x_{mj})^T$

$$\mathcal{C}_i(x_j) = \begin{cases} -1 & \text{if } x_{ij} < \tilde{x}_i \\ +1 & \text{else} \end{cases}$$

### 3.2.4 Feature Selection via the Adaptive Lasso

The 'least absolute shrinkage and selection operator' (Lasso) [54] is based in the usual regression setting where the sample data $X$ is assumed to be standardized, i.e., $\frac{\sum_j x_{ij}}{n} = 0$ and $\frac{\sum_j x_{ij}^2}{n} = 1,\ i = 1, \ldots, m$, and the class observations of $y$ are assumed independent or conditionally independent given $X$. If we additionally assume $y$ to be centered, i.e., $\bar{y} = 0$, then the lasso estimate $\hat{\beta} = (\hat{\beta}_1, \ldots, \hat{\beta}_m)$ is given by

$$\begin{aligned} \hat{\beta} = \text{argmin}_\beta & \sum_{j=1}^n \left(y_j - \sum_{i=1}^m \beta_i x_{ij}\right)^2 \\ \text{s.t.} & \sum_{i=1}^m |\beta_i| \leq \lambda. \end{aligned} \tag{3.7}$$

The parameter $\lambda \geq 0$ is called tuning parameter since it determines the amount of shrinkage of the solution of the unconstraint version of (3.7), the ordinary least square estimate $\hat{\beta}^0$, towards zero. If $\lambda$ is chosen appropriately, the algorithm therefore implicitly performs variable selection, since only those features $i$ such that $\hat{\beta}_i > 0$ contribute to classification.

An amplification of this algorithm is the *adaptive Lasso* [65] defined by

$$\hat{\beta} = \text{argmin}_\beta \sum_{j=1}^n \left(y_j - \sum_{i=1}^m \beta_i x_{ij}\right)^2 + \lambda \sum_{i=1}^m w_i |\beta_i|, \tag{3.8}$$

for some weight vector $w$, e.g., $w_i = \frac{1}{|\hat{\beta}_i^0|^\gamma},\ i = 1, \ldots, m$ for some $\gamma > 0$. For $w = (1, \ldots, 1)$ the model coincides with (3.7).

### 3.2.5 Feature Selection via the Adaptive Lasso for Cox's Model

The proportional hazards model or Cox's model is given in the setup of survival data. This means that instead of the learning set $\mathcal{L} = (X, y)$ the input data consists of the triple $(s, X, \delta)$, where $s$ is the survival vector (encoding disease free time) and $\delta$ a vector of censoring events. For $\delta_j = 1$ the survival time $s_j$ corresponds to the actual time until disease development is observed, whereas for $\delta_j = 0$ the follow up of the $j$-th sample ended at time $s_j$ until which no disease development has been observed. As before we assume the input data $X$ to be standardized and denote the failure times (disease development at a sample) by $t_1 < \cdots < t_k$. Then the Cox model [12] is given by

$$h(t \mid X) = h_0(t) e^{\beta^T X},$$

where $h(t \mid X)$ is the *hazard* at time $t$ given $X$ and $h_0$ is a baseline hazard function. For simplicity we do not consider ties, i.e., at each failure time $t_l$ there is exactly one failure. The hazard function estimates the survival rate of the individuals under risk given the feature vectors in $X$ starting from time $t$. In order to estimate the value of $h(t \mid X)$ the parameter $\beta$ is estimated through maximization of the partial likelihood

$$L(\beta) = \prod_{\{j \mid \delta_j = 1\}} \frac{e^{\beta^T x_j}}{\sum_{l=1}^{n} e^{\tau_j \beta^T x_l}}, \tag{3.9}$$

where $\tau_j = (\mathbb{1}_{s_l \geq s_j})_l$ is a binary vector determining the individuals at risk at time $t_l - 0$ defined by

$$\mathbb{1}_{s_l \geq s_j} = \begin{cases} 1 & \text{if } s_l \geq s_j \\ 0 & \text{else.} \end{cases}$$

Again we define the Lasso estimate by restraining the solution $\hat{\beta}^0$ of (3.9) via $\sum_i |\beta_i| \leq \lambda$ (for some $\lambda \geq 0$) and the corresponding a*daptive Lasso estimate for Cox's model* by weighting the elements of $\beta$ which leads to $\sum_i w_i |\beta_i| \leq \lambda$ for some weight vector $w$. The maximization of the partial likelihood in (3.9) is equivalent to the minimization of the negative log partial likelihood and we therefore define the adaptive Lasso estimate for Cox's model as the solution of the following equation [55, 64, 66]:

$$\hat{\beta} = \operatorname{argmin}_\beta - \ln\left(L(\beta)\right) + \lambda \sum_{i=1}^{m} w_i |\beta_i|. \tag{3.10}$$

# Chapter 4

# Testing Nearest Centroid Classifiers

The basis of the following investigations will be the MammaPrint algorithm, a nearest centroid classifier for the prediction of tumor outcome, which is described in detail in [60]. We will consider modifications of this classifier with respect to similarity measures, classification rules and the construction of centroids. However, in this section we will not change the set of features selected for MammaPrint; this problem will be addressed later in chapter 5.

## 4.1 MammaPrint

The MammaPrint algorithm has been developed in 2002 on the basis of 78 samples of lymph node negative[1] breast cancer patients younger than 55 years. These samples were subdivided into two groups of low risk and high risk patients depending on whether they developed distant metastasis within 5 years. Initially, for each group a gene expression profile (centroid) was defined consisting of the average expression across a set of 70 preselected features (see section 3.1 and [26]). In principle this algorithm works just as a classical nearest centroid classifier except for the fact that the low risk profile is used as the only centroid (disregarding the high risk profile) and there is a shift incorporated in the classification rule: A sample is classified as low risk, if the cosine correlation[2] to the low risk profile is greater than 0.415.[3] This value was determined by cross-validation in order to reduce the type II error to 10% (i.e. misclassifying a high risk patient as low risk) at the expense of increasing the type I error (i.e. misclassifying a low risk patient as high risk), which correspondes to associating a higher cost to the type II error than to the type I error. However,

[1] Lymph node negative means that no adjacent lymph nodes are affected by tumor metastases.

[2] See Appendix B for definition

[3] A classical nearest centroid classifier would set this value to 0.

the main benefit of the MammaPrint classification compared to clinical characteristics used within the guidelines of NIH [27] and St. Gallen [20] and to the Adjuvant!Online system [47], is the reduction of the type I error rate since this means saving a lot of people from having to undergo chemotherapy [10, 59, 60].

## 4.2 Test Settings

Just like the MammaPrint classifier we aim at classifying tumor gene data according to their probability of developing distance metastasis considering two classes: high risk and low risk. For the clinical data available, we define a patient who did not develop metastasis within 5 years to belong to the low risk group and patients who did develop metastasis in this time to belong to the high risk group. For this purpose we will define several nearest centroid classifiers, which will be compared on two levels:

- The classifier is tuned by optimizing its parameters on a training set. Subsequently its performance is measured on a validation set being independent from the training set.
- A (complete) leave-one-out cross-validation (LOOCV) is performed optimizing the classifier's parameters in each cross-validation step and then applying the classifier (defined by the optimal parameters) to the sample which was left out.

Although it has been shown that the leave-one-out method is in general inferior to $k$-fold-cross-validation (for $k = 5$ or 10) for the estimation of the true classification errors (on independent sets) [6, 33], it is more suitable for our purpose, as we can therefore perform a complete cross-validation which is reproducible whereas even 10*10-fold cross validation produces insufficiently stable results for properly comparing the classifiers (data not shown).

Using these methods we investigate the nearest centroid classifiers considering three aspects:

- What is the impact of using different gene expression profiles as centroids (based on different data sets) on the classifier's performance?
- Is classification based on a low risk profile only (as is done for MammaPrint) optimal or can we improve the classifier by introducing other (additional) profiles and thereby defining new classification rules?
- What is the impact of using cosine correlation compared to other correlation coefficients such as Pearson, Spearman or Kendall as a similarity measure for the nearest centroid classifier?

### 4.2.1 Data

For the following tests we used the NEJM260 and the Transbig data as training and validation sets, where NEJM260 is used for validation of classifiers that have

been trained on Transbig and vice versa.[4] Here training refers to parameter optimization of the nearest centroid classifier (see section 4.2.3). The conjunction of training and validation set is called the learning set $\mathcal{L}$. NEJM260 comprises in total 260 samples from which we excluded those that had insufficient follow-up time and those that originated from the Nature study, i.e., the data set used in [60] for the definition of MammaPrint (see Appendix C). This resulted in a set of 176 remaining samples, 38 being low risk and 138 high risk, which we will refer to as *NEJM176*. The Transbig data set consists of 324 samples, where we also excluded those that did not have enough follow up time and we will therefore only consider a total of 283 samples (simply referd to as *Transbig*) containing 48 low risk and 235 high risk samples. We consider a sample to have insufficient follow-up time, if no metastases have been diagnosed and the follow-up was less than 5 years. This equates to the exclusion of samples for which it is unknown, whether there has been metastasis development within a 5-year time period.

### 4.2.2 Construction of Centroids

First we constructed gene expression profiles that serve as centroids using the 78 samples of the Nature study (where 34 belong to the high risk and 44 to the low risk group). Here, we did not use the original MammaPrint (low risk) profile, which has been obtained by combining the gene expression of the low risk samples measured on the *Hu25K array* and using the signal to noise metric *xdev* [26] for calculating gene expression levels. Later it has been recalculated by analyzing the low risk samples on a 'miniarray', *LD 8-pack array*, custom-built for the MammaPrint test and employing the (log ratio) error-weighting approach presented in section 2.2 [26]. We built new centroids based on the Nature samples using the error-weighting method but measured on the Hu25K array, because the high risk samples have not been reanalyzed on the LD8pack array. For comparison, we also extracted profiles from the Transbig data which was measured on the LD8pack array. In the following we will refer to these profiles by *nLR, nHR, tLR* and *tHR* (see table 4.1 for an overview of the definitions). Please note that the centroids built on the Nature data are independent of the

| profile name | description |
|---|---|
| nLR | *nature Low Risk* profile averaging over expression levels of the Nature low risk patient group |
| nHR | corresponding *transbig High Risk* based on the Nature data. |
| tLR | *transbig Low Risk* profile averaging over expression levels of the Transbig low risk patient group |
| tHR | corresponding *transbig High Risk* profile based on the Transbig data |

Table 4.1: Profiles used as centroids

[4] See Appendix C for an overview of the data sets used.

data sets used for training and validation. When evaluating the test results, we should keep in mind that this does not hold true for tLR and tHR.

### 4.2.3 Classification Rules

Depending on the profiles used, different kinds of prediction rules for nearest centroid classifiers have been established depending on the parameters *threshold* and *gap* and the kind of profile used (table 4.2). Although basing the classification on just one (low risk) profile has been considered sufficient for the MammaPrint algorithm, we would like to also incorporate classifiers based on two profiles into the tests. For this purpose, we defined two new classification rules: R3 and R4. The former is the classification rule being most similar to the classical nearest centroid rule: A sample is classified low risk, if its similarity is at least *gap* counts greater than the similarity to the high risk profile. The last rule, R4, is a combination of R1 and R3: it assigns a sample to the low risk class if the conditions of R1 and R3 hold true for some values of *gap* and *threshold*.

| Name | Profile | Parameter | Rule |
|---|---|---|---|
| R1 | LR | threshold | If the sample's correlation to the low risk profile is greater than *threshold*, the sample is classified low risk and high risk otherwise. |
| R2 | HR | threshold | If the sample's correlation to the high risk profile is greater than *threshold*, the sample is classified high risk and low risk otherwise. |
| R3 | LR, HR | gap | If the difference between the sample's correlation to the low risk profile and the correlation to the high risk profile is greater than *gap*, the sample is classified to be low risk and high risk otherwise. |
| R4 | LR, HR | threshold, gap | If the sample's correlation to the low risk profile is greater than *threshold* and the difference between this correlation and the correlation to the high risk profiles is greater than *gap*, the sample is classified to be low risk and high risk otherwise. |

Table 4.2: Classification rules for the modified nearest centroid classifier

Graphically, these classification rules can be view as a linear separation of the 2-dimensional space which is the range of the mapping $\Phi : \mathcal{X} \leftarrow \mathbb{R},\ x \mapsto (sim(x, \mathrm{LR}), sim(x, \mathrm{HR}))$, where $sim(x, \mathrm{LR})$ and $sim(x, \mathrm{HR})$ is the similarity measure of the sample $x \in \mathcal{X}$ to the low risk and the high risk centroid, respec-

tively. Figure 4.1 shows the separation of $\Phi(\mathcal{X})$ according to a threshold based rule (horizontal line) and a gap based rule (diagonal line). The gray areas right of the threshold line and below the gap line indicate samples that would be classified low risk by either of the two; the intersection of the two areas (dark gray), represents samples that would be classified low risk by a classifier applying R4.

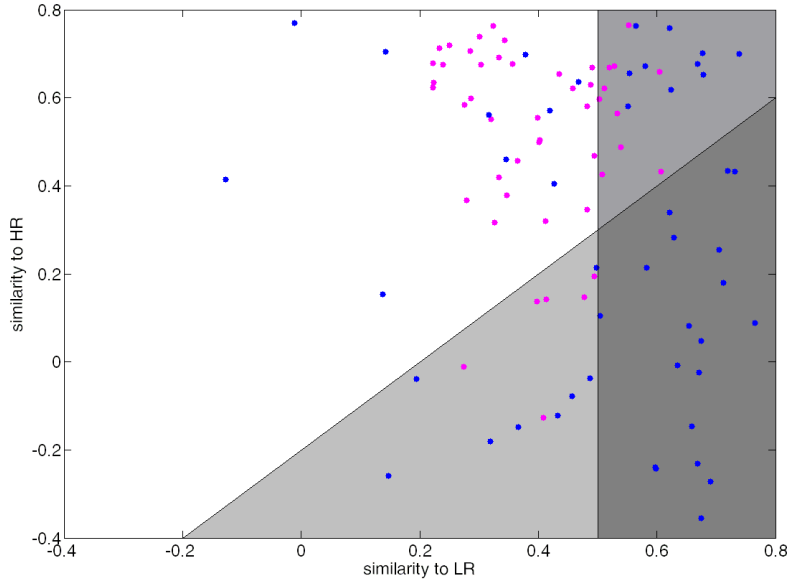



Figure 4.1: Example of a binary classification problem: blue and pink dots represent samples of two different classes (high risk and low risk). The samples similarity to the LR-centroid is plotted against their similarity of the HR-centroid. Classification separates the space according to threshold-based (vertical line), gap-based (diagonal line) or threshold-gap-based (vertical and diagonal line) classification rules.

### 4.2.4 Similarity Measures

Considering nearest centroid classification, naturally the question of how to choose an appropriate distance or similarity measure arises. In the particular problem setting of classifying tumor or, in general, disease outcome one common approach is using correlations. Besides the commonly known (standard) correlation coefficient (also known as Pearson correlation coefficient), other measures of interrelation have been developed and are frequently used in the literature such as Spearman rank, Kendall's tau or cosine correlation (see Appendix B for

definitions). For MammaPrint the cosine correlation coefficient is used which is simply defined as the cosine of the angle between two vectors.

### 4.2.5 Parameter Optimization

For determining the optimal training parameters (gap and/or threshold) we employed a similar accuracy measure[5] as has been used in [60] for MammaPrint: Considering only those classifiers whose false negative rate (FNR) is at most 10%, i.e., sensitivity $= 1 - \text{FNR} \geq 0.9$, the one with highest specificity is considered optimal.[6] This can be view as the following optimization problem:

$$\begin{aligned} &\max_{\text{param}} \ \text{specificity} \\ &\text{s.t.} \ \ \text{sensitivity} \ \geq \ 0.9 \end{aligned} \tag{4.1}$$

In case there are two parameters defining classifiers with equal specificity (complying with the constraint), we will choose the one with higher sensitivity. If the classifiers also agree on this measure, they are considered equally good and both parameters are stored as optimal. However, for validation we will only use the biggest (or smallest for R2, respectively) optimal parameter. This again is done to emphasize the importance of achieving high sensitivity: A higher threshold or gap parameter (or smaller threshold parameter for R2, respectively) results in higher sensitivity accepting a lower specificity measure.

We chose this method in contrast to the standard approach of measuring the *misclassification error rate*, because we want to put special emphasis on the *FNR* (type II error rate), i.e., the rate of misclassifying a high risk patient as low risk. By restricting the *FNR* to 10%, we make sure that in the training set at most 10% of the high risk patient are incorrectly classified. One could argue about how to choose this threshold value. Here we decided to take 10% since this coincides with the threshold defined in [60] for the development of MammaPrint. We think it is a reasonable choice, because it keeps the type II error rate very low without demanding too much of the classifier, i.e., it is still allowed to misclassify some potential 'outlier' samples.

For classification rules R1, R2 and R3 the optimization has been implemented by an approach based on the calculation of ROC curves: Since for these rules, there is only one parameter to be optimized, we could easily extract a vector of scores, which estimates the probability for a sample to be classified positive. This is been done by calculation of the similarity of the samples $x = (x_1, \ldots, x_n)$ to the centroid(s) and calculating the scores depending on the classification rule:

**R1** $s(x) = (-\text{similarity}(x_j, \text{LR}))_j$

[5] Definitions of the error measures can be found in section 2.3

[6] Please note that any of our classifiers will fullfill this constraint for suitable parameters as one could for instance force the sensitivity to be 1 by just classifying all samples positive, i.e., high risk

**R2** $s(x) = (\text{similarity}(x_j, \text{HR}))_j$

**R3** $s(x) = (\text{similarity}(x_j, \text{HR}) - \text{similarity}(x_j, \text{LR}))_j$

The vector of scores $s$ is then ordered in ascending order and for each entry the classifier's sensitivity and specificity measure is calculated (where $-s_j$ (for R1 and R3) or $s_j$ (for R2) are taken as parameter values, respectively). Of the resulting sensitivity and specificity vectors, sens and spec, we extract those indices $I = \{k, k+1, \ldots, n\}$ whose sensitivity values are above the constraint of 0.9. The index set $I$ is of that form because of the ordering of $s$:
Suppose $s_1 < \cdots < s_n$. Using the score value $s_j$ as cutoff (see section 2.4.2), the first $j$ samples with scores $s_1, \ldots s_j$ are classified negative and the rest positive. Thus, changing to $s_{j+1}$ as cutoff, results in a change of the class of the sample corresponding to $s_{j+1}$ from positive to negative. Therefore, if the true outcome of this sample is positive, it is now misclassified and specificity decreases, but sensitivity remains unchanged since it only measures falsely positive classified samples. If the true outcome class is negative, then sensitivity increases since the former misclassification has been corrected (specificity remains unchanged). Hence, $\text{sens}_{j+1} \geq \text{sens}_j$ (and equivalently $\text{spec}_{j+1} \leq \text{spec}_j$) holds for all $j = 1, \ldots, n-1$.

Given $I$, we then take the maximum of $\text{spec}_j$ over all $j \in I$ which is obtained at $j = k, \ldots, l$ for $l \leq n$. By the previous considerations we get that

$$\text{spec}_k = \cdots = \text{spec}_l = \max(\text{spec}_j \mid j \in I)$$

and

$$\text{sens}_l \geq \cdots \geq \text{sens}_k,$$

where equality only holds if the samples $x_k, \ldots, x_l$ are equal with respect to the similarity measure to the centroid(s). We therefore set $s^* = -s_l$ (for R1 and R3) or $s^* = s_l$ (for R2) as the optimal parameter value. For storage $s^*$ is rounded at the fourth post decimal position either up (for R1 and R3) or down (for R2). This is due to the fact that the classification rules are defined by $s(x) > s^*$; so by rounding $s^*$ up (or down for R2) the classification result remains unchanged on the training set (except if there would be another sample score being at most $10^{-4}$ greater than $s^*$).

For R4 on the other hand a more greedy approach has been applied since one cannot calculate such a probability score (as the classification rule depends on two criteria that have to be fulfilled at the same time). Here we tested for a step length of 0.01 within the maximal possible range $[-2, 2]$ and $[-1, 1]$ every parameter value for gap and threshold, respectively. Please note that using this approach for the before mentioned classification rules would result in an optimal parameter of $\hat{s} = \lfloor s_l/100 \rfloor \cdot 100$, where $\hat{s}$ and $s^*$ determine the same classification on the training set, again up to rounding errors. Here, we obtain a less accurate parameter value than for the previous method which is due to the extensive computational effort caused by the larger amount of parameter values that need to be tested.

| Correlation | Threshold | Gap | Diagnostic Table | Specificity | Sensitivity |
|---|---|---|---|---|---|
| Cosine | -1.0 - 0.42 | 0.46 - 0.49 | 35 75<br>3 63 | 0.4565 | 0.9211 |
| Pearson | -1.0 - 0.27 | 0.38 - 0.4 | 35 78<br>3 60 | 0.4348 | 0.9211 |
| Spearman | -1.0 - 0.4 | 0.41 | 35 81<br>3 57 | 0.4130 | 0.9211 |
| Kendall | -1.0 - 0.29 | 0.29 - 0.3 | 35 81<br>3 57 | 0.4130 | 0.9211 |

Table 4.3: Performance of the nearest centroid classifaction on the NEJM176 training set applying R4 with profiles nLR and nHR. Optimal parameters are tested within the maximal possible ranges of $[-1, 1]$ for threshold and $[-2, 2]$ for gap. The test step length was chosen to be 0.01 for both.

## 4.3 Results

Using the *Matlab* program [53] we implemented tests on nearest centroid classifiers considering the three characteristics defined before: we performed a validation (and cross-validation) on each classifier being defined by (1) the data the centroid has been extracted from, (2) the classification rule used and (3) the similarity measure applied.

Already during training it turned out that apparently R4 does not produce very meaningful classifiers as it shows that for most test cases the best way to choose the parameters is such that the classification rule reduces to R3: As can be seen in table 4.3, for each distance measure the optimal threshold lies within some interval $[-1, x]$ which means that all classifiers with threshold values in this range (and corresponding optimal gap values) perform equally good, i.e., produce the exact same classification on NEJM176. Since the condition 'similarity to low risk profile $> -1$' is always fulfilled, a threshold value of $-1$ does not contribute to the distinction of the sample classes. Hence, this classifier equates to the one using R3 on the same profiles. We therefore decided to exclude this classification rule from any further tests knowing that it is also computationally the most expensive one: Testing for optimal parameters has quadratic complexity compared to the other rules since two parameters have to be determined at a time. Since we use a different approach for parameter optimization for R3 (compared to R1,R2 and R4) as explained in section 4.2.5, the difference in computation time is even more pronounced.

Using the remaining three classification rules, which we will from now on for simplicity call LR (R1), HR (R2) and LRHR (R3), we consider a total of 24 different classifiers: Employing 3 classification rules combined with 4 similarity measures to gene expression profiles extracted from 2 data sets. These classifiers are now tested against each other via measuring their performance on LOOCV and validation on an independent data set.

Let us first focus on the latter performance measurement: Training (i.e. parameter optimization) is conducted on NEJM176 or Transbig data, keeping

the respective opposite set for validation. As defined in 4.1 we employe the following accuracy criterion on the classifiers for training: sensitivity $\geq 0.9$ and specificity as big as possible. This criterion can of course not be used to evaluate performance on validation sets, since in this case error rates are expected to be higher than on the training set. On account of this, we define a slightly different criterion to be used as measure of the classifiers performance on a validation set:

$$\begin{aligned} &\max_{\text{param}} \ \text{specificity} \\ &\text{s.t.} \ \ \text{sensitivity} \ \geq \ 0.85. \end{aligned} \tag{4.2}$$

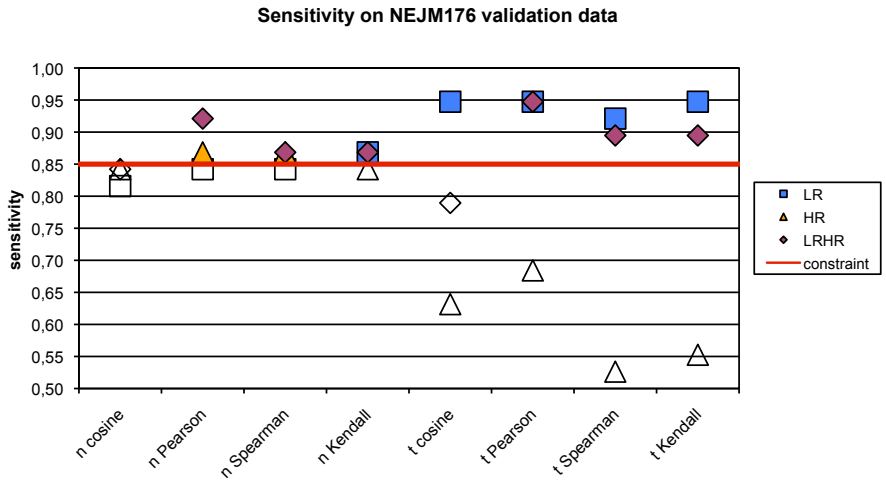
Evaluating the classification performance, we therefore first have a look at the sensitivity of the classifiers that have been validated on NEJM176 and Transbig (i.e. trained on Transbig and NEJM176 respectively) and exclude those that have a sensitivity value of less than 0.85 (figure 4.2). Together we exclude 14 of the 24 classifiers (11 one the first and 3 on the second validation), because their sensitivity is less than the given threshold for one of the independent validations (figure 4.2).

Additionally we perform leave-one-out cross-validation on NEJM176 and Transbig for all classifiers. For all cross-validations, we observe sensitivity values higher than 0.85 (in particular higher than 0.8947); thus, we do not exclude any classifiers based on LOOCV.

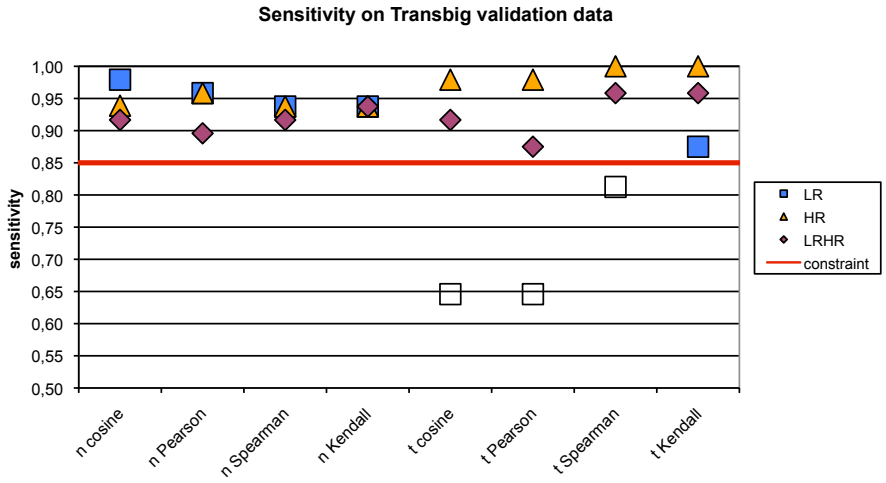
The remaining classifiers are compared based on their average specificity value taken over LOOCV on Transbig and NEJM176 as well as validation on these data sets after training (for determining optimal parameters) on the respective opposite set (figure 4.3). We would like to point out, that the values are very similar, if we take the average over the independent validations only (for details see Appendix D). The classifier using classification rule R3 (LRHR) with Transbig centroids and Pearson correlation as similarity measure is superior to the rest with respect to the quality criterion defined above.

Taking a closer look at figures 4.2 and 4.3, we can see that it indeed makes a substantial difference which data is used for centroid construction. For the Nature-centroids (n-centroids) we see a very similar performance for all classification rules and similarity measures indicating that the low risk profile provides as much information as the high risk profile (and as both together), i.e., they are all (almost) equally suitable for distinguishing low risk samples from high risk samples. This observation illustrates and endorses the use of just one centroid for the MammaPrint algorithm.[7] For the Transbig-centroids (t-centroids) on the other hand, we observe a very inferior performance when using either tLR or tHR alone: almost all classifiers being defined on just one of the two profiles are excluded by the sensitivity-constraint and do also – in general – not provide high specificity values. Nevertheless, combining both of them within one classifier (tLRHR) results – when using Pearson correlation as similarity measure – in the overall best classification. For the n-centroid based classifiers

[7]Please keep in mind that the nLR profile used here is based on the same samples as the MammaPrint profile.




(a)




(b)

Figure 4.2: Sensitivity values of the 24 tested nearest centroid classifiers on two validation sets (NEJM176 (a) and Transbig (b)), where training has been performed on the respective other set. Classifier with sensitivity below 0.85 in one of the two validations will be excluded for specificity comparison.

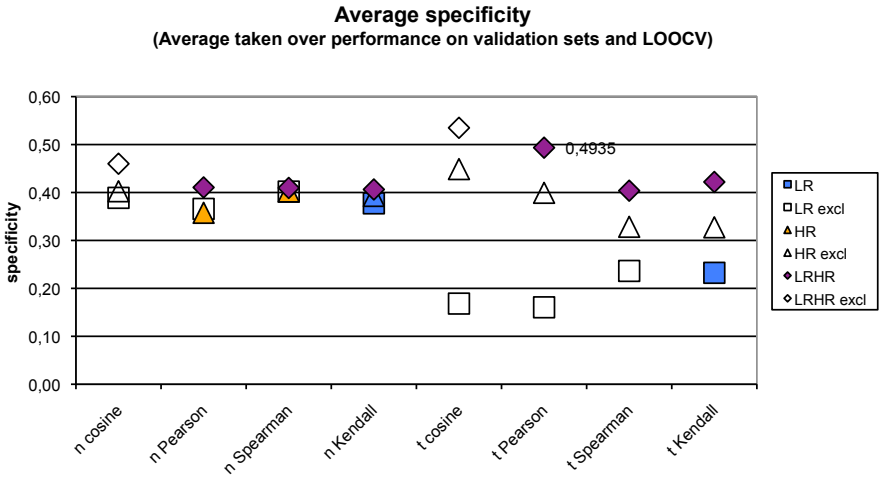



Figure 4.3: Average specificity taken over validation and LOOCV on NEJM176 and Transbig for all 24 tested nearest centroid classifiers. Empty symbols indicate the exclusion of the classifier because of failing the sensitivity constraint in one of the two validations. The highest specificity value of all classifiers fulfilling the sensitivity criterion is obtained by t-Pearson-LRHR (0.49).

we observe the same tendency, but less pronounced (e.g. many sensitivity values of those classifiers that are excluded due to the sensitivity constraint, are still very close to 0.85). Another interesting observation is, that all classifiers using cosine correlation as similarity measure are excluded by this constraint.

A ROC curve analysis (see section 2.4.2) further illustrates the inferior performance of the tLR profile on the NEJM (figure 4.4) as well as on the Transbig data (figure 4.5) which results in a classification hardly better than chance: Averaging the area under the curve, AUC, over each four classifications (employing different similarity measures) results in $\overline{\text{AUC}} = 0.58$ and $\overline{\text{AUC}} = 0.56$ compared to the expected AUC-value 0.5 of a random classifier lying along the diagonal from $(0, 0)$ to $(1, 1)$. In contrast to this, classification based on the Nature low risk profile (figures 4.6 and 4.7) is far from being random: $\overline{\text{AUC}}$ is almost 0.7 in both cases.

The poor performance of the tLR-based classifiers on the Transbig data set is particularly remarkable as the test data coincides with the data used for centroid construction, thus low (FN and FP) error rates should be expected. The same effect, though alleviated, can be observed for classifiers using the Transbig high risk profile only (figure 4.9). However, also for the ROC analysis this phenomenon cannot be observed when both profiles are used as centroids as is shown in figure 4.11.

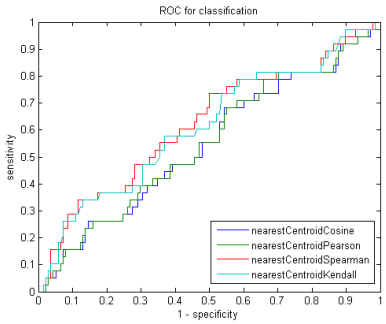



Figure 4.4: ROC curve of the tLR classifiers on the NEJM176 data set. $\overline{\text{AUC}} = 0.5837$.

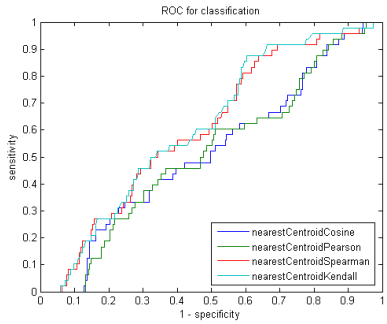



Figure 4.5: ROC curve of the tLR classifiers on the Transbig data set. $\overline{\text{AUC}} = 0.5631$.

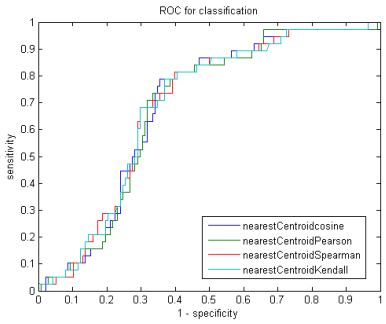



Figure 4.6: ROC curve of the nLR classifiers on the NEJM176 data set. $\overline{\text{AUC}} = 0.6845$.

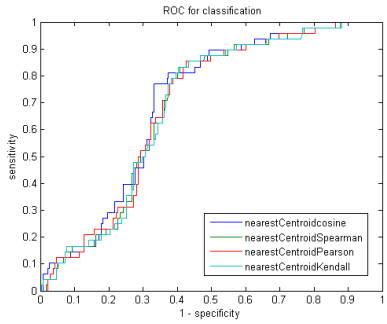



Figure 4.7: ROC curve of the nLR classifiers on the Transbig data set. $\overline{\text{AUC}} = 0.6899$.

Having a closer look at the Transbig profiles, we encountered a strong correlation between the Transbig low and high risk profiles (tLR and tHR): cosine correlation of 0.6453 and Pearson correlation of 0.693, compared to approx. $-0.0474$ and 0.05 for the Nature profiles. This might be an explanation for the need of both Transbig profiles to effectively classify data: As the centroids themselfs are very close to each other, i.e., have a high similarity measure, a sample close to the low risk centroid might also be close to the high risk centroid and we therefore cannot define a meaningful threshold value which allows to distinguish low risk from high risk samples.

For the n-centroid classifiers there is not such a strong need for two reference profiles, since the centroids are more distant. This coincides with the observations made before (figures 4.2 and 4.3). However, considering the ROC curves in figures 4.10 and 4.8 gives reason to assume that also in this case at least LR- and HR-centroid classifications that give a sensitivity value around 0.9 are slightly inferior to those on both profiles.

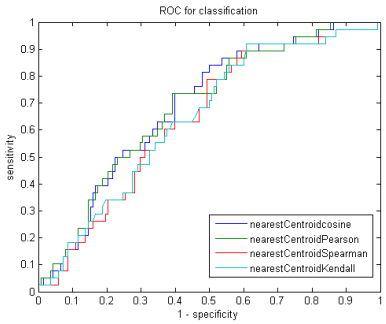



Figure 4.8: ROC curve of the nHR classifiers on the NEJM176 data set. $\overline{\text{AUC}} = 0.6675$.

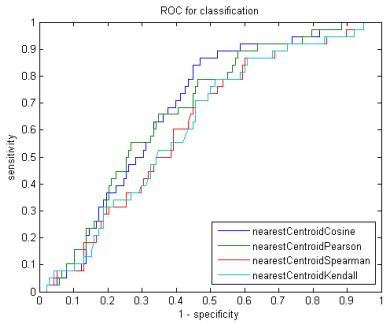



Figure 4.9: ROC curve of the tHR classifiers on the NEJM176 data set. $\overline{\text{AUC}} = 0.6491$.

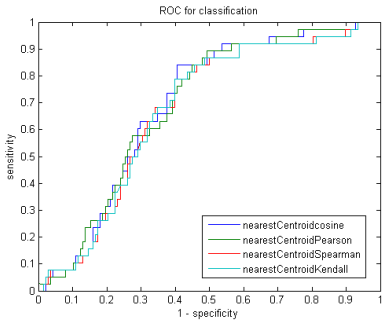



Figure 4.10: ROC curve of the nLRHR classifiers on the NEJM176 data set. $\overline{\text{AUC}} = 0.6846$.

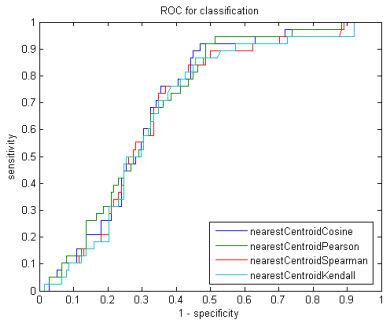



Figure 4.11: ROC curve of the tLRHR classifiers on the NEJM176 data set. $\overline{\text{AUC}} = 0.6931$.

In contrast to the previous analysis (figures 4.2 and 4.3), a comparison of the four distance measures gives inconclusive results, as non of the correlation coefficients outperforms any of the others on the basis of ROC curve analysis. This gives rise to assume that the seemingly inferiority of the cosine-based classifiers on the evaluation made before can be due to random fluctuations.

We would also like to point out the good performance of the tLRHR classifiers (figure 4.11) - compared to the others - especially in the area of the true positive rate (sensitivity) being around 0.9, which is our region of interest, since we put this value as a sensitivity-constraint for measuring classiifcation performance.

## 4.4 Conclusion

In this test setting, we compared a total of 24 nearest centroid classifiers defined by: (centroid construction data) x (classification rule) x (similarity measure) ($2 \cdot 3 \cdot 4 = 24$). Comparison has been done via performance measurement (given by sensitivity and specificity) of the classifiers on two different data sets: NEJM176 and Transbig. In order to control the amount of falsely positive classified samples we applied the quality criteria (4.1) and (4.2) for training and validation, respectively.

Leave-one-out cross-validation as well as (training and subsequent) validation on an independent data set and ROC curve analysis have been conducted. The analyses indicate a superior performance of the classifier defined by Transbig centroids using Pearson's correlation coefficient for similarity measurement and the gap based classification rule LRHR (R4) (see figure 4.3). In the following this classifier will therefore be called *best nearest centroid classifier (bNC)*. bNC fulfills the optimality criterion constraints of (4.2), i.e., it achieved sensitivity values $\geq 0.85$ for both LOOCVs and both independent validations. Furthermore, within the group of 10 classifiers, that comply with the sensitivity constraint, it provides the highest average specificity value (averaging over the LOOCVs and the independent validations). Performance details of bNC as well as optimal gap values obtained by training on the two data sets NEJM176 and Transbig are shown in table 4.4[8]. The optimal gap values obtained during training (according to (4.1)) coincide with the respective optimal LOOCV parameters which are determined via choosing the optimal (training) gap value that occurred most frequently within all correctly classified left out samples.

Training on NEJM176

| Optimal gap-parameter | *0.2086* | |
|---|---|---|
| | Validation | LOOCV |
| Sensitivity | 0.8750 | 0.8947 |
| Specificity | 0.5191 | 0.5145 |

Training on Transbig

| Optimal gap-parameter | *0.2718* | |
|---|---|---|
| | Validation | LOOCV |
| Sensitivity | 0.9474 | 0.8958 |
| Specificity | 0.4565 | 0.4723 |

Table 4.4: Performance (measured by sensitivity and specificity) and optimal training gap parameters of the t-Pearson-LRHR classifier: bNC. Validation performance is measured on the set not used for training, i.e., on Transbig in the first case and NEJM176 in the second.

Considering the different dimensions, which define our nearest centroid classifiers, separately we can furthermore observe some general trends:

[8]More detailed information on the optimal parameters and training and validation performances for the remaining classifiers is given in Appendix D.

**similarity measure** The comparison of the different similarity measures (cosine, Pearson, Spearman and Kendall correlation coefficients) reveals only minor differences. ROC curve analysis showed that performance curves of four similar classifiers varying in the similarity measure dimension only are very close and cross multiple times within the ROC space. The different performances, that we observed on validation sets for varying similarity measures of a classifier, might therefore be due to random fluctuations. On the other hand, we also need to keep in mind that although ROC curves give a more general overview of classifier performances (considering all possible parameter values at once), they show training performance only and therefore do not give insight into the performance on 'unknown' data sets (validation). Focusing on validation performance on independent data sets, classifiers using cosine correlation showed to be least stable with respect to the sensitivity constraint. All six of them that were tested here fail to reach the constraint value of 0.85 in one of the two validations (figure 4.2).

**classification rule** The combined use of two centroids (applying classification rule LRHR) is generally superior to using just one centroid: Six out of eight LRHR-classifiers fullfil the sensitivity constraint for all validations (and cross-validations) compared to only two out of eight LR- and HR-classifiers, respectively. Furthermore, the average specificity values of the LRHR-classifiers are always greater than the averaged specificity of the LR- and HR-classifiers within the group of classifiers varying only by the classification rule applied (see figure 4.3 and Appendix D). This trend can be confirmed by the ROC curve analysis performed when focusing on the area where sensitivity $\approx 0.9$. The phenomenon can be observed for both t- and n-centroids, although it is much more pronounced for the t-centroids than for the n-centroids.

**centroid construction data** As we have just seen, there is a huge difference between classifiers using t- and n-centroids respectively. These differences are mainly reflected by more strongly varying performances within the t-centroid group when modifying one of the other two dimensions. Therefore, we do not think, that the difference in the data size (283 samples (Transbig) versus 78 samples (Nature)) has a considerable impact on the classifier's performances, since in this case we would expect stronger varying performances within the n-centroid group. However, we would like to make some remarks on two very important issues that have to be taken into account:

First, the 70 genes we based our classifications on have been selected in such a way that they optimize a NC classifier which uses centroids that are similar to nLR and nHR [60]. In particular, the MammaPrint (LR-) classifier was found to be equally good as the corresponding LRHR-classifier (section 4.1). Therefore, it is not surprising that the performances of nLR classifiers are found to be similar to those of the nLRHR classifiers. For

the t-centroid classifiers on the other hand, the features selection is independent of the Transbig data and we could even see that both centroids are very similar with respect to the cosine and Pearson correlation coefficient (correlation coefficient $\approx 0.7$) while the n-centroids are more distant (correlation coefficient $\approx 0$) (see section 4.3). Those measures have been used for the selection of the 70 genes.

Second, we would like to point out once more that we used the Transbig data set for centroid construction and as part of the learning set, whereas the n-centroids have been constructed on the Nature data set and NEJM176 and Transbig do not contain samples of this set. Thus, comparing validation performance of t- and n-centroid classifiers on Transbig, might be biased due to the fact that the t-centroids consist of the average gene expressions of this data set. However, t-LRHR classifiers also showed superior performance to n-LRHR classifiers for the validation on NEJM176 (see Appendix D for details).

Thus, we conclude that declaring bNC the best performing classifier is justified and focus on this classifier for studying the influence of different feature selection methods on its performance in the following chapter.

# Chapter 5

# Feature Selection

In the previous chapter our focus was on the definition of an optimal nearest centroid classifier based on 70 genes, which were selected during the definition of MammaPrint in [60]. These 70 genes were chosen out of a set of 231 genes which had been determined on the basis of their correlation to disease outcome as has been described in section 3.2.1. By rank ordering of the 231 genes according to the correlation coefficients a list has been obtained. Classifiers defined on the top-5, top-10, top-15, ... genes of this list are then compared with respect to their performance. This is done by means of leave-one-out cross-validation which identifies the set of top-70 genes to be optimal for this classifier.

However, as has been shown by Ein-Dor et al [21], correlation-based rankings of genes strongly depend on the training set chosen and differences between correlations are small. They even proved that many sets of 70 genes ranked between 1 and 770 perform similar to the set of the top-70 genes on the rank list. A multiple random validation strategy revealed that in fact 5 of 7 classification methods published do not classify patients better than chance due to strong dependence of the feature selection on an usually very small set of training data [41]. Hence, using correlation coefficient rankings does not seem to be the ideal way of addressing the problem of feature selection for the classification of tumor outcome.

In this chapter we are going to investigate different feature selection methods that have been introduced in section 3.2. This is done by measuring bNC's accuracy when the feature sets selected by those methods are considered as the basis for classification. We view this classifier as a class of classifiers, meaning that we do not pre-specify its optimal gap-parameter, since the two previous choices (0.21 and 0.28 as explained in the chapter 4) have been obtained by training on the set of 70 preselected features.

In the following we would ideally like to start with a set of non-preselected genes. Unfortunately, for most sample data we do not have access to the full genome since it has been measured on user-specific microarrays that only con-

sider the set of 231 (plus some normalization) genes.[1] These genes are measured in (2 times)[2] triplicate on the LD8pack array and in quintuple on the 44k array. Hence we can benefit from the fact that very little measurement noise can be expected. Since it is also computationally much more efficient to work on a smaller set of genes that one can select from, we will in the beginning restrict ourselves to this set of 231 genes knowing that results might be biased due to the preselection step. Thereafter, we will give an outlook of how those methods work when considering a larger base set than just the 231 genes.

Consider now just this set of 231 features, from which we want to select an optimal subset (using different feature selection methods), such that bNC's performance is 'best' on this subset. In order to do so, we apply the same training and validation technique (for the Transbig and NEJM176[3] data sets) as used in chapter 4, except for the fact that we conduct only one training/validation (and no LOOCV). Thereby we obtain one optimal feature set for each feature selection method. The performance of bNC on these subsets is then compared on a third independent data set (LNpos).

The reasons why we decided to use only one validation measurement for determining the respective optimal feature sets are, that first it is more efficient and second the combination of training on Transbig and validation on NEJM176 is the most powerful prediction of classification accuracy: since bNC is defined on two Transbig centroids[4], we only want to measure its performance on the independent set NEJM176.

For the (gap) parameter optimization which is done during training for each set of features, we apply the same optimality criteria as defined before in (4.1):

$$\begin{aligned} &\max_{\text{param}} \ \text{specificity} \\ &\text{s.t.} \ \ \text{sensitivity} \ \geq \ 0.9 \end{aligned}$$

for training and with a lower sensitivity constraint of 0.85 for validation (4.2).

Three of the four following feature selection methods are used to extract a list of subsets of genes $\mathcal{G} = [G_1, \ldots, G_T]$ that serves as input to a *sequential forward selection algorithm (SFS)*:
For each feature set $G_t, \ t = 1, \ldots, T$ we train bNC on Transbig (using only the features contained in $G_t$) and subsequently measure its performance on NEJM176.[5] The set with optimal validation performance according to the optimality criterion defined in (4.2) is chosen as the set of selected features.

[1] See section 2.1 for details regarding the different the microarrays.

[2] One normal and one dye-swap measurement.

[3] Please note that here again, we do not want to incorporate the Nature samples into the NEJM data set, since the 231 genes have been selected on their basis and we do not know what effects that would have on performance measurements.

[4] We constructed centroids of the Transbig data on the set of 231 genes in the same way as described before in chapter 4.

[5] If the set contains less than two features, no classification can be conducted, since at least two distinct values are needed for the calculation of the Pearson correlation. The performance measurements are then set to zero.

In sections 5.1 - 5.4 we select features based on each of the feature selection methods defined in section 3.2:[6] nearest shrunken centroid (NSC), AdaBoost feature selection (AdaBoost),[7] the adaptive Lasso (Lasso) and the adaptive Lasso for Cox's model (LassoCox). These methods are then (in section 5.5) compared against each other by measuring the performance of bNC using the selected feature sets, on a third data set (LNpos), that has not been used so far. Thereafter, in section 5.6 we give an outlook on how these procedures influence the performance of nearest centroid classification when features are selected from a larger set of features than only the one consisting of the preselected 231 genes.

## 5.1 Nearest Shrunken Centroid

As defined in section 3.2.2, the nearest shrunken centroid method is designed to shrink the class centroids to the overall centroid in order to filter for those features whose average expression values differ most in both classes, i.e., those that differ most from the overall centroid in at least on class. For our binary classification problem we thus consider the low and the high risk profiles as centroids $\bar{x}_0$ and $\bar{x}_1$,[8] respectively, whose components are to be shrunken to the overall centroid $(\bar{x}_i)_i = (\sum_{j=1}^{n} \frac{x_{ij}}{n})_i$. This is done by shrinkage of[9]

$$d_{i0} = \frac{\bar{x}_{i0} - \bar{x}_i}{m_0 \cdot s_i} \quad \text{and} \quad d_{i1} = \frac{\bar{x}_{i1} - \bar{x}_i}{m_1 \cdot s_i}$$

for each gene $i = 1, \ldots, m$ by $\Delta$ towards zero:

$$d'_{i0} = \text{sign}(d_{i0})(|d_{i0}| - \Delta)_+ \quad \text{and} \quad d'_{i1} = \text{sign}(d_{i1})(|d_{i1}| - \Delta)_+$$

and thereby shrinking the class centroids towards the overall centroid:

$$\bar{x}'_{i0} = \bar{x}_i + m_0 \cdot s_i \cdot d'_{i0} \quad \text{and} \quad \bar{x}'_{i1} = \bar{x}_i + m_1 \cdot s_i \cdot d'_{i1}.$$

If both $d_{i0}$ and $d_{i1}$ are smaller than $\Delta$, it follows that $d'_{i0} = d'_{i1} = 0$ and therefore $\bar{x}'_{i0} = \bar{x}'_{i1} = \bar{x}'_i$. Hence, gene $i$ does not contribute to the classification rule any more since a sample's distance to both centroids at index $i$ is necessarily the same. The bigger $\Delta$ is chosen, the less genes contribute to classification. The optimal value for $\Delta$ is usually chosen by measuring performance of the classifier defined by the classification rule given in (3.6).

[6] Except for feature selection by correlation, which has been used to determine the set of 231 genes we start with

[7] For AdaBoost it is not necessary to use an additional SFS algorithm, because this method provides the possibility to directly incorporate the performance of a given learning classifier into this embedded method (see section 5.2).

[8] For convenience we use class indices 0 and 1 representing the negative and the positive class instead of 1 and 2, respectively.

[9] See section 3.2.2 for definitions of the variables $m$ and $s$

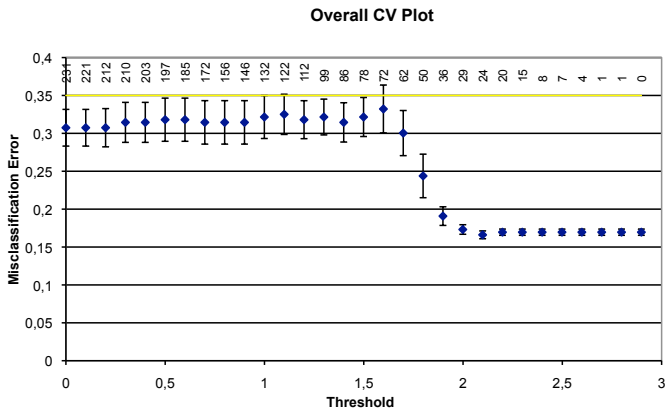
In order to adapt this method to the specific problem of controlling sensitivity as has been explained in chapter 2, we will use the idea of the previously described algorithm modifying the last step of choosing the optimal parameter $\Delta$: we do not measure the performance of the NSC classifier defined by (3.6), but the performance of bNC on the validation set NEJM176 after training on Transbig.

Therefore, we conducted the following steps: First we use the 'pam' program [4], which embeds the R-code [45] for the NSC algorithm in Excel, to extracted a list of genes $\mathcal{G}'$. Second we use this list as an input to the SFS-algorithm described above and thereby determine the optimal set of features.

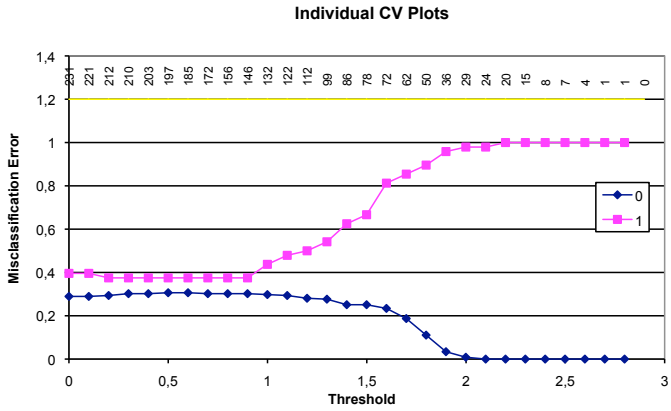
$\mathcal{G}' = [g_1, \ldots, g_{231}]$ is obtained by setting $\Delta = 0$ and thereby ordering the 231 genes by their distances to the overall centroid, i.e., by ordering of the set $\{\max(|\bar{x}_i - \bar{x}_{i0}|, |\bar{x}_i - \bar{x}_{i1}|) \mid i = 1, \ldots, m\}$ . This order corresponds to the inverse order in which they would be shrunken to the overall centroid since genes with larger distances are shrunken to the overall centroid only for bigger values of $\Delta$. Thus, taking the first $k$ elements of this list translates to implicitly choosing $\Delta$ such that the elements $k+1, \ldots, m$ of the class centroids are shrunken to the overall centroid.[10] Figure 5.1 shows the cross-validation error depending on the number of genes selected. Varying values of $\Delta$ (leading to different numbers of selected features) are plotted against the misclassification error (see definition 4) of NSC. The differences in the individual and the overall misclassification errors are due to the fact that the true positive class is far smaller than the true negative class (38 versus 138 samples). Nevertheless, it is surprising that for large values of $\Delta$, NSC classifies all samples negative (misclassification error is 1 for the positive and 0 for the negative class), which might be caused by the preselection of the 231 genes. However, this again emphasizes the need for a different evaluation method for choosing the optimal value of $\Delta$.

Hence, we extract a list $\mathcal{G} = [G_1, \ldots, G_{231}]$, where $G_t = \{g_1, \ldots, g_t\}$ for $g_l \in \mathcal{G}'$, $l = 1, \ldots, t$ and $t = 1, \ldots, 231$. This list is then taken as an input to the sequential forward selection algorithm, where we measure performance (on the NEJM176 validation set) for each subset of features starting with 2 ($G_2$) up to the full set of 231($G_{231}$) features (results are shown in figure 5.2). Validation performance is depicted by the solid line, training performance by the dashed line, where we measure sensitivity (red 'upper' lines) and specificity (green 'lower' lines) in both cases. The maximum validation specificity of 56% (of those classifications fulfilling the sensitivity constraint of 0.85) is reached at a set of 37 features. We conjuncture that further increasing the feature set size does not provide any new information for bNC's classification since we cannot obtain any improvement of its accuracy and furthermore results remain very stable form this point on (indicating that all valuable information is already present in the set of 37).

[10] This implies that these elements do not contribute to classification.




(a)




(b)

Figure 5.1: Threshold = $\Delta$ (and the corresponding number of selected features (top)) is plotted against the 10-fold cross-validation error of NSC on the whole data set (with confidence intervals) in (a) and for the individual groups positive (1) and negative (0) in (b).

## 5.2 AdaBoost Feature Selection

We adopt the approach of using the AdaBoost algorithm for feature selection introduced in [14]. This method requires a weak learner as well as a learning classifier, where the latter is used for performance measurement in the boosting algorithm. Since we can choose bNC as the learning classifier our quality

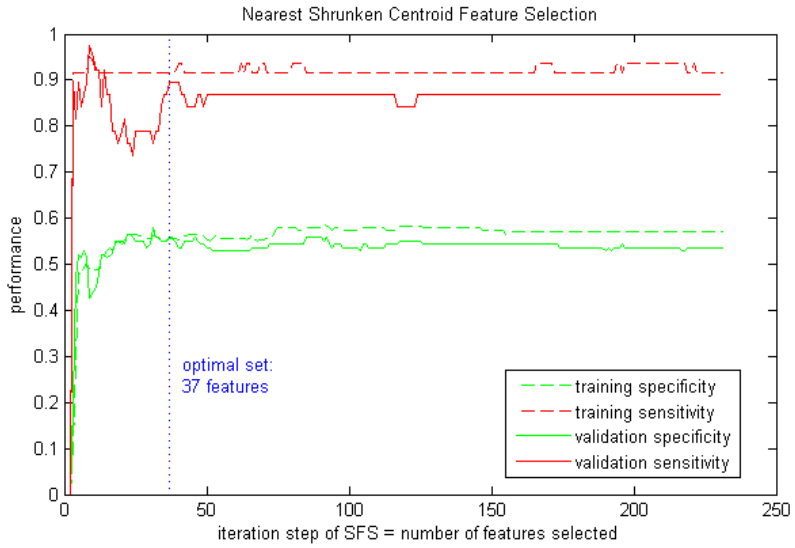



Figure 5.2: Training and validation performance of bNC depending on the number of features selected form the list obtained by the nearest shrunken centroid algorithm. Optimal validation performance is obtained for 37 features.

measurement (4.2) can directly be incorporated into this embedded method. Therefore, we do not need to use it for the extraction of a list of genes that serves as an input to our sequential forward selection algorithm, but use the method as a stand alone feature selection procedure.

We apply (a modified version of) the AdaBoost algorithm (algorithm 1 in section 3.1.6), where in each boosting iteration step $t = 1, \ldots, T$ a decision stump is used as weak learner on the learning set $\mathcal{L} = (X, y)$ given a distribution $D_t$. This tree stump selects from the set of currently non-selected features a gene $i$ such that the corresponding optimal classification rule $\mathcal{C}_i$ maximizes the information gain (see section 3.2.3), i.e., minimizes the conditional entropy

$$\begin{aligned} H(\mathcal{L} \mid \mathcal{C}_i) &= p_{D_t}(\mathcal{C}_i(X) = -1) \cdot H(\mathcal{L} \mid \mathcal{C}_i(X) = -1) \\ &\quad p_{D_t}(\mathcal{C}_i(X) = +1) \cdot H(\mathcal{L} \mid \mathcal{C}_i(X) = +1). \end{aligned}$$

The probability estimate $p_{D_t}$ is given by $p_{D_t}(\mathcal{C}_i(X) = k) = \sum_{\{j \mid \mathcal{C}_i(x_j) = k\}} D_t(j)$ and $\mathcal{C}_i$ is defined (up to changes of signs) by

$$\mathcal{C}_i(x_j) = \begin{cases} -1 & \text{if } x_{ij} < \tilde{x}_i \\ +1 & \text{else} \end{cases}$$

for some optimal threshold value $\tilde{x}_i$.

To obtain the weak hypothesis we modify the AdaBoost algorithm according to [14], in that we use the classification result of the learning classifier (bNC)

instead of the classification result of the weak learner (the decision stump). bNC's weak hypothesis is the outcome of the training on Transbig, i.e., the training performance, where sensitivity is guaranteed to be at least 0.9. On the basis of this the new distribution $D_{t+1}$ is calculated which serves as an input to the weak learner for selecting the next feature in step $t+1$.

The optimal features set is (as in SFS) determined by the performance measurement on the validation set NEJM176. Results depending on the boosting iteration steps are shown in figure 5.3, where the optimal performance (specificity of 56%) is found at a set containing 129 features. Here we can clearly

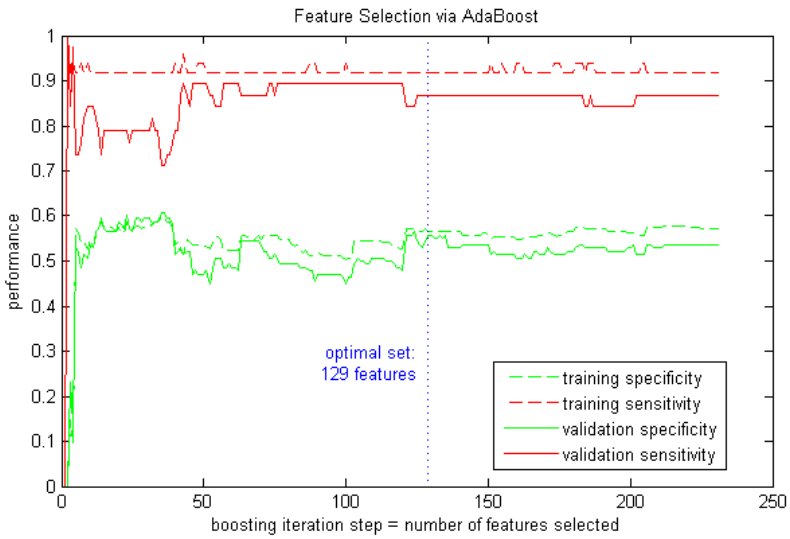



Figure 5.3: Training and validation performance of bNC depending on the number of features selected by the AdaBoost feature selection method. Optimal validation performance is obtained for 129 features.

see poor validation sensitivity measurements up to approximately 50 selected features. Thereafter, specificity fluctuates a lot until the optimal feature set is found and performance stabilizes (around 52% for specificity and 85% for sensitivity). Compared to NSC, a larger set of features is needed to attain the optimal performance, which is very similar to the performance of the optimal NSC set of 37 features.

## 5.3 The Adaptive Lasso for Feature Selection

Recall the definition of the adaptive Lasso algorithm given in section 3.2.4: Let $X \in \mathbb{R}^{m \times n}$ be a standardized data matrix for $n$ samples and $m$ genes ($\frac{\sum_j x_{ij}}{n} = 0$, $\frac{\sum_j x_{ij}^2}{n} = 1$, $i = 1, \ldots, m$) and $y \in \mathbb{R}^m$ the corresponding centered class outcome vector ($\bar{y} = 0$), where the class observations are assumed to be

independent or conditionally independent given $X$. Let $\lambda \geq 0$ be a scalar and $w \in \mathbb{R}^m$ a vector of weights. Then the adaptive Lasso estimate $\hat{\beta} \in \mathbb{R}^m$ is given by (3.8):

$$\hat{\beta} = \text{argmin}_\beta \sum_{j=1}^{n} \left( y_j - \sum_{i=1}^{m} \beta_i x_{ij} \right)^2 + \lambda \sum_{i=1}^{m} w_i |\beta_i|.$$

In order to solve this minimization problem, we will use the algorithm proposed by Zou in 2006 [65]. It defines the weight vector as the inverse of a power of the ordinary least square estimate $\hat{\beta}_0$:

$$w = \frac{1}{|\hat{\beta}_0|^\gamma}$$

for some $\gamma \geq 0$. As their paper lacks an explanation on how to choose $\gamma$, we followed their example and simply set $\gamma = 1$. To solve (3.8) we implemented algorithm 2 which makes use of the Lars algorithm [19].

**Algorithm 2** The Lars algorithm for the adaptive Lasso

1. Define $x_i^* = \frac{x_i}{w_i},\ i = 1, \ldots, m$, where $x_i = (x_{i1}, \ldots, x_{in})$.
2. Solve the Lasso problem

$$\hat{\beta}^* = \text{argmin}_\beta \sum_{j=1}^{n} \left( y_j - \sum_{i=1}^{m} \beta_i x_{ij}^* \right)^2 + \lambda \sum_{i=1}^{m} |\beta_i|.$$

   for all $\lambda$.
3. Compute $\hat{\beta}_i = \frac{\hat{\beta}_i^*}{w_i},\ i = 1, \ldots, m$.

For the implementation of this algorithm in Matlab [53], we used the standard *lscov* algorithm to solve the ordinary least square problem. Since this algorithm cannot deal with values that are 'not a number' (NaN), we used the standard *knnimpute* command for replacing NaN-entries with the corresponding entry of the nearest column in euclidean distance (with the entry of the samples that is closest to the one with the missing number). The solution of *lscov* is then used to define $w$ and hence $x^*$, where the latter together with the outcome vector $y$ is given as an input to the Lars package [51], which solve the Lasso problem of step two. This package can solve the problem for all possible values of $\lambda$, $\lambda^{(1)}, \ldots, \lambda^{(T)}$, that correspond to the solutions $\hat{\beta}^{*(1)}, \ldots, \hat{\beta}^{*(T)}$. At each iteration step $t = 1, \ldots, T$ a feature is either added or deleted from the set of currently selected features. A feature $i$ is delete if the element $\hat{\beta}_i^{*(t)}$ of the Lasso estimate is set to 0 for $\hat{\beta}_i^{*(t-1)} \neq 0$ and added if $\hat{\beta}_i^{*(t)} \neq 0$ for $\hat{\beta}_i^{*(t-1)} = 0$.

Since we are not interested in the weighting of the features $i = 1, \ldots, m$, we omit the last step and take the list of features $\mathcal{G} = [\hat{\beta}^{*(1)}, \ldots, \hat{\beta}^{*(T)}]$ as an input to our sequential forward selection algorithm (results shown in figure 5.4).

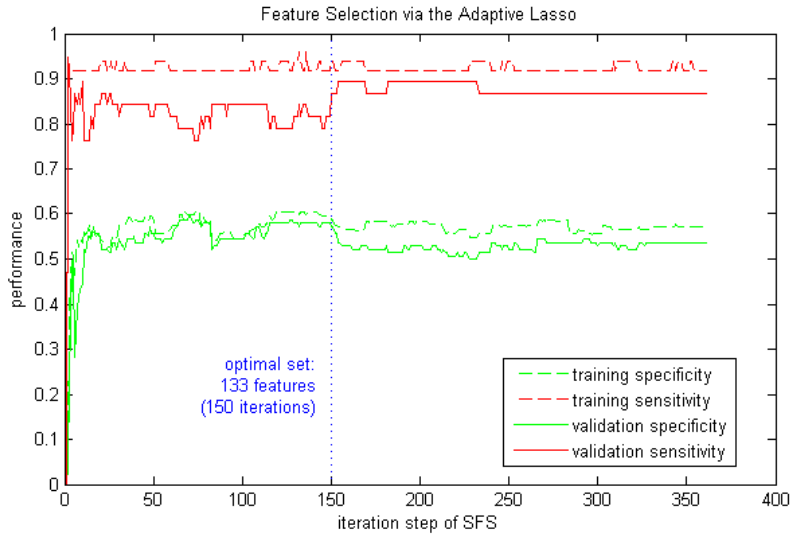



Figure 5.4: Training and validation performance of bNC depending on the iteration steps of the adaptive Lasso algorithm. Optimal validation performance is obtained for 133 features.

For this approach we observe poor performance (with respect to validation sensitivity) up to iteration point 150 where the optimal set of 133 features is found. The validation specificity at this point is 57%, which for further iterations rapidly declines to values slightly above 50%.

## 5.4 The Adaptive Lasso for Feature Selection in the Cox Model

The last feature selection algorithm we consider in this study is the only one that is not based on the usual data input $X$ and $y$, a sample data matrix and the corresponding binary outcome vector, that is defined by the clinical outcome (of metastasis development) within 5 years. The Cox model takes into account continuos measurements of metastasis development: the input is given by the triple $(s, X, \delta)$, where $s$ is called the survival vector, which for each patient $j$ contains the exact time span $s_j$ of either the detection of disease development or the end of the follow-up time. Hence, we do not restrict ourselves to one particular point in time (5 years after diagnosis) where we look at disease development, but incorporate more detailed information into the feature selection algorithm. This method therefore provides a more flexible adaptation to the input data $X$ than the before mentioned ones, since it does not depend on an arbitrary definition of class labels [25, 29]. Consider for example two patients, one developing metastasis after 5 years and one day and the other one never de-

veloping distant metastasis. In the previous settings the true outcome class for both patients is the same. Now, the Cox model can account for the differences within each of the two possible outcome classes. Since it might not be known for all patients when (if at all) they developed metastases, the binary censoring vector $\delta$ is introduced, which for each patient $j$ encodes whether $s_j$ is the time of the detection of disease development or the end of the follow-up time. In other words, $\delta_j = 0$ indicates that patient $j$ developed distance metastases after time $s_j$ and $\delta_j = 1$ indicates that he did not develop metastases up to time $s_j$. Since in the later case the follow-up ended at time $s_j$ (after diagnosis), we do not know whether the patient developed metastases after that point in time (i.e. the data is censored).

For failure times $t_1 < \cdots < t_k$ (corresponding to the elements of the set $\{s_j \mid d_j = 0,\ j = 1, \ldots, n\}$ when ordered in ascending order) the adaptive Lasso estimate for Cox's model is given by (3.10):

$$\hat{\beta} = \text{argmin}_\beta - \ln\left(L(\beta)\right) + \lambda \sum_{i=1}^{m} w_i |\beta_i|,$$

where $L(\beta)$ is the partial likelihood of the hazard $h(t \mid X)$ at time $t$ estimating the survival rates of the patients at risk starting from time $t$ (see section 3.2.5):

$$h(t \mid X) = h_0(t) e^{\beta^T X}.$$

**Algorithm 3** The Lars algorithm for the adaptive lasso in Cox's model

1. Calculate the estimate $\beta^C = \max_\beta \ln(L(\beta))$ and the Hessian $I(\beta^C)$ of the log partial likelihood.
2. Obtain the (Cholesky) decomposition $I(\beta^C) = V^T V$.
3. Compute $u = V\beta^C$ and carry out the transformation $v_i^* = v_i |\beta_i^C|$ for the rows $v_i,\ i = 1, \ldots, m$ of $V$.
4. Solve the Lasso problem

$$\hat{\beta}^* = \text{argmin}_\beta \sum_{j=1}^{n} \left( u_j - \sum_{i=1}^{m} \beta_i v_{ij}^* \right)^2 + \lambda \sum_{i=1}^{m} |\beta_i|.$$

   for all $\lambda$.
5. Compute $\hat{\beta}_i = \frac{\hat{\beta}_i^*}{w_i},\ i = 1, \ldots, m$, where $w_i = \frac{1}{|\beta_i^C|},\ i = 1, \ldots, m$.

As suggested in [66], we calculated the vector of weights $w$ by maximization[11] of the log partial likelihood

$$\ln(L(\beta)) = \sum_{\{j|\delta_j=1\}} \beta^T x_j - \ln\left(\sum_{l=1}^{n} e^{\tau_j \beta^T x_l}\right),$$

where $\tau_j = (\tau_{j1}, \ldots, \tau_{jn})$ is defined by $\tau_{jl} = \mathbb{1}_{s_l \geq s_j},\ l, j = 1, \ldots, n$.

However, we did not follow the approach of [66] any further since we were confronted with problems in the convergence of the proposed modified shooting algorithm that we could not solve. Therefore, we decided to implement algorithm 3 proposed by Zou [64], where in step 1 the inverse weight vector $\hat{\beta}^C$ is calculated according to [66]. Again we omit the last step and take the list of features $\hat{\beta}^{*(1)}, \ldots, \hat{\beta}^{*(T)}$ returned by the Lars package in step 4 as an input to our sequential forward selection algorithm. Results are shown in figure 5.5, where the optimal performance is determined at iteration step 539, for 229 features.

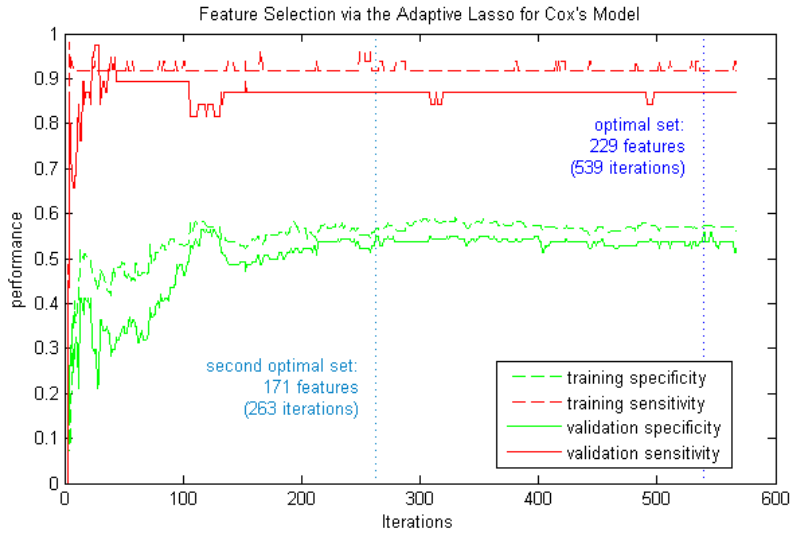



Figure 5.5: Training and validation performance of bNC depending on the iteration steps of the adaptive Lasso algorithm for the Cox model (proportional hazard model). Optimal validation performance is obtained for 229 features.

However, we can see that the performance curve is very stable already before this iteration step and the increase in performance could be an 'outlier' result. Therefore, we also depicted the second best performance measurement obtained at iteration step 263 for a set of 171 features. Clearly, from that point on we observe only minor changes in the validation specificity, similar to the observations we made before for the other feature selection methods. Therefore, we will

[11] We used the standard Matlab routine *fminunc* with the Simplex algorithm for optimization. Here again, *knnimpute* was applied to the data matrix when necessary.

for this method not only consider the optimal feature set of 229 features (validation specificity 56%), but also the second optimal set of 171 features (validation specificity 55%).

## 5.5 Comparison of Feature Selection Methods

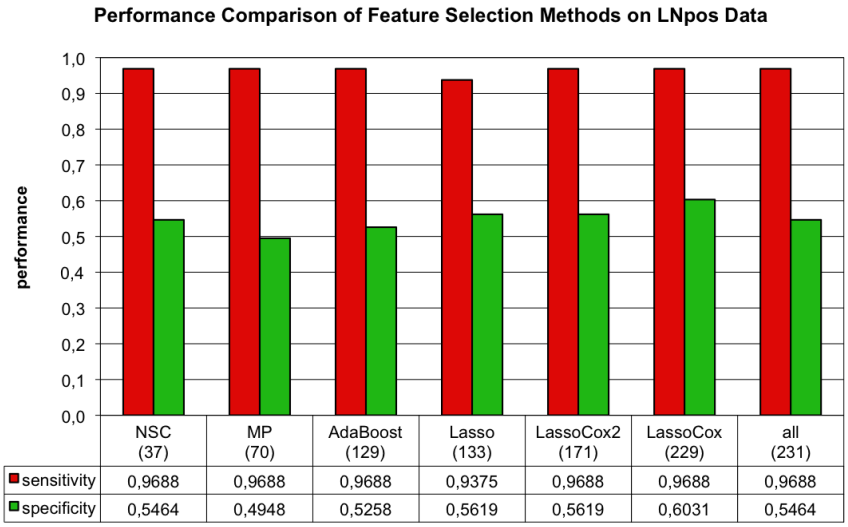
Looking at the performance development as the number of selected features increases, we can generally observe a similar pattern for all feature selection methods:
Each curve starts at 0 indicating that we cannot perform classification (at least 2 distinct values are needed for the calculation of the Pearson correlation coefficient). Then performance increases very fast suggesting that each additional feature adds a lot of information and that using a set of very few features does not suffice for the prediction of distant metastasis development. This characteristic is followed by an interval, where training performance is quite good, but validation yields very unstable results until it peaks at the optimal feature set and thereafter becomes very stable for both training and validation performances. Please note, that only for feature selection via NSC and AdaBoost the number of features coincides with the number of iterations on the x-axis. For the Lasso methods, in each iteration either a feature is deleted or added and the size of the feature set is therefore not (strictly) monotonically increasing. However, since the number of selected features increases over time (given by the iteration steps) from 2 to the full set of 231, we can still consider a general trend of increase in the size of the selected feature subsets.

| classifier | NSC | AdaBoost | Lasso | LassoCox2 | LassoCox |
|---|---|---|---|---|---|
| no of features | 37 | 129 | 133 | 171 | 229 |
| gap value | 0.3603 | 0.2357 | 0.1791 | 0.2313 | 0.2032 |
| train. sensitivity | 0.9167 | 0.9167 | 0.9375 | 0.9167 | 0.9167 |
| train. specificity | 0.5574 | 0.5660 | 0.5872 | 0.5660 | 0.5745 |
| valid. sensitivity | 0.8947 | 0.8684 | 0.8684 | 0.8684 | 0.8684 |
| valid. specificity | **0.5580** | **0.5580** | **0.5725** | **0.5507** | **0.5580** |

Table 5.1: Optimal number of features as well as optimal gap parameter and training and validation performance of bNC on the feature sets determine by different feature selection methods. LassoCox2 denotes the second best feature set for the adaptive Lasso for Cox's model.

Consider now the best performing classifiers on the respective optimal feature sets in some more detail. Table 5.1 shows their training performances on Transbig and validation performances on NEJM176 as well as their optimal (training) gap values and the number of features forming the sets of selected features. The validation performance measurement is used to determine the optimal feature subsets. Clearly, the optimal validation specificity values are

very similar for all tests: between 55% (LassoCox2) and 57% (Lasso). However, as NEJM176 was part of the learning procedure (since it is used for choosing the optimal feature sets) we do not want to compare these five classifiers by their performance on this set. Therefore, we measure their performance on the independent[12] LNpos data set which has not been used before in this analysis (see figure 5.6). The LNpos set consists of samples where the patients are lymph




| | NSC (37) | MP (70) | AdaBoost (129) | Lasso (133) | LassoCox2 (171) | LassoCox (229) | all (231) |
|---|---|---|---|---|---|---|---|
| sensitivity | 0,9688 | 0,9688 | 0,9688 | 0,9375 | 0,9688 | 0,9688 | 0,9688 |
| specificity | 0,5464 | 0,4948 | 0,5258 | 0,5619 | 0,5619 | 0,6031 | 0,5464 |

Figure 5.6: Comparison of bNC classification depending on the feature sets used for training (on Transbig). 'MP (70)' denotes the feature set of 70 features on which MammaPrint was developed and which was the basis for choosing bNC in the previous chapter. 'all (231)' denotes the bNC classifier trained on all 231 features.

node positive for 1 to 3 lymph nodes, i.e., there was a metastasis found in 1 to 3 regional lymph nodes.[13] In contrast to this, the data sets we used so far contain only lymph node negative samples (without any metastases). However, this is still in the scope of samples (defined in chapter 1) that have maximal three positive lymph nodes. The LNpos data sets consists of a total of 226 samples, 32 of them being high risk and 194 low risk samples.

All five classifiers are bNC classifiers that differ by their gap parameter values, which are given in table 5.1 and were determined by training on Transbig. Additionally, we also consider bNC trained on Transbig for the full set of 231 genes, because we are interested in the difference of this classifier to the one trained on the LassoCox feature set of 229 genes, i.e., a feature set where only 2 of the 231 genes were excluded for classification. Surprisingly, the difference in performance between those two sets is remarkable high (specificity increases

[12] Independent of both data sets Transbig and NEJM176 used during learning.
[13] Lymph nodes adjacent to the primary tumor. See section 2.3 for details.

by almost 6% from 54.6% for the set of 231 genes to 60.3% for the set of 229 genes, while sensitivity remains the same (96.9%)). Generally, the performance of the original bNC classifier on the set of 70 genes can be improved by 3 - 11 percentage points, i.e., by 6 - 22% depending on the feature selection method used.

The overall best performance is achieved when using the set of 229 features determined by the adaptive Lasso feature selection method for Cox's model. Even the performance on the Lasso feature set, which showed the most promising results on NEJM176, is inferior to this set.

However, we also see that the performance on the whole set of 231 genes is superior to the one on the 70 selected features and the AdaBoost features and equal to the one after feature selection by NSC. The biggest improvement from no selection (taking all 231 genes) is given by the feature set selected by Lasso-Cox, which is almost identical to the full set, disregarding only two genes. One possible conclusion of this surprising result is, that these 2 genes only introduce noise into classification. The good result when using the preselected set of 231 genes might indicate that it already has a good predictive power and most of the genes are actually needed for the classification of breast cancer outcome. Other explanations are that either the feature selection methods studied here are not as powerful as expected (and therefore comparable to the correlation based selection) or the set of 231 is too small for a basis set and furthermore strongly biased due to the preselection.

In order to get some more insight into the differences between the feature selection methods that give distinctive results regarding the optimal number of selected features, we have a brief look at the overlap of the optimal feature sets in table 5.5. Similar to the results shown in [21], the feature sets obtained in this study also have very little overlap, in that the observed overlap is very close to the expected number of overlapping features when choosing randomly from a hypergeometric distribution with 231 features.[14]

Only when comparing the overlap of AdaBoost and NSC or AdaBoost and the 70 features of MammaPrint, respectively, we can see a considerable difference in the observed and expected number of overlapping features. A possible interpretation of this little overlap is, that indeed (as stated in [21]) there is no unique set that is optimal for the prediction of distant metastasis development in breast cancer tumors. On the other hand, here again, we must take into account that the basic set of 231 features we considered is relatively small and preselected by the correlation of their gene expression (in the Nature data) to the disease outcome (see section 3.2.1).

[14]The random overlap between two subsets $A$ and $B$ of the 231 genes is calculated by the expected number of elements in $B$ that are chosen, when randomly drawing $|A|$ elements from the 231 genes.

(a) Observed overlap

| | MP | NSC | AdaBoost | Lasso | L-Cox2 | L-Cox |
|---|---|---|---|---|---|---|
| MP | 70 | 11 | 20 | 38 | 49 | 69 |
| NSC | | 37 | 27 | 22 | 28 | 37 |
| AdaBoost | | | 129 | 79 | 94 | 128 |
| Lasso | | | | 133 | 95 | 132 |
| L-Cox2 | | | | | 171 | 169 |
| L-Cox | | | | | | 229 |

(b) Expected Overlap

| | MP | NSC | AdaBoost | Lasso | L-Cox2 | L-Cox |
|---|---|---|---|---|---|---|
| MP | - | 11.21 (2.57) | 39.09 (3.48) | 40.3 (3.46) | 51.82 (3.07) | 69.39 (0.65) |
| NSC | | - | 20.66 (2.77) | 21.3 (2.76) | 27.39 (2.45) | 36.68 (0.52) |
| AdaBoost | | | - | 74.27 (3.74) | 95.49 (3.32) | 127.88 (0.7) |
| Lasso | | | | - | 98.45 (3.3) | 131.85 (0.7) |
| L-Cox2 | | | | | - | 169.52 (0.62) |

Table 5.2: The observed overlap of feature sets selected by the different feature selection methods (a) and the expected overlap (and standard deviation in brackets) assuming that the features are randomly chosen according to a hypergeometric distribution (b).

## 5.6 Feature Selection on a Larger Set of Features

In order to get an idea of how the aforementioned feature selection procedures work when considering a larger set of features to select from, we would like to redo the analysis described before on a new enlarged set of genes. However, we cannot conduct the exact same analysis as presented before, since we have little data of the full genome: of the three data sets, Transbig, NEJM176 and LNpos, only NEJM176 is measured on the 44k array, which measures the full genome.

However, if we considered the full genome of approximately 25,000 genes, the analysis would not be very efficient and furthermore a lot of noise would be introduced by genes that do not influence tumor behavior. One possibility of addressing this problem is preselecting genes by a test for differentially expression [17]. We did not have to do that, because for this study we were provided with an already preselected set of 2577 genes that are used for various cancer tests.

With the base set of 2577 genes and the NEJM176 data set[15] we now analyze

[15]Please note, that here the Nature samples could be included into the analysis, but we still do not know, how this would affect the performance measurements, since the set of 2577 genes contains the 231 genes that have been selected during the Nature study.

the feature selection methods NSC, AdaBoost and Lasso again. The LassoCox procedure is not included here, since we encountered numerical problems: First, the calculation of the estimate $\beta^C$ in step 1 of algorithm 3 results in values to be 'not a number' ('NaN') or 'infinity' ('inf') for standard Matlab optimization algorithms. Second, even if we exchange this step by a different approach (e.g. choose $\beta^C = (1, \ldots, 1)$ which defines the (nonadpative) Lasso), we are still confronted with a very slow calculation of the Hessian matrix and numerical instabilities in the Lars algorithm. Hence, we focus on the remaining three algorithms only and conduct a similar analysis as before:
Since the Transbig data is not available on the full genome, we calculate low risk and high risk gene expression profiles form NEJM176 that are used as centroids. Thus, we cannot construct a version of bNC any more, but define a new nearest centroid classifier, bNC2, as the classifier using the NEJM176 centroids, Pearson correlation as similarity measure and the same classification rule (LRHR) as bNC. On account of the limited number of samples provided by NEJM176, we also decided not to split the data into a training and a validation set, but use 5-fold cross-validation instead for determining the optimal feature set. The performance of the different feature selection methods as a function of the iteration steps are shown in figure 5.7. The performance curves are only shown up to maximum of 350 iteration steps, because all optimal feature sets are found within this range. Furthermore, the Lars package used within the adaptive Lasso feature selection algorithm, restricts the number of iterations to less than 350 and therefore, no data is available for the subsequent iterations.[16] Full performance curves for NSC and AdaBoost can be found in figures D.1 and D.2 in the appendix.

Generally, when compared to the previous results a higher average performance (specificity $\geq 0.6$ and sensitivity $\approx 0.85$) can be observed for the first 350 iteration steps. This might of course be a bit overoptimistic, since cross-validation instead of an independent validation measurement is used. Therefore, it is not possible to draw a direct comparison between the two analyses, since the cross-validation is not independent of training. This should be kept in mind for the interpretation of the results in the following part.

Using NSC an optimal set of 32 features is selected, where specificity is 67%. Further increasing the number of features does not yield any improvement and the specificity value remains very stable for increasing numbers of selected features. It is remarkable, that compared to the previous analysis approximately the same number of features (37 versus 32) form the optimal feature sets and nevertheless, the overlap between the two sets is zero. The performance of the validation on NEJM176 (after training on Transbig) in the previous test setting compared to the cross-validation performance on the same set here, is 12% higher in specificity.

For the feature selection via the (modified) AdaBoost algorithm, this difference is even more pronounced: The optimal feature set (of 124 features) selected

[16] When forcing the increase of the number of iterations in the Lars algorithm, we encountered numerical instabilities during matrix inversions.

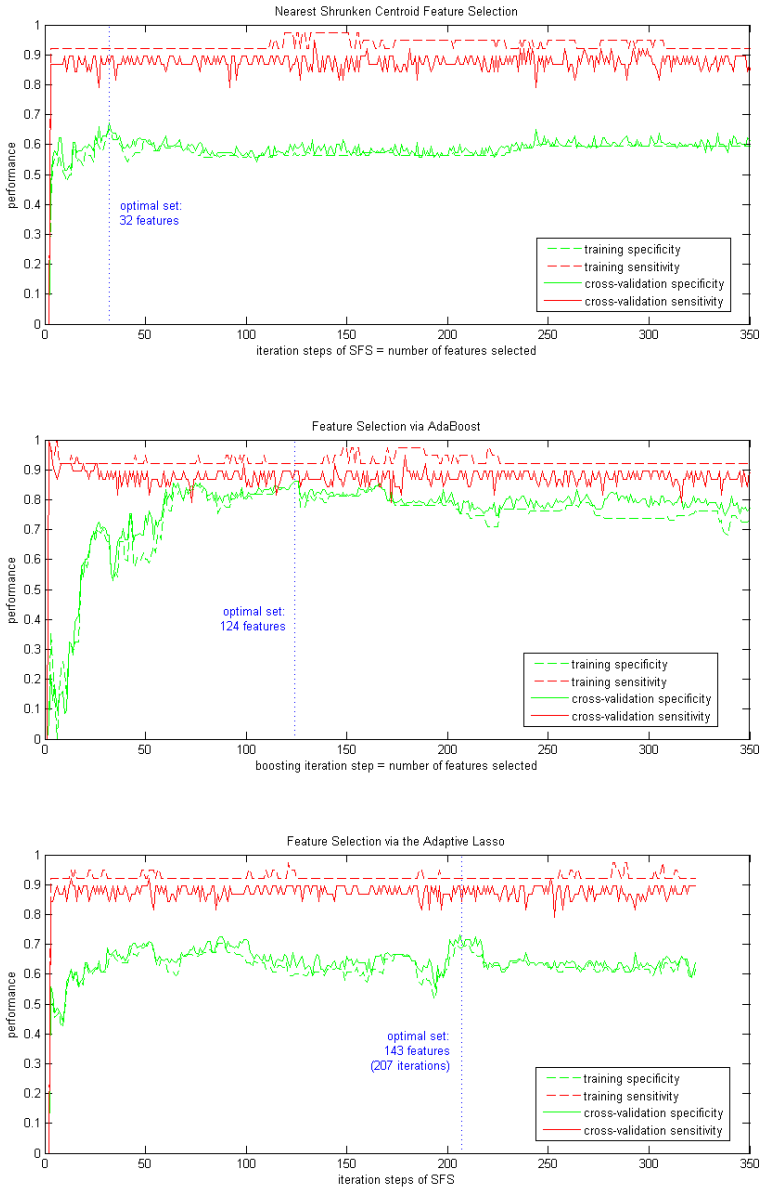



Figure 5.7: Training and validation performance of bNC depending on the iteration steps for the NSC, AdaBoost and the adaptive Lasso feature selection methods, respectively. Optimal validation performance is obtained for sets of 32 (NSC), 134 (AdaBoost) and 143 (Lasso) features, respectively.

via cross-validation showed 86% specificity, whereas previously the (independent) validation specificity on NEJM176 was only 53% (33% less). Interestingly, here we cannot see a clear 'peak' in the cross-validation performance curve, but rather an interval between iteration steps 70 and 170, where specificity is constantly very high and thereafter decreases (until the maximum iteration step 2577 as can be seen in figure D.2 in the appendix).

Again the feature selection via the adaptive Lasso, selects for the largest optimal features set (of the three methods compared here) of 143 features, where specificity is found to be 73%. Generally, the performance curve fluctuates more than for the other two feature selection methods, which might be due to the fact, that the number of features does not increase with each iteration step. However, after the specificity reaches its optimum, the specificity curve seems to stabilize around 65%.

| classifier | NSC | AdaBoost | Lasso |
|---|---|---|---|
| no of features | 32 | 124 | 143 |
| gap value | 0.2042 | 0.0034 | 0.0120 |
| training sensitivity | 0.9211 | 0.9211 | 0.9211 |
| training specificity | 0.6522 | 0.8623 | 0.6884 |
| cross-validation sensitivity | 0.8684 | 0.8947 | 0.8684 |
| cross-validation specificity | **0.6739** | **0.8623** | **0.7319** |

Table 5.3: Optimal number of features as well as optimal gap parameter and training and validation performance of bNC2 on the feature sets determined by different feature selection methods applied to the set of 2577 genes.

Table 5.3 summarizes the performance measurements of the bNC2 classifiers on the optimal feature sets obtained by the respective feature selection method. Training and cross-validation performance on NEJM176 as well as the optimal gap parameter values (of the training) are shown.

The superiority of bNC2 on the feature set selected by the AdaBoost feature selection algorithm is remarkable. Here, both sensitivity and specificity are found to lie between 85% and 90% for the cross-validation measurement. This also explains, why the optimal gap value in this case is close to 0: if the training performance can achieve an accuracy of around 90% for both patient groups, the constraint sensitivity $\geq 0.9$ causes only a small shift of the gap parameter value. If on the other hand sensitivity and specificity range around for example 70% for the classical nearest centroid classification (gap = 0), then the gap parameter value has to be increased, i.e., more samples should be classified high risk, in order to achieve a higher sensitivity (at the expense of decreasing specificity).

Table 5.6 shows that also for this analysis, the overlap of the different optimal feature sets is very low. This also holds true for the comparison of the two feature sets determined by the same feature selection method, but on different basic sets of 231 and 2577 features, respectively. The only exception is the

| | | on 2577 features | | |
|---|---|---|---|---|
| | | NSC | AdaBoost | Lasso |
| on 2577 features | NSC | 32 | 3 (1.54) | 2 (1.78) |
| | AdaBoost | | 124 | 56 (6.88) |
| | Lasso | | | 143 |
| on 231 features | all | 2 (2.87) | 7 (11.12) | 13 (12.82) |
| | NSC | 0 (0.46) | - | - |
| | AdaBoost | | 4 (6.21) | - |
| | Lasso | | | 3 (7.38) |

Table 5.4: The observed overlap of feature sets selected by the different feature selection methods on the base set of 2577 features. The expected overlap according to a hypergeometric distribution is given in parenthesis.

overlap between the feature sets selected by AdaBoost and Lasso. They show an overlap of 56 genes, where only 6.88 would be expected by randomly choosing features (standard deviation is 2.49). It is especially remarkable that most of the overlapping features are selection in the beginning of the AdaBoost iteration steps, e.g. 19 of the first 30 features are also selected by the adaptive Lasso and in particular all 56 overlapping genes are chosen within the first 111 boosting iteration steps.

Consider now the overlap of the genes selected by the feature selection procedures (on the larger base set of 2577 genes) to the set of 231 genes, that were considered as a base set before. Here, the number of overlaps is also not very different form what would be expected by change. This again confirms our assumption that this preselected set is not suitable for performing meaningful feature selection. Furthermore, it offers an explanation for the huge difference in performance of the AdaBoost feature selection: Although it showed the worst results before, it now clearly outperforms the other two feature selection procedures selecting only 7 of the 231 genes.

These results hint at the conclusion that the AdaBoost and adaptive Lasso feature selection methods might be superior to NSC and truly select genes that have high predictive values (alone and especially in conjunction with the rest of the selected features). The characteristics of the different feature selection methods support this assumption: NSC does not evaluate the predictive power of the conjunction of features, but chooses those that are farthest from the overall centroid. This method might also suffer from the unbalanced data sets we used: The low risk group is considerably bigger than the high risk group and thus the former influences the elements of the overall centroid more than the latter. AdaBoost and the adaptive Lasso on the other hand are able to adapt the selection process to the intermediate evaluation of the set of selected features. The adaptive Lasso tries to minimizes a regression function and implicitly adds or deletes a feature of the set of currently selected features by modifying the value of $\lambda$. The AdaBoost feature selection algorithm even incorporates the classification result of the learning classifier (bNC or bNC2, respectively) on the set of selected features for the selection of the next feature. Since (repeat-

edly) misclassified samples are weighted higher than correctly classified ones, the algorithm will eventually select those features that support the correct classification of those repeatedly misclassified outlier samples. Thereby the predictive power of the set of selected features is increased probably more than by adding a feature that has only a high predictive value on its own.

## 5.7 Conclusion

In this chapter we focused on the analysis and comparison of different feature selection methods: the nearest shrunken centroid (NSC) [56], the AdaBoost algorithm for feature selection [14], the adaptive Lasso [65] and the adaptive Lasso for Cox's model [64]. This has been done by investigating the influence of these feature selection methods on the nearest centroid classifier bNC, which showed to be the best performing classifier in the analysis of chapter 4.

Most data sets that are available to us, contain only measurements of a small subset of 231 genes, that were selected during the development of MammaPrint [60]. We therefore decided to conduct a first analysis of feature selection methods taking those 231 genes as a basis set to select from. For performance measurement, we decided to use Transbig as a training and NEJM176 as a validation data set and apply the same quality criteria as before (defined in equations 4.1 and 4.2). Since we want to compare the feature selection methods by means of their influence on the performance of bNC, we defined a simply sequential forward selection algorithm (SFS). It takes as input a list of subsets of genes generated by a feature selection procedure and evaluates bNC's performance for each of those subsets on NEJM176 after training on Transbig. The one with best validation performance, is considered optimal. NSC and the two Lasso feature selection approaches are used in combination with SFS, whereas the AdaBoost feature selection works as a stand-alone method, that uses bNC as a so called learning classifier.

The resulting nearest centroid classifiers (defined on the respective selected feature sets) are then compared against each other and against the classifier defined on the 70 genes used in chapter 4. This is done by measuring their performance on a third, independent data set (LNpos), that is not used during the learning procedure. All feature selection methods provide bNC with a feature set, that leads to an increase in specificity by 6 - 22% compared to the original set of 70 genes. However, only minor differences between the performances on the respective sets of selected features can be observed. Surprisingly, the whole set of 231 genes provides bNC with a feature set exactly as good as the one selected by NSC and even better than the one selected by AdaBoost. The best feature set in this analysis consists of 229 genes (nearly all of the 231), that are selected by the adaptive Lasso for Cox's model.

In order to get an idea of how those feature selection methods influence nearest centroid classification when considering a larger base set of features, we conducted a second analysis on an enlarged set of 2577 genes that are used

for various cancer tests. The only suitable data set for this set of genes is NEJM176 and we therefore defined a new classifier bNC2, which coincides with bNC, except for the data that is used for centroid construction, because bNC2's centroids are obtained from NEJM176 instead of Transbig. We conduct a similar analysis as before for three of the four feature selection methods (the adaptive Lasso for Cox's model had to be disregarded due to numerical problems). This time, we did not split the already small learning set NEJM176 into two parts for training and validation, but used 5-fold cross-validation instead.

For this analysis bNC2's performance is considerably higher than bNC's in the previous one with only 231 genes as a base set.[17] This is particularly pronounced for the AdaBoost feature selection method, which showed the best cross-validation performance with sensitivity 89% and specificity 86%. This result looks very promising especially since we can see a whole interval for feature set sizes between 70 and 170 elements, that show similar performance. However, we would like to point out, that in this analysis we could not test the adaptive Lasso for Cox's model, which turned out to be the best method in the previous analysis and might therefore outperform AdaBoost even for the second test setting. Furthermore, the cross-validation results might be a bit over-optimistic, since the profiles are constructed on the whole data set and hence the left-out samples used for validation are always part of the classifier's centroid. Therefore, a second independent validation would be needed to confirm the results. Nevertheless, the cross-validation performance for the AdaBoost feature selection is very impressive and far from our expectations.

[17] Please note, that a direct comparison of the performance measurements cannot be done, since cross-validation is a less powerful estimate of the classification accuracy than independent validation.

# Chapter 6

# Conclusion and Future Work

In this study we analyzed classification and feature selection methods for the prediction of breast cancer tumor outcome on the basis of microarray gene expression data. More specifically, we considered the problem of predicting the development of distant metastases within a timespan of 5 years. This problem is translated to a binary classification problem with two possible outcomes: $+1$ denoting distant metastasis development and $-1$ denoting the event of remaining distant metastasis free. In contrast to classical analyses, we considered the two kinds of classification errors (type I and type II, see section 2.3) separately. This has been done by restricting ourselves to accept a type I error rate, the rate of misclassification of patients that develop metastases, of maximal 10% during classifier training.

The microarray technology offers a fast and efficient way for the extraction of thousands of gene expressions in parallel [30]. However, since it has only been available since the 1990s, usually for classification one is confronted with the problem of having only very little sample data compared to a vast amount of gene expression data. This can often be the cause for overfitting or reporting overoptimistic results [50]. We tried to avoid this problem by making use of various data sets, that in total comprise around 1000 samples. Two data sets (NEJM176 and Transbig of 176 and 283 samples, respectively) were chosen as a learning set and a third data set (LNpos of 226 samples) for the comparison of classification results.

After reviewing some general classification methods, to our belief the most suitable one for this analysis is the nearest centroid classifier, which has been used in other studies in the literature [34, 59, 60]. Simple classification methods like this have been shown to give good empirical results also compared to more sophisticated ones [29]. Furthermore, it is easy to adapt the constraint of restricting the type II error to 10% during training. The MammaPrint test [60] is such a nearest centroid classifier incorporating this constraint. We tested different nearest centroid classifiers, where we observed big differences in performance and were able to extract the best performing classifier on which we later tested different feature selection methods.

But first, we focused on the nearest centroid classification method itself based on a set of 70 genes, which is also used for MammaPrint. We investigated the influence of three dimensions on the nearest centroid classifier:[1]

- The data used for centroid construction
- The classification rule applied (based on one or two centroids)
- The similarity measure used

In this analysis it turned out that indeed there are huge differences in classification performance depending on how each of the dimensions is defined. For example, we showed, that the 'gap'-classification rule using two centroids is in general superior to classification on only one centroid. The best performing classifier (on the learning set ) is defined by 'Transbig' centroids, using a 'gap'-based classification rule and the Pearson correlation coefficient as similarity measure.

Next, we investigated how different feature selection methods influence the performance of this classifier, which showed that not only the characteristics of the nearest centroid classifier itself, but also the method of selecting features, strongly influences classification accuracy. For this analysis we considered two different sets (consisting of 231 and 2577 genes, respectively) as a base sets for the feature selection. The first one has been chosen since, for the second one, only one data set of 176 samples was available that contains measurements for all 2577 genes. Results on the smaller base set of 231 show some, but little differences between the tested feature selection methods, and did not improve the performance on the full set of 231 genes much. For the analysis on the bigger set, however, we could see huge differences between the methods and surprisingly good classification results especially for the AdaBoost algorithm, which showed to be best for this test set.

On the small feature set of 231 genes we compared four feature selection methods against each other: The nearest shrunken centroid (NSC) [56], the AdaBoost algorithm for feature selection [14], the adaptive Lasso [65] and the adaptive Lasso for Cox's model [64]. Since we want to evaluate their suitability for improving classification, the comparison is done via measuring performance of the nearest centroid classifier determined before on the feature sets selected by these methods. For the AdaBoost feature selection algorithm, this classifier can directly be incorporated in the selection procedure. In order to achieve the same for the other methods and thereby controlling the type II error, we defined a simple sequential forward selection algorithm (SFS). It determines the optimal subset of genes (of a list of subsets defined by the feature selection methods) by choosing the one with optimal validation performance. Thereafter, we compared the nearest centroid classification on those selected subsets by measuring classification performance on an independent data set (not used during the learning procedure).

For the analysis based on the set of 231 genes, it turned out, that the performance of the classification on the 70 genes can be improved by 6 - 22%,

[1] See chapter 4 for a detailed description.

but compared to the performance on the whole set of 231 the increase is less. The classification on the set of features selected by AdaBoost and NSC is even worse or only equally good, respectively, than classification on the whole set. The best feature selection method according to this test setting is the adaptive Lasso for Cox's model, that showed 60% validation specificity (and 97% validation sensitivity).

Another interesting observation is, that the number of overlapping genes between the feature sets selected by the different methods is not very different from the overlap that can be expected by randomly choosing features according to a hypergeometric distribution. This might hint at the fact, that indeed as stated in [21], there is no unique set of features that is suitable for disease classification in this context. On the other hand, considering that also the differences between the feature selection methods are only minor, we do not want to draw any general conclusions since the base set of 231 might be to small.

Therefore, we wanted to repeat the aforementioned analysis on the larger set of 2577 genes being used for various cancer tests. Unfortunately, of the three data sets used before, only one provides measurements for this set of 2577. Hence, we decided to substitute the training/validation measurement for the feature subset selection by 5-fold cross-validation, which allows us to work with one data set only. Since we encountered numerical problems with the LassoCox algorithm for this huge data input, we only compared the NSC, AdaBoost and Lasso methods using the enlarged base set of 2577 features. The cross-validation performances are considerably higher than the validation performances in the previous analysis. The best feature set (selected by AdaBoost) even achieved up to 86% specificity (and 89% sensitivity) for cross-validation.

When taking a look at the overlap between the feature sets selected by these methods, we observed, that there is a large overlap of features selected by AdaBoost and Lasso (56 overlapping features compared to an expected number of only 7 when drawing randomly from a hypergeometric distribution). Other overlaps – also to the set of 231 genes and the corresponding features selected on its basis – are very low. Hence, it seems that AdaBoost and Lasso need a larger, less preselected set of features they can select from in order to effectively contribute to classification improvement. The big overlap between feature sets selected by these two methods might hint at their superiority compared to NSC, in that they truly select for genes with high predictive values (also in conjunction with other features) and therefore, many of those features are selected by both methods. The characteristics of these algorithms in contrast to those of NSC support this idea: while NSC only selects features that differ most in both outcome groups, Lasso and AdaBoost, evaluate the power of the group of selected genes and not the single genes.

These hypotheses should of course in a next step (if possible) be validated by using a second independent data set to confirm the superiority of nearest centroid classification on the sets of selected features by Lasso and especially AdaBoost.

An alternative way of measuring classification performance, which could

be used to support the given results, is to conduct a Kaplan-Meier survival analysis [31]. This method is based on the proportional hazards or Cox model (as defined for the adaptive Lasso for Cox's model). It can be used to determine the differences in the probability of developing distant metastases for the two classification outcome groups. This is particularly interesting from a clinical point of view, since one can then state that for a positive test outcome the probability for metastasis development is $x$ times higher than for a negative outcome. We performed this analysis for the best nearest centroid classifier defined in chapter 4 on the Raster and LNpos data (results not shown) and were able to confirm, that indeed the group that has been classified positive (+1) has a significantly higher risk of developing metastasis than the group that has been classified negative (-1).

This method also provides the advantage of incorporating continuos survival data, i.e., we do not need to choose an arbitrary threshold of, e.g., 5 years that defines the separation of the true outcome classes. This class labeling reduces precision of the true outcome and can be a source of errors in class labels [29]. For classification methods, however it is indispensable to define a finite number of class labels. One possibility of addressing this problem is considering regression instead of classification [25].

This or other more sophisticated classification or feature selection methods might in the future be able to even further improve the prediction of distant metastasis development for breast cancer patients by analyzing gene expression data. For now, we are quite confident that nearest centroid classification – if the classifier is defined appropriately – is a well suited tool to address this complex classification task. We have shown that by combining it with a suitable feature selection method such as the AdaBoost feature selection it can become even more powerful.

# Bibliography


[1] Agendia Technologies Inc. DNA microarrays. [online] `http://www.agilent.com/chem/DNA`.

[2] Agilent Technologies Inc. *Agilent Feature Extraction Software (v10.7) User Guide*. Available as `http://www.chem.agilent.com/Library/usermanuals/Public/G4460-90025_FE_User.pdf`.

[3] Francis R. Bach, David Heckerman, and Eric Horvitz. Considering cost asymmetry in learning classifiers. *J. Mach. Learn. Res.*, 7:1713–1741, 2006.

[4] Thomas Baier and Erich Neuwirth. Excel :: Com :: R. *Computational Statistics*, 22(1):91–108, April 2007.

[5] Amir Ben-Dor, Laurakay Bruhn, Nir Friedman, Iftach Nachman, Michèl Schummer, and Zohar Yakhini. Tissue classification with gene expression profiles. In *RECOMB '00: Proceedings of the fourth annual international conference on Computational molecular biology*, pages 54–64, New York, NY, USA, 2000. ACM.

[6] Ulisses M. Braga-Neto and Edward R. Dougherty. Is cross-validation valid for small-sample microarray classification? *Bioinformatics*, 20(3):374–380, 2004.

[7] Peter Bühlmann and Torsten Hothorn. Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, 22(4):477–505, 2007.

[8] Peter Bühlmann and Torsten Hothorn. Rejoinder: Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, 22(4):516–522, 2007.

[9] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121 – 168, 1998.

[10] Marc Buyes et al. Validation and clinical utility of a 70-gene prognostic signature for women with node-negative breast cancer. *Journal - national cancer institute*, 98(17):1183–1192, September 2006.

[11] William S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the american statistical association*, 74(368):829–836, December 1979.

[12] David Roxbee Cox. Regression models and life-tables. *Journal of the royal statistical society*, 34:187 – 220, March 1972.

[13] Hongyue Dai, Michael Meyer, Sergey Stepaniants, Michael Ziman, and Roland Stoughton. Use of hybridization kinetics for differentiating specific from non-specific binding to oligonucletide microarrays. *Nucleic Acids Research*, 30(16), 2002.

[14] Sanmay Das. Filters, wrappers and a boosting-based hybrid for feature selection. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 74–81, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[15] Chad A. Davis, Fabian Gerick, Volker Hintermair, Caroline C. Friedel, Katrin Fundel, Robert Küffner, and Ralf Zimmer. Reliable gene signatures for microarray classification: assessment of stability and performance. *Bioinformatics*, 22(19):2356–2363, 2006.

[16] Sandrine Dudoit, Jane Fridlyand, and Terence P. Speed. Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association*, 97(457):77–87, March 2002.

[17] Sandrine Dudoit, Yee Hwa Yang, Matthew J. Collow, and Terence P. Speed. Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statistica sinica*, 12(1):111–140, 2002.

[18] Robert P. W. Duin. A note on comparing classifiers. *Pattern Recognition Letters*, 17(5):529–536, 1996.

[19] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *ANNALS OF STATISTICS*, 32:407, 2004.

[20] P. Eifel et al. National institutes of health consensus development conference statement: adjuvant therapy for breast cancer, november 1-3, 2000. *Journal of the national cancer institute*, 93(13):979–989, July 2001.

[21] Liat Ein-Dor, Itai Kela, Gad Getz, David Givol, and Eytan Domany. Outcome signature genes in breast cancer: is there a unique set? *Bioinformatics*, 21(2):171–178, January 2005.

[22] Tom Fawcett. An introduction to ROC analysis. *Pattern Recogn. Lett.*, 27(8):861–874, 2006.

[23] J Ferlay, P Autier, M Boniol, M Heanue, M Colombet, and P Boyle. Estimates of the cancer incidence and mortality in Europe in 2006. *Annals of Oncology*, 18(3):581–592, 2007.

[24] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *EuroCOLT '95: Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37, London, UK, 1995. Springer-Verlag.

[25] Jerome H. Friedman. Comment: Classifier technology and the illusion of progress. *Statistical Science*, 21:1, 2006.

[26] Annuska Glas, Arno Floore, Leonie Delahaye, Anke Witteveen, Rob Pover, Niels Bakx, Jaana Lahti-Domenici, Tako Bruinsma, Marc Warmoes, Rene Bernards, Lodewyk Wessels, and Laura Van 't Veer. Converting a breast cancer microarray signature into a high-throughput diagnostic test. *BMC Genomics*, 7(1):278, October 2006.

[27] Aron Goldhirsch, John H. Glick, Richard D. Gelber, and Hans-Jörg Senn. Meeting highlights: international consensus panel on the treatment of primary breast cancer. *Journal of the national cancer institute*, 90(21):1601–1608, 1998.

[28] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3:1157–1182, 2003.

[29] David J. Hand. Classifier technology and the illusion of progress. *Science*, 21:1, 2006.

[30] Gary Hardiman. Microarray platforms - comparisons and contrasts. *Pharmacogenomics*, 5:487–502, 2004.

[31] E. L. Kaplan and Paul Meier. Nonparametric estimation from incomplete observations. *Journal of the american statistical association*, 53(282):457 – 481, June 1958.

[32] Kyung-Joong Kim and Sung-Bae Cho. Ensemble classifiers based on correlation analysis for DNA microarray classification. *Neurocomputing*, 70:187–199, December 2006.

[33] Ron Kohavi. A study of cross-validation and boostrap of accuracy estimation and model selection. *International joint conference on artificial intelligence*, 14(2):1137–1145, 1995.

[34] Ilya Levner. Feature selection and nearest centroid classification for protein mass spectrometry. *BMC Bioinformatics*, 6(1):68, 2005.

[35] Sofus Macskassy and Foster Provost. Confidence bands for ROC curves: Methods and an empirical study. *CeDER working papers*, August 2004.

[36] Dragos D. Margineantu and Thomas G. Dietterich. Bootstrap methods for the cost-sensitive evaluation of classifiers. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 583–590, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[37] Dragos D. Margineantu and Thomas G. Dietterich. Bootstrap methods for the cost-sensitive evaluation of classifiers. In *In Proc. 17th International Conf. on Machine Learning*, pages 583–590. Morgan Kaufmann, 2000.

[38] David Mease and Abraham Wyner. Evidence contrary to the statistical view of boosting. *J. Mach. Learn. Res.*, 9:131–156, 2008.

[39] Ting Lee Mei-Ling. *Analysis of microarray gene expression data*. Springer, 2004.

[40] Ron Meir and Gunnar Rätsch. An introduction to boosting and leveraging. In *Advanced lectures on machine learning*, pages 118–183, New York, NY, USA, 2003. Springer-Verlag New York, Inc.

[41] Stefan Michiels, Serge Koscielny, and Catherine Hill. Prediction of cancer outcome with microarrays: a multiple random validation strategy. *Lancet*, 365:488–492, February 2005.

[42] National cancer institute. Metastatic cancer: questions and answers. [online] `http://www.cancer.gov/cancertopics/factsheet/Sites-Types/metastatic`.

[43] Andrew Y. Ng. Preventing 'overfitting' in cross-validation data. *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 245–253, 1997.

[44] N. Perry, M. Broeders, C. de Wolf, S. Törnberg, R. Holland, and L. von Karsa. European guidelines for quality assurance in breast cancer screening and diagnosis. fourth edition – summary document. *Annals of Oncology*, 19(4):614–622, 2008.

[45] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[46] Laura Elena Raileanu and Kilian Stoffel. Theoretical comparison between the gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence*, 41(1):77–93, 2004.

[47] Peter M. Ravdin et al. Computer program to assist in making decisions about adjuvant therapy for women with early breast cancer, February 2001.

[48] Robert E. Schapire. A brief introduction to boosting. In *IJCAI'99: Proceedings of the 16th international joint conference on Artificial intelligence*, pages 1401–1406, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[49] Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. In *COLT' 98: Proceedings of the eleventh annual conference on Computational learning theory*, pages 80–91, New York, NY, USA, 1998. ACM.

[50] Richard Simon, Michael D. Radmacher, Kevin Dobbin, and Lisa M. McShane. Pitfalls in the use of DNA microarray data for diagnostic and prognostic classification. *Journal of the National Cancer Institute*, 95(1):14–18, January 2003.

[51] K. Sjöstrand. Matlab implementation of LASSO, LARS, the elastic net and SPCA, jun 2005. Version 2.0.

[52] Terry P. Speed. *Statistical analysis of gene expression microarray data*. Interdisciplinary statistics. Chapman and Hall/CRC, 2003.

[53] The MathWorks Inc. Matlab. Version V 7.9 (2009b).

[54] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B*, 58(1):267–288, 1996.

[55] Robert Tibshirani. The lasso method for variable selection in the cox model. *Statistics in medicine*, 16(4):385–395, 1997.

[56] Robert Tibshirani, Trevor Hastie, Balasubramanian Narasimhan, and Gilbert Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *PNAS*, 99(10):6567–6572, May 2002.

[57] Robert Tibshirani, Trevor Hastie, Balasubramanian Narasimhan, and Gilbert Chu. Class prediction by nearest shrunken centroids, with applications to DNA microarrays. *Statistical science*, 18(1):104–117, 2003.

[58] Anna V. Tinker, Alex Boussioutas, and David D.L. Bowtell. The challenges of gene expression microarrays for the study of human cancer. *Cancer Cell*, 9(5):333–339, May 2006.

[59] Marc J. van de Vijver et al. A gene-expression signature as a predictor of survival in breast cancer. *The New England Journal of Medicine*, 347(25):1999–2009, December 2002.

[60] Laura J. van't Veer et al. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415:530–536, January 2002.

[61] K. Veropoulos, C. Campbell, and N. Cristianini. Controlling the sensitivity of support vector machines. In *Proceedings of the International Joint Conference on AI*, pages 55–60, 1999.

[62] Sijian Wang and Ji Zhu. Improved centroids estimation for the nearest shrunken centroid classifier. *Bioinformatics*, 23(8):972–979, 2007.

[63] Lee Weng, Hongyue Dai, Yihui Zhan, Yudong He, Sergey B. Stepaniants, and Douglas E. Bassett. Rosetta error model for gene expression analysis. *Bioinformatics*, 22(9):1111–1121, 2006.

[64] Hao Helen Zhang and Wenbin Lu. Adaptive lasso for cox's proportional hazards model. *Biometrika*, 94(3):691–703, 2007.

[65] Hui Zou. The adaptive alsso and its oracle properties. *Journal of the american statistical association*, 101(476):1418–1429, December 2006.

[66] Hui Zou. A note on path-based variable selection in the penalized proportional hazards model. *Biometrika*, 95(1):241–247, 2008.

# Appendix A

# Abstract

## A.1 Abstract (English)

Cancer and especially breast cancer has become a more and more frequent disease within the last few years. In the clinical context, tumors are subdivided into different categories to provide a more individual treatment. One important aspect that is used for this categorization is the development of metastases distant from the primary tumor. In this thesis, we aim at predicting this outcome by classifying the tumor's gene expression data. This data is obtained from microarrays, which is a technology providing a fast and efficient way of extracting gene expressions, thereby enabling such classification.

The binary classification problem studied here is to decide whether a tumor will develop distant metastases or not. In contrast to classical studies in this field, we are not interested in maximizing overall classification performance, but focus on keeping the type II error (misclassification of metastases developing patients) low and only in the second place minimize the type I error.

We define different nearest centroid classifiers, where the centroids are given by gene expression profiles consisting of average gene expression values for each outcome group. We then compare their performance and analyze the influence of feature selection methods on classification accuracy.

We show that the performance of nearest centroid classification varies a lot depending on the specific definition of the classifier. Furthermore, we demonstrate that the feature set, on which the classification is based, has a big influence on the classifier's accuracy and choosing an appropriate feature selection method can therefore lead to a huge improvement in performance. The best classification result can be observed when combining a specific nearest centroid classifier with an AdaBoost feature selection algorithm: 5-fold cross-validation showed 89% sensitivity and 86% specificity.

## A.2 Abstract (German)

Krebs und besonders Brustkrebs ist in den letzten Jahre zu einer mehr und mehr verbreitete Krankheit geworden. Im klinischen Kontext werden Tumore in verschieden Kategorien unterteilt, um eine individuellere Behandlung zu ermöglichen. Ein wichtiger Aspekt zur Bestimmung dieser Kategorien ist die Entwicklung von Metastasen, die sich (weit) entfernt vom Primärtumor bilden. Das Ziel dieser Arbeit ist die Vorhersage dieses Krankheitsbildes durch Klassifikation der Genexpressionsdaten des Tumors. Die dafür benötigten Daten werden mit Hilfe von Mikroarrays gewonnen, einer Technologie die es erlaubt Genexpressionsdaten schnell und effiziente zu extrahiert und dadurch eine solche Klassifikation ermöglicht.

Wir untersuchen hier das binäre Klassifikationsproblem der Bestimmung ob ein Tumor entfernte Metastasen bilden wird oder nicht. Im Gegensatz zu klassischen Studien in diesem Bereich wollen wir nicht die globale Klassifikationsgüte maximieren, sondern versuchen den Fehler zweiter Art (Fehlklassifikation eines Metastasen entwickelnden Patienten) niedrig zu halten und erst an zweiter Stelle den Fehler erster Art zu minimieren.

Wir definieren verschiedene *nearest centroid* Klassifikatoren, wobei die centroids so genannte "Genexpressionsprofile" sind, die aus den durchschnittlichen Genexpressionswerten von Patient jedes Krankheitsbildes bestehen. Danach vergleichen wir die Klassifikationsgüte dieser Klassifikatoren und analysieren, welchen Einfluss Featureselektionsmethoden darauf haben.

Es wir gezeigt, dass die Güte der nearest centroid Klassifikation stark von der genauen Definition des Klassifikatores abhängt. Des weiteren zeigen wir, dass die Featuremenge, auf welcher die Klassifikation basiert, einen großen Einfluss auf die Genauigkeit des Klassifikators hat und durch die Wahl einer geeigneten Featureselektionsmethode daher desssen Güte erheblich verbessert werden kann. Das beste Klassifikationsergebnis wird erreicht durch die Kombination eines bestimmten nearest centroid Klassifikatores mit einem AdaBoost-Featureselektionsalgorithmus: Eine 5-fache Kreuzvalidierung erreicht 89% Sensitivität (*sensitivity*) und 89% Spezifität (*specificity*).

# Appendix B

# Definition of Statistical Measures

**Definition 8** (Pearson correlation)
*The* Pearson correlation coefficient *or* correlation coefficient *of two random variables $X$ and $Y$ is given by*

$$corr(X,Y) = \frac{E(X-\mu_X)\cdot E(Y-\mu_Y)}{\sigma_X \sigma_Y}$$

*and*

$$corr(x,y) = \frac{\sum (x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum (x_i-\bar{x})^2}\sqrt{\sum (y_i-\bar{y})^2}}$$

*for the sample correlation coefficient respectively.*

**Definition 9** (cosine correlation)
*The* cosine correlation coefficient *of two vectors $x, y \in \mathbb{R}$ is defined as the cosine of the ankle $\alpha$ enclosed by the vectors:*

$$cosinecorrelation(x,y) = cos(\alpha) = \frac{x\cdot y}{||x||||y||}.$$

**Definition 10** (Spearman correlation)
*The* Spearman correlation coefficient *or* Spearman rank correlation coefficient *of two sample vectors $x$ and $y$ is given by*

$$r(x,y) = 1 - 6\frac{\sum {d_i}^2}{n(n^2-1)}$$

*where $d_i$ is the difference of the statistical ranks of $x_i$ and $y_i$ and $n$ is the number of samples in the data set.*

**Definition 11** (Kendall correlation)
*The* Kendall correlation coefficient *or* Kendall $\tau$ coefficient *of two sample vectors* $x$ *and* $y$ *is given by*

$$\tau(x, y) = 2\frac{n_c - n_d}{n(n-1)}$$

*where* $n_c$ *and* $n_d$ *is the number of* concurdant *and* discordant *pairs resp., i.e., pairs* $\{x_i, y_i\}$ *and* $\{x_j, y_j\}$ *that fullfill*

$$sgn(x_j - x_i) = sgn(y_j - y_i)$$

*and*

$$sgn(x_j - x_i) = -sgn(y_j - y_i)$$

*respectively and* $n$ *is the total number of samples in the data set.*

# Appendix C

# Data Sets

| Data set | Size | 70 genes | 231 genes | Full genome | Microarray |
|---|---|---|---|---|---|
| Nature | 78 | X | X | X | Hu25K |
| NEJM | 295 | X | X | X | Hu25K |
| NEJM260 | 260 | X | X | X | HD44k |
| Transbig | 324 | X | X | | LD8pack |
| LNpos | 241 | X | X | | LD8pack |
| Raster | 427 | X | X | (100) | LD8pack/44K(100) |

Table C.1: Available data sets

| Data set | Size | Selected samples | Positive samples | Negative samples |
|---|---|---|---|---|
| Nature | 78 | 78 | 44 | 34 |
| NEJM176 | 199 | 176 | 38 | 138 |
| Transbig | 324 | 283 | 48 | 235 |
| LNpos | 241 | 226 | 32 | 194 |
| Raster | 427 | 0 | - | - |

Table C.2: Samples selected form the data sets that are used for classification and the amount of low risk (negative) and high risk (positive) samples thereof.

Only those samples have been used that had a follow-up time of at least five years with respect to distance metastasis development (otherwise class labels cannot be defined). For the Nature data set, we used all samples that had been selected previously in [60] by a slightly different approach. This data set is included in the NEJM data set and NEJM260 comprises samples of NEJM that are reanalyzed on the 44k microarray. For classification NEJM260 is subdivided into two data sets: the samples that coincide with those from Nature (61 samples) and those that do not (199 samples, where 176 with sufficient follow-up form the set NEJM176).

# Appendix D

# Supplementary Information on Classification Results

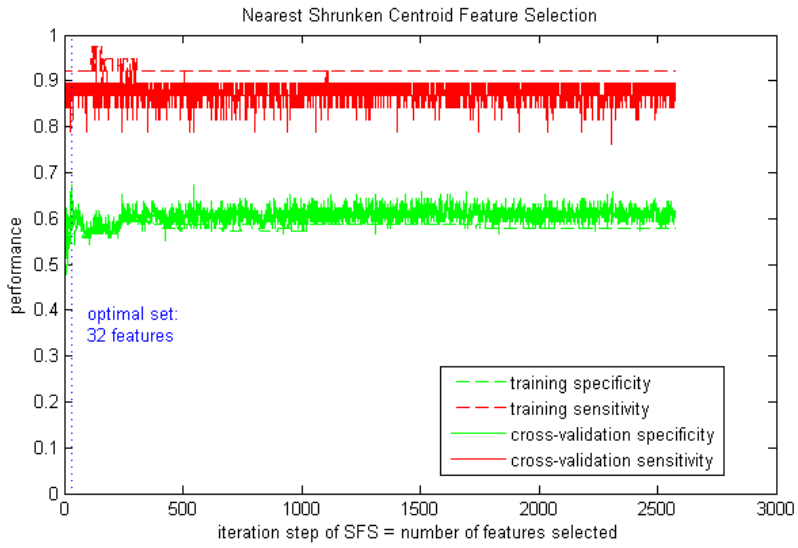



Figure D.1: Training and 5-fold cross-validation performance (on NEJM176) of the nearest centroid classifier bNC2 depending on the number of features selected by the nearest shrunken centroid feature selection algorithm. The set with optimal cross-validation performance contains 32 features.

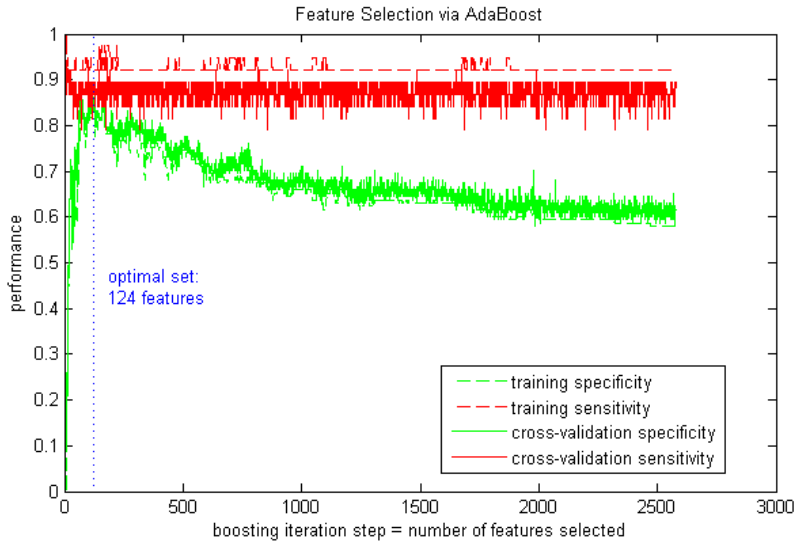



Figure D.2: Training and 5-fold cross-validation performance (on NEJM176) of the nearest centroid classifier bNC2 depending on the number of features selected by the AdaBoost feature selection algorithm. The set with optimal cross-validation performance contains 134 features.

**Optimal training Parameter on NEJM**

| | LR | HR | LRHR |
|---|---|---|---|
| n cosine | 0,5922 | -0,0544 | 0,4501 |
| n Pearson | 0,5336 | -0,0683 | 0,3748 |
| n Spearman | 0,5112 | 0,0658 | 0,4004 |
| n Kendall | 0,3723 | 0,0476 | 0,2849 |
| t cosine | 0,4475 | -0,0622 | 0,2732 |
| t Pearson | 0,4717 | -0,0350 | 0,2086 |
| t Spearman | 0,4812 | -0,1077 | 0,3501 |
| t Kendall | 0,3549 | -0,0758 | 0,2228 |

**Optimal training Parameter on Transbig**

| | LR | HR | LRHR |
|---|---|---|---|
| n cosine | 0,4225 | 0,0825 | 0,3001 |
| n Pearson | 0,4322 | 0,0658 | 0,4267 |
| n Spearman | 0,3832 | 0,0838 | 0,2994 |
| n Kendall | 0,2779 | 0,0559 | 0,2220 |
| t cosine | 0,5216 | 0,1506 | 0,2189 |
| t Pearson | 0,5330 | 0,1476 | 0,2718 |
| t Spearman | 0,5125 | 0,1794 | 0,2899 |
| t Kendall | 0,3640 | 0,1204 | 0,2063 |

**Sensitivity on Transbig validation data**

| | LR | HR | LRHR |
|---|---|---|---|
| n cosine | 0,9792 | 0,9388 | 0,9167 |
| n Pearson | 0,9583 | 0,9583 | 0,8958 |
| n Spearman | 0,9375 | 0,9375 | 0,9167 |
| n Kendall | 0,9375 | 0,9375 | 0,9375 |
| t cosine | 0,6458 | 0,9792 | 0,9167 |
| t Pearson | 0,6458 | 0,9792 | 0,8750 |
| t Spearman | 0,8125 | 1,0000 | 0,9583 |
| t Kendall | 0,8750 | 1,0000 | 0,9583 |

**Sensitivity on NEJM176 validation data**

| | LR | HR | LRHR |
|---|---|---|---|
| n cosine | 0,8158 | 0,8421 | 0,8421 |
| n Pearson | 0,8421 | 0,8684 | 0,9211 |
| n Spearman | 0,8421 | 0,8684 | 0,8684 |
| n Kendall | 0,8684 | 0,8421 | 0,8684 |
| t cosine | 0,9474 | 0,6316 | 0,7895 |
| t Pearson | 0,9474 | 0,6842 | 0,9474 |
| t Spearman | 0,9211 | 0,5263 | 0,8947 |
| t Kendall | 0,9474 | 0,5526 | 0,8947 |

**Specificity on Transbig Validation data**

| | LR | HR | gap |
|---|---|---|---|
| n cosine | 0,1787 | 0,3149 | 0,3787 |
| n Pearson | 0,2340 | 0,2936 | 0,4128 |
| n Spearman | 0,2894 | 0,3915 | 0,3532 |
| n Kendall | 0,2723 | 0,3830 | 0,3532 |
| t cosine | 0,3447 | 0,2681 | 0,4979 |
| t Pearson | 0,3106 | 0,2213 | 0,5234 |
| t Spearman | 0,4000 | 0,0383 | 0,3362 |
| t Kendall | 0,3617 | 0,0383 | 0,3957 |

**Specificity on NEJM176 validation data**

| | LR | HR | gap |
|---|---|---|---|
| n cosine | 0,5942 | 0,4638 | 0,5507 |
| n Pearson | 0,4928 | 0,4203 | 0,4203 |
| n Spearman | 0,5290 | 0,4130 | 0,4420 |
| n Kendall | 0,4855 | 0,4058 | 0,4420 |
| t cosine | 0,0507 | 0,6377 | 0,5725 |
| t Pearson | 0,0652 | 0,5797 | 0,4638 |
| t Spearman | 0,1014 | 0,6087 | 0,4638 |
| t Kendall | 0,1159 | 0,6087 | 0,4348 |

**Sensitivity of LOOCV on Transbig**

| | LR | HR | gap |
|---|---|---|---|
| n cosine | 0,8958 | 0,8958 | 0,8958 |
| n Pearson | 0,8958 | 0,8958 | 0,8958 |
| n Spearman | 0,8958 | 0,8958 | 0,8958 |
| n Kendall | 0,8958 | 0,8958 | 0,8958 |
| t cosine | 0,8958 | 0,8958 | 0,8958 |
| t Pearson | 0,8958 | 0,8958 | 0,8958 |
| t Spearman | 0,8958 | 0,8958 | 0,8958 |
| t Kendall | 0,8958 | 0,8958 | 0,8958 |

**Sensitivity of LOOCV on NEJM176**

| | LR | HR | gap |
|---|---|---|---|
| n cosine | 0,8947 | 0,8947 | 0,8947 |
| n Pearson | 0,8947 | 0,8947 | 0,8947 |
| n Spearman | 0,8947 | 0,8947 | 0,8947 |
| n Kendall | 0,8947 | 0,8947 | 0,8947 |
| t cosine | 0,8947 | 0,8947 | 0,8947 |
| t Pearson | 0,8947 | 0,8947 | 0,8947 |
| t Spearman | 0,8947 | 0,8947 | 0,8947 |
| t Kendall | 0,8947 | 0,8947 | 0,8947 |

**Specificity of LOOCV on Transbig**

| | LR | HR | gap |
|---|---|---|---|
| n cosine | 0,4128 | 0,4766 | 0,4468 |
| n Pearson | 0,3957 | 0,4340 | 0,3745 |
| n Spearman | 0,4340 | 0,4128 | 0,4298 |
| n Kendall | 0,4255 | 0,3915 | 0,4170 |
| t cosine | 0,1617 | 0,4766 | 0,5404 |
| t Pearson | 0,1574 | 0,4340 | 0,4723 |
| t Spearman | 0,3064 | 0,3915 | 0,4383 |
| t Kendall | 0,3362 | 0,3872 | 0,4298 |

**Specificity of LOOCV on NEJM176**

| | LR | HR | gap |
|---|---|---|---|
| n cosine | 0,3696 | 0,3551 | 0,4638 |
| n Pearson | 0,3406 | 0,2826 | 0,4348 |
| n Spearman | 0,3551 | 0,3913 | 0,4130 |
| n Kendall | 0,3261 | 0,3913 | 0,4130 |
| t cosine | 0,1159 | 0,4130 | 0,5290 |
| t Pearson | 0,1087 | 0,3623 | 0,5145 |
| t Spearman | 0,1377 | 0,2754 | 0,3768 |
| t Kendall | 0,1159 | 0,2754 | 0,4275 |

Figure D.3: Results of the comparison of nearest centroid classifiers trained, validated and leave-one-out cross-validated on Transbig and NEJM176. In the first row, for LR and HR the parameter tested is threshold, whereas for LRHR it is gap. Optimal CV parameters coincide with the optimal training parameters.

**Melanie Görner**

Date of birth: 23. November 1984
Nationality: German

Universität Wien
Doktor Karl Lueger Ring 1
1010 Wien, Austria
Telephone: +43 (0)1 42770

## Education

| | |
|---|---|
| **10/2008 – today** | **Master of Science in Mathematics (Universität Wien, Austria), specializing in Biomathematics**<br>• Master's thesis: 'Tumor Classification Based on Gene Expression Profiles'<br>• Expected graduation: July 2010 (current average grade: 'very good' approximately equivalent to an 'A'-grade) |
| **10/2004 – 06/2008** | **Bachelor of Science in ‚Mathematics with Computer Science' (Technische Universität Darmstadt, Germany)**<br>• Study abroad in 2006/2007 (Universidad de Granada, Spain)<br>• Bachelor's thesis: 'Efficient Generation of Random Graphs with Given Degree Sequence via Markov Chains'<br>• Final grade: 'very good' approximately equivalent to an 'A'-grade |
| **03/2004** | **Allgemeinen Hochschulreife (approximately equivalent to A-levels) (Gymnasium Kirn, Germany)**<br>Final grade: 1,6 (approximately equivalent to an overall 'A'-grade) |

## Publications

T. von Landesberger, M. Görner, R. Rehner and T. Schreck. **A System for Interactive Visual Analysis of Large Graphs Using Motifs in Graph Editing and Aggregation.** *Proceedings of the 14th International Fall Workshop on Vision, Modeling, and Visualization (VMV),* 2009.

T. von Landesberger, M. Görner and T. Schreck. **Visual Analysis of Graphs with Multiple Connected Components.** *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST),* 2009.

## Working experience

| | |
|---|---|
| **10/2009 – today** | **Bioinformatician (Internship followed by a regular position since 04/2010) at Agendia BV** (Amsterdam, the Netherlands) in the research and development department<br>• Adjusting and implementing (in Matlab) various classification and feature selection methods for the classification of breast cancer tumors based on gene expression data<br>• Developing and applying a framework for the comparison of these methods<br>• Interpreting and presenting of results in reports, talks and my master's thesis |
| **02/2008 – 09/2008** | **Scientific assistant (part-time) at Fraunhofer Institute for Computer Graphics (IGD)** (Darmstadt, Germany) in the research department A3: 'Realtime Solutions for Simulation and Visual Analytics'<br>• Assisted a project team developing a software tool for graph visualization and graph analysis<br>• Extended this tool by adding features for searching and visualizing 'network-motifs'<br>• Developed and implemented a new graph visualization tool for results obtained by a 'Self-Organizing-Map' (SOM) algorithm |

## Social commitment

| | |
|---|---|
| **02/2009 – 09/2009** | **Cooperation at IAESTE Austria (Technischen Universität Wien, Austria)**<br>• Assisted IAESTE's internship placement service for international students at local companies<br>• Supported the interns prior to and during their stay in Austria<br>• Developed a marketing strategy for the 'Firmenmesse' career fair 2009 |
| **10/2007 & 10/2005** | **Tutoring (Technische Universität Darmstadt, Germany)** of international students during orientation activities |

## Programming skills

| | |
|---|---|
| **Java** | Very good programming skills |
| **Matlab** | Very good programming skills |
| **OpenGL** | Good command (embedded in Java) |

## Language proficiency

| | |
|---|---|
| **German** | Native speaker |
| **Spanish** | Fluency |
| **English** | Fluency |
| **Latin** | ‘Latinum’ proficiency certificate |
| **French** | Working knowledge |
| **Dutch** | Basic knowledge |

**Melanie Görner**

Geburtsdatum: 23. November 1984
Nationalität: Deutsch

Universität Wien
Doktor Karl Lueger Ring 1
1010 Wien, Österreich
Telefon: +43 (0)1 42770

## Ausbildung

| | |
|---|---|
| **10/2008 – heute** | **Masterstudium Mathematik (Universität Wien, Österreich) im Schwerpunkt Biomathematik**<br>• Abschlussarbeit: „Tumor Classification Based on Gene Expression Profiles“<br>• Abschluss voraussichtlich Juli 2010 (aktuelle Durchschnittsnote: sehr gut) |
| **10/2004 – 06/2008** | **Bachelorstudium „Mathematics with Computer Science” (Technische Universität Darmstadt, Deutschland)**<br>• Auslandsstudium im Studienjahr 2006/2007 (Universidad de Granada, Spanien)<br>• Abschlussarbeit: „Effizientes Generieren von Zufallsgraphen mit vorgegebener Gradsequenz über Markov-Ketten“<br>• Abschlussnote: sehr gut |
| **03/2004** | **Allgemeinen Hochschulreife (Gymnasium Kirn, Deutschland)**<br>Abschlussnote: 1,6 (sehr gut) |

## Publikationen

T. von Landesberger, M. Görner, R. Rehner and T. Schreck. **A System for Interactive Visual Analysis of Large Graphs Using Motifs in Graph Editing and Aggregation.** *Proceedings of the 14th International Fall Workshop on Vision, Modeling, and Visualization (VMV),* 2009.

T. von Landesberger, M. Görner and T. Schreck. **Visual Analysis of Graphs with Multiple Connected Components.** *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST),* 2009.

## Berufserfahrung

| | |
|---|---|
| **10/2009 – heute** | **Bioinformatician (Praktikum, danach Festanstellung seit 04/2010) bei Agendia BV** (Amsterdam, Niederlande) im Bereich Forschung und Entwicklung<br>• Anpassung und Implementierung (in Matlab) verschiedener Klassifikations- und Feature Selektions-Methoden auf Basis von Microarray Genexpressionsdaten für die Klassifizierung von Brustkrebs Tumoren<br>• Erstellung und Anwendung eines Frameworks zum Vergleich der Methoden<br>• Interpretation und Präsentation der Ergebnisse in Berichten, Vorträgen und in der darauf aufbauenden Masterarbeit |
| **02/2008 – 09/2008** | **Wissenschaftliche Hilfskraft am Fraunhofer Institute für Graphische Datenverarbeitung (IGD)** (Darmstadt, Deutschland) in der Arbeitsgruppe A3: „Echtzeitlösungen für Simulation und Visual Analytics“<br>• Mitarbeit in einem Projektteam zur Entwicklung eines Software Tools zur Graphvisualisierung und Graphanalyse<br>• Erweiterung des Tools durch Features zur Suche und Visualisierung von Graphmotiven<br>• Entwicklung und Implementierung eines neuen Graphvisualisierungstools für Ergebnisse eines „Self-Organizing-Map“ (SOM) Algorithmus |

## Soziales Engagement

| | |
|---|---|
| **02/2009 – 09/2009** | **Mitarbeit bei IAESTE Austria (Technischen Universität Wien, Österreich)**<br>• Vermittlung von internationalen Praktikanten an lokale Unternehmen<br>• Betreuung der Praktikanten vor und während ihres Aufenthaltes in Österreich<br>• Ausarbeitung des Marketingplans für die „Firmenmesse“ 2009 |
| **10/2007 & 10/2005** | **Tutorentätigkeit (Technische Universität Darmstadt, Deutschland)** für internationale Studenten während Orientierungsveranstaltungen |

## Programmierkenntnisse

| | |
|---|---|
| **Java** | Sehr gute Kenntnisse |
| **Matlab** | Sehr gute Kenntnisse |
| **OpenGL** | Kenntnisse als Einbettung in Java |

## Sprachkenntnisse

| | |
|---|---|
| **Deutsch** | Muttersprache |
| **Spanisch** | Fließend |
| **Englisch** | Fließend |
| **Latein** | Latinum |
| **Französisch** | Erweiterte Grundkenntnisse |
| **Niederländisch** | Grundkenntnisse |

Amsterdam, April 2010