Master Thesis 20p

Analysis of Proteomic Patterns for Detection of Prostate Cancer

author: Anders Lindegren

28th June 2004

Abstract

The SELDI process is a relatively new medical technique that measures the content of different proteins in blood samples from patients. Recently, many research teams have shown that there is a relation between the concentrations of specific proteins and cancer disease. This report has focused on the area of prostate cancer. The output from the SELDI system is created through mass-spectrometry and is a spectrum containing the concentrations of thousands of separate proteins for each sample. The aim of this work has been to use pattern recognition algorithms such as Bayesian Discriminant functions, Fisher Linear Discriminant (FLD) and KNN to separate samples into three classes: healthy patients, cancer patients and patients with benign prostate condition. In addition to theory and application of these algorithms, the report also includes different methods to reduce the high dimensionality of the SELDI output to a more manageable number of variables before the classification step is performed. Principal Component Analysis (PCA) and Discrete Wavelet Transform (DWT) are two described methods with this property. The proposed algorithms have been applied to separate data sets containing samples from prostate patients with promising results. Using PCA in combination with FLD on one of the data sets (containing 197 samples) yielded a sensitivity of 97% , a specificity for the healthy samples of 100% and classified 94% of the samples with benign condition correctly. To achieve as accurate results as possible, jack-knifing was used in these experiments. The report also includes a method called Back-Projection used to identify the most important proteins in the initial SELDI spectra.

Contents

1 Introduction

Prostate cancer is the most common cancer disease among men in Sweden. The most widely used method for cancer detection has up to this point been measuring the concentration of the prostate specific antigen (PSA). PSA is the best marker used in clinical practise. The method has the desirable property of yielding high sensitivity but also the drawback that the specificity is relatively low. This means that most of the patients with cancer will be diagnosed correctly but of several of the non-cancer patients will also be diagnosed to have cancer. Recently, several studies have focused on a relatively new technique as an alternative to PSA that can be used for cancer detection. It is a protein chip system from Ciphergen Biosystems; "Surface Enhanced Laser Desorption/Ionization time of flight" (SELDI) that in combination with mass spectrometry enables rapid identification of differentially expressed or altered proteins.

The input samples to the SELDI technology can be taken from human serum, plasma or tissue. The idea is that the concentration (or intensity) of certain proteins is different in the groups of healthy and cancer samples respectively. This has indeed shown to be the case in the performed studies. The output from the SELDI process applied on one sample is a curve with very high resolution containing the intensity of thousands of separate proteins (Figure 1 shows an example output). This fact makes it difficult and time consuming to process and evaluate the data by hand. This report describes different ways of analysing the output data with pattern recognition algorithms using a computer to separate the different categories of samples, e.g. cancer, healthy and benign prostate condition.

An important part of this problem is how to handle data with the number of variables (the different protein masses) being much greater than the number of available samples. This calls for the use of dimensionality reduction techniques, that must be carried out before the pattern recognition (or *classification*) algorithms can be applied on the data. The report contains the theory and application of three dimensionality reduction methods and four classification algorithms. In addition, it includes test runs with three separate sample sets and a short description of the working principle of the SELDI technology. A research team at the Department of Medical Bioscience, Pathology, Umeå University has produced two of the sample sets used. The master thesis project resulting in this report is a part of a larger study by Pernilla Wikström at this department. To get higher reliability of the classification results, two additional data sets produced by other research teams (Petricoin et al. at http://ncifdaproteomics.com/ and Adam et al. at http://www.evms.edu/vpc/seldi/) have also been used to evaluate the performance of the described methods. An additional task in this work has been to detect the specific proteins that have the greatest impact on the class belonging of the samples in the sample sets.

2 Previous work

In the last couple of years, several research teams have been working in the area of combining the SELDI process with pattern recognition algorithms to detect cancer in sample solutions from patients. Some of the published articles have focused on the classification part (separating cancer samples from healthy) and others have put more effort in finding specific important protein or peptides. A wide range of classification methods has been used and many different types of cancer have been studied in this research. Petricoin et al. have produced data sets (spectra) from both prostate and ovarian cancer patients and used *cluster analysis* combined with *genetic algorithms* to classify the samples in these sets [19], [14]. Adam et al. have applied the SELDI process on a set of prostate cancer samples and controls, and have in a series of articles tried different classification approaches to this data. In [1], they used the SELDI software program to detect peaks in the spectra and a decision tree for the final classification. The work in [20] uses the *Discrete Wavelet Transform* to reduce the dimensionality of the data and Fisher Linear Discriminant (FLD) for classification. In [21], that seems to be a development of the ideas in [1], two different methods have been used. These are called a *Boosted Decision Stump Feature Selection* and an *AdaBoost* classifier. Yasui et al. (one of the co-workers in Adam et al.) have in [23] proposed peak detection followed by a boosting algorithm to analyse prostate cancer samples and controls. Coombes et al. [8] have used a method including *principal component analysis* (PCA) to analyse a sample set from breast cancer patients. *Support Vector Machines* with preceding manual peak detection has been applied for prostate cancer in [5] by Ben-Hur et al. Ball et al. [4], have used SELDI together with *Neural Nets* to examine and classify different types of brain tumours. Donald et al. [10] have developed an interesting method called *Q5* that includes dimensionality reduction with PCA and the use of FLD in the classification step. They have tested this method on several of the data sets produced by authors of the other referenced articles in this section. Some other published material in this area are [15](prostate cancer), [17](breast cancer), [7](ovarian cancer) and [24](lung cancer).

It is hard to make a fair comparison between the classification results obtained in these articles. This is primarily due to the following two reasons. Firstly, the grouping of the samples is different in the separate cases. Some authors use two subsets, for example disease and healthy, while others use three or four subsets including a set with benign samples and/or two different types of cancer cells. Secondly, the way of dividing of the samples in training and test sets varies between the articles. In some cases *cross validation* (see section: 8) of the samples is used and in others a fixed training and test set are selected before the classification is performed. The former is known to be a reliable approach while the latter introduces uncertainty in the results especially when the number of samples is limited (as is often the case). If one is focused only on a comparison between classification algorithms, another source of uncertainty is of course the varying experimental conditions in the SELDI processes. Despite

these facts, clearly some of the algorithms proposed in the referenced work show great performance in classifying different categories of samples.

3 SELDI and mass spectrometry

3.1 Method description

The SELDI technology is a time-of-flight mass spectrometry. The name SELDI stands for "Surface Enhanced Laser Desorption/Ionization". The technology includes a special ProteinChip Array whose surface captures proteins using chemically or biologically defined docking sites. The array usually contains eight such chemically modified spots. There are several distinct chip chemistries, e.g. hydrophobic, anionic, cationic and metal binding. The proteins initially come from sample solutions based on human plasma, serum, or tissue. When the proteins have been captured on the surface of the ProteinChip, the surface is purified through several steps of washing. This is done to remove unbound proteins and other interfering substances. The proteins are then crystallized with small energy-absorbing molecules also known as 'matrix'. The task of these molecules is to absorb laser energy and transfer it to the proteins when in the next step a laser is shone on each spot on the chip. This laser causes the energized proteins to be released from the surface and then an electric field accelerates the molecules causing them to fly into and through a vacuum tube. The time it takes for the different proteins to pass through the tube (the *"time-of-flight"*) is a function of the molecular weight and charge of the proteins. The detector at the end of the tube measures the intensity of proteins at each discrete time of flight and the Data Analysis Workstation of the SELDI system then creates an output of many thousands of pairs containing the data (mass/charge, intensity), providing ultra-high resolution mass information. This can be done due to the fact that each discrete time of flight corresponds to a unique ratio of a molecular weight of a protein to the number of charges introduced by the ionization [23]. The unit of the mass to charge ratio is dalton (one dalton, Da is equal to one atomic mass unit). All these pairs together form a SELDI spectrogram. An example of such a spectrogram for a random sample is shown in figure 1. The working principal of the SELDI technology is pictured in figure 2.

3.2 Accuracy of output

The accuracy of the SELDI output has been tested in several studies by separate research teams [23, 1, 5]. It is obvious that the results of these studies depend highly on the experimental conditions but there are some common conclusions. One is that the coefficient of variation (CV) of the intensity values is relatively high (The CV is given by the standard deviation divided by the mean value and is estimated by repeating the SELDI process on the same samples). This means that there are experimental errors of the intensity at a given mass/charge point. The intensity values can sometimes vary considerably when the same sample is tested several times. In [23] for example, the CV is as high as approximately 50-60%. Another issue is that the intensity value is a function of the laser energy applied and the current mass/charge value, given that the amounts of proteins are constant. The intensity value increases with higher laser energy and lower

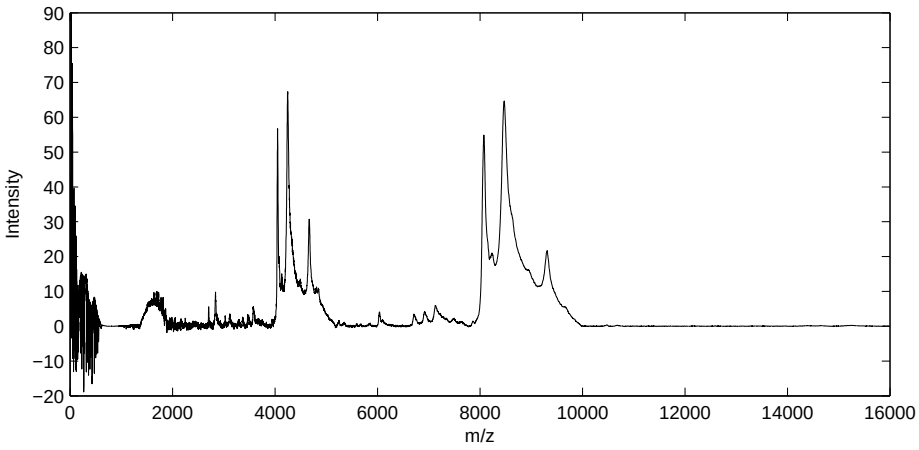


Figure 1: An example output from the SELDI system

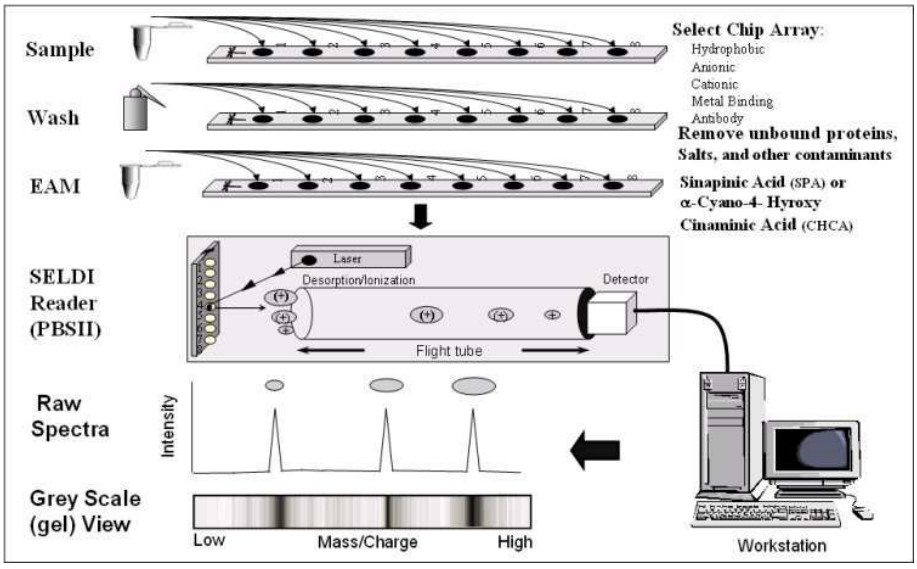


Figure 2: the SELDI system as illustrated in [13]

mass/charge value. This means that if the level of laser energy is fixed, the intensity is generally higher for low mass/charge points than for high [23]. A third uncertainty is the fact that the mass/charge-axis of the output shifts between experiments. The level of this error is approximately 0.1-0.3% [5]. The error increases along the axis and is higher for high mass/charge points than for low, but can be compensated for by day-to-day calibration of the instrument.

Large proteins usually have wider peaks in the spectrogram since they have more isotopes than small proteins. Another fact that causes the peak width to increase for large proteins is that they have more collisions with other molecules in the time-of-flight tube used in the technology. Protein glycosylation is also a source of uncertainty. Glycosylation means that each protein can have varying number of sugar groups attached to it at a certain point in time. These groups control the cellular activity of the protein and have the effect that they widen the peak width and make it difficult to identify the exact position of the peak in the spectrogram [5]. The first part of the spectrogram (corresponding to molecule weights below $\approx$ 2000 Da) mainly contains signals from the matrix thereby causing more noise than true information. This initial part is usually removed from the spectrogram before the data analysis is initiated. The majority of the proteins weigh considerably more than 2000 dalton [5].

The difficulty of obtaining reproducible measurements is a well-known limitation of the present SELDI systems [15] (April 2003). All these issues contribute to make classification and biomarker detection with SELDI data complicated.

4 Dimensionality reduction

In this particular application, the number of variables k describing each sample is very high. Every mass/charge value in the spectrogram corresponds to one variable and the intensity is the value of the variable. The magnitude of k depends on the chosen range in the SELDI process. A typical value of k is between 10 000-200 000. Before a classification method can be applied on the data, the number of variables for each sample must be reduced. This reduction is called a *dimensionality reduction*. The goal of dimensionality reduction is to represent each sample with d variables instead of k (where $d < k$), with as little loss of information as possible. This reduction must be performed to lower the computational time and space consumption in the following classification. Without this step, the classification algorithms would not be able to run on an ordinary workstation or personal computer. In this project, three different reduction methods have been studied:

- Principal Component Analysis (PCA)
- Independent Component Analysis (ICA)
- Discrete Wavelet Transform (DWT)

These methods have fundamentally different working principles. This is one of the reasons why they have been chosen in the first place. As an example, ICA and DWT can be performed on a single sample, while PCA must have access to a set of samples to be useful. It is interesting to study how well these working principles minimizes the information loss in the reduction. The method with the smallest loss is clearly the most suitable for the application. It is quite difficult to measure the information loss on the d variables directly. One possible way is to reconstruct the variables to the k-dimensional space and then compare the result to the original spectra. The method chosen in this report is to use the d variables from the three reduction algorithms on the same classifier and compare the classifier's performance in the separate cases.

4.1 Principal Component Analysis

One way to reduce the dimensionality of the samples is principal component analysis (PCA). Assuming that k is the initial number of variables, the goal is to describe each sample with d variables, where $d < k$ and with the loss of information being as small as possible (this goal of course applies to all different dimensionality reduction algorithms). In the case of PCA, the d new variables of a sample are linear combinations of the initial k variables.

Initially there are n_a samples of class ω_a constituting the set $\mathbf{X}_a$ and likewise n_b samples of class ω_b constituting the set $\mathbf{X}_b$. Let $\mathbf{X}'$ be $\mathbf{X}_a \cup \mathbf{X}_b$ $(k \times n)$ and $n = n_a + n_b$.

The first step is to normalize the data by calculating the all-class mean μ as

below:

$$\mu = \frac{1}{n}\left(\sum_{a \in \mathbf{X}_a} \mathbf{a} + \sum_{b \in \mathbf{X}_b} \mathbf{b}\right)$$

Then μ is subtracted from each sample in the set $\mathbf{X}'$ yielding $\mathbf{X}$. The samples in $\mathbf{X}$ are now zero-meaned and the mean of all variables corresponds to the origin in the k-dimensional space.

The first principal component (PC) can be thought of as the line in k-dimensional space that best approximates the data in a least squares sense [16]. If the data has been normalized as above, this line goes through the origin and to get the value of the first PC variable for the samples, the points corresponding to each sample is projected onto this line. The second PC is a line that also runs through the origin (all PC:s have this property) but is orthogonal to the first PC. The second PC has a direction which best approximates the data while preserving the orthogonality. The two components now form a plane and the PC variables of the samples are calculated by projecting the corresponding points onto this plane. In higher dimensions the PC:s span a d-dimensional subspace of the original k-dimensional space but the general principal of projection is the same. All PC:s are orthogonal to the previous components [16]. An expression of the original data and the PC:s is:

$$\mathbf{X} = \mathbf{VP} + \mathbf{E}$$

where $\mathbf{P}$ is a matrix $d \times n$, $\mathbf{V}$ is a matrix $k \times d$ and $\mathbf{E}$ is a matrix $k \times n$. The columns of $\mathbf{V}$ are the PC:s. $\mathbf{P}$ contains values describing the influence of each original variable on the d PC:s respectively and the residual matrix $\mathbf{E}$ contains the remaining noise (information not described by the PC:s).

Normal Approach

To evaluate the PC:s of a set of data the $k \times k$ covariance matrix $\mathbf{C}$ is computed:

$$\mathbf{C} = \mathbf{X}\mathbf{X}^T$$

The next step is to apply an eigendecomposition of $\mathbf{C}$. This yields at most $w = min(n, k)$ non-zero eigenvalues $\lambda_i (i = 1, 2, ..., w)$ with corresponding normalized eigenvectors $\mathbf{v}$. Each eigenvalue λ_i is proportional to the variance of the original data in the direction of the ith PC. By sorting the w eigenvalues and then selecting the d largest values, the matrix $\mathbf{V}$ can be obtained. $\mathbf{V}$ is constructed by inserting as columns the d eigenvectors corresponding to the d largest eigenvalues [10].
The dimensionality reduced variable matrix $\mathbf{X}_{red}$ for all samples can then be computed:

$$\mathbf{X}_{red} = \mathbf{V}^T\mathbf{X} \quad (1)$$

The dimension of the samples has now been reduced from k to d. Typically, only a few PC:s accounts for nearly all of the sample variation [10]. The best value of d must however be carefully evaluated in each separate situation.

Computational Complexity

In this particular application where the number of variables k is much greater than the number of samples another approach can be chosen for better computational efficiency. Instead of calculating the covariance matrix $\mathbf{C}$ with size $k \times k$ the *Gram matrix* $\mathbf{C}'$ is calculated as:

$$\mathbf{C}' = \mathbf{X}^T\mathbf{X}$$

The Gram matrix has size $n \times n$ and has the property that the eigenvalues of $\mathbf{C}'$ equals the eigenvalues of $\mathbf{C}$. The eigenvectors $\mathbf{v}_i$ of $\mathbf{C}$ can then be evaluated as:

$$\mathbf{v}_i = \mathbf{X}\mathbf{v}'_i$$

where $\mathbf{v}'_i$ $(i = 1, 2, ..., w)$ are the normalized eigenvectors of $\mathbf{C}'$. The PC matrix $\mathbf{V}$ and the variable matrix $\mathbf{X}_{red}$ can then be computed as above.

It is important to note that PCA does not take into account that the samples belong to separate classes. In the worst case scenario it is possible to leave out exactly the components that are needed to distinguish between the different sets of samples in the following classification step [9]. In this application however, the PCA has shown to preserve much of the information contained in the initial data. PCA should be used only for dimensionality reduction and never directly for classification purposes [10].

4.2 Independent Component Analysis

Independent component analysis (ICA) is another method for dimensionality reduction and feature extraction. ICA enables data the data $\mathbf{X}$ to be represented by statistically independent components. PCA described in the previous section, leads to uncorrelated components, which is a weaker statistical property than statistical independence [18]. Assume that $\mathbf{X}$ is a matrix containing the original data $(k \times n)$, where k is the number of variables and n is the number of samples. ICA projects the data $\mathbf{X}$ in a direction, which leads to maximum independence of the d components. This projection is linear and non orthogonal [18]. ICA reduces the dimensionality from k to d and maximizes at the same time the non-gaussianity of the d new features. By doing this ICA obtains the features that contribute the most information [3]. The calculation of the vector $\mathbf{s}$ containing the d components can be written as:

$$\mathbf{x} = \mathbf{A}\mathbf{s}$$

where $\mathbf{x}$ is sample in $\mathbf{X}$ and $\mathbf{A}$ is called the mixing matrix, containing linear combinations of the d independent components (IC). Rather than focusing on $\mathbf{A}$, its inverse $\mathbf{W}$ $(d \times k)$, is of more importance when $\mathbf{s}$ is to be evaluated. This can be done as:

$$\mathbf{s} = \mathbf{W}\mathbf{x}$$

$\mathbf{W}$ is called the filter or projection matrix [6]. In practise, $\mathbf{W}$ cannot be determined exactly and must be approximated. To do this, a measure of non-gaussianity is needed. This measure is to be used on the d IC:s in $\mathbf{s}$. Two such measures are *the Kurtosis* and *negentropy*, the former is more sensitive to outliers in the data and negentropy is therefore considered more robust [18]. The matrix $\mathbf{W}$ is usually initiated with random values and is then updated through numerous iterations i until the values in the update matrix $\mathbf{dW}$ is below a fixed level.

$$\mathbf{W}_{i+1} = \mathbf{W}_i + \mathbf{dW}_i$$

The values in $\mathbf{dW}$ depend on the measure of non-gaussianity applied to $\mathbf{Wx}$ in each iteration. Every iteration step normally also applies a non-linear function to $\mathbf{Wx}$ to speed up the convergence of the algorithm. More information about *negentropy* and *the Kurtosis* can be found in [3, 6]. The implementation of ICA in this project uses the *Fast-ICA algorithm* [2] for the determination of $\mathbf{W}$.

When ICA is performed as a preprocessing step in classification (as in this case), separate projection matrices can be calculated for each class. This can in some cases lead to better discrimination in the classification step [6].

The main disadvantage with ICA is the computational time complexity for estimating $\mathbf{W}$ [6]. This becomes apparent, especially when several different models are to be compared, e.g. using cross validation. Once $\mathbf{W}$ is estimated, the calculation of the IC:s for the original samples is straightforward. Another important issue to consider is that ICA does not work if the IC:s are gaussian in nature. In this case only one component can be calculated.

4.3 Discrete Wavelet Transform

A wavelet transform is a way to decompose a signal in a chosen number of its constituent parts. Fourier analysis also has this property but wavelet analysis has some advantages when it comes to analysing signals of non-stationary nature. Wavelets have more irregular shape and are more compactly supported than sine waves (used in Fourier analysis), which are smooth and infinite in length. These properties make wavelets ideal for analysing signals with discontinuities and sharp changes while it enables temporal localisation of the features of the signal [11]. In the first step the discrete wavelet transform (DWT) decomposes the original signal into two parts. The first part (l_1 in figure 3) includes the low frequency components of the original signal. These components usually give the signal its main characteristics. The second part (h_1 in Figure 3) is

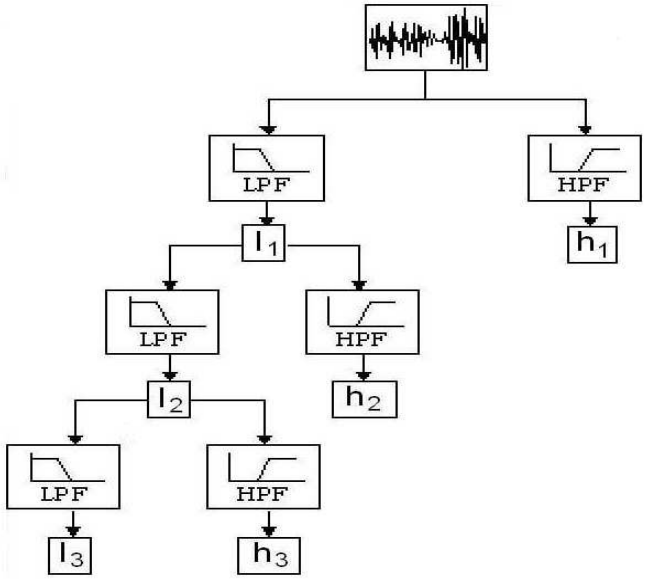


Figure 3: DWT decomposition tree

constituted by the high frequency components of the signal and gives the signal its details and nuance's. These properties make it possible to view the wavelet transform as a filter bank and the transforming of a signal as passing the signal through this bank [22]. A low pass filter is used to construct the smooth part and a high pass filter to create the detailed part. This can be an effective way to eliminate noise in a signal. Both parts have bandwidth that is half of the original signal. This means that if the original signal includes k sampled points, the two constructed signals have $k/2$ points each. The low frequency part can then be recursively decomposed into two new filtered signals with different frequency content. The amplitudes of the signals increase with each step. The principal of decomposition is shown in figure 3. Let j denote the number of recursive steps of the decomposition. The maximum number of steps j_{max} can be calculated by the formula: $1 = \lfloor k/2^{j_{max}} \rfloor$. Figure 4 shows an example of a decomposed spectra.

In this application, the output spectrum of a sample from the SELDI system is interpreted as the original signal. The DWT is like PCA an orthogonal transform (a rotation of the coordinate system) but the rotation directions are in this case prespecified. This means, unlike PCA, that the transformation can be performed on one sample at a time [20]. The transformation can be written as $\mathbf{y} = \mathbf{W}\mathbf{x}$ where $\mathbf{x} \in \mathbf{X}$(the sample matrix, $k \times n$) and fortunately, the matrix $\mathbf{W}$ $(k \times k)$ does not have to be produced to calculate $\mathbf{y}$ [20]. The DWT algo-

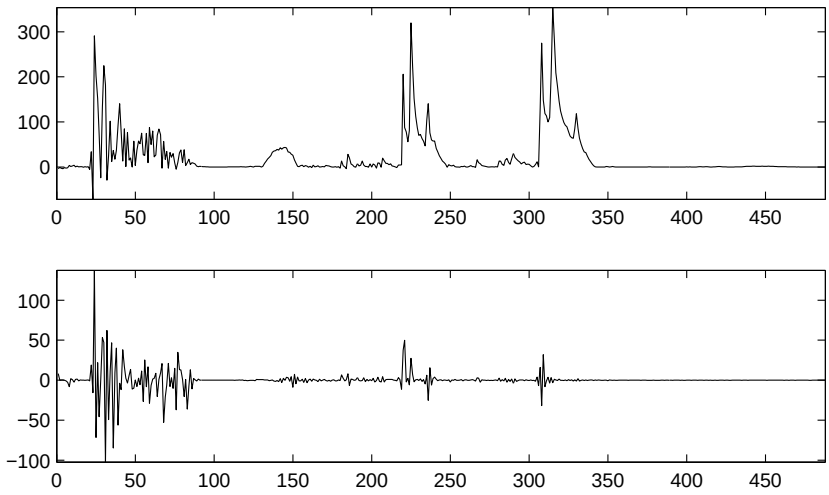


Figure 4: The decomposed spectrogram from Figure 1 (j=5, the smooth part is pictured above).

rithm makes the transformation much faster. The vector $\mathbf{y}$ contains the *wavelet coefficients* of $\mathbf{x}$. Let l denote a smooth part and h denote a detailed part of one decomposition step. Then in

step 1: $\mathbf{x} = l_1 + h_1$
step 2: $\mathbf{x} = l_2 + h_2 + h_1$
step j: $\mathbf{x} = l_j + h_j + h_{j-1} + \ldots + h_1$
The total transformation is $\mathbf{x} \rightarrow \mathbf{y}$ where $\mathbf{y} = (l_j, h_j, h_{j-1}, \ldots, h_1)$

A wavelet is a square integrable function. Two such functions, called a father and a mother wavelet are used in the transformation [20]. The father wavelet integrates to 1 and the mother wavelet to 0. Many different such functions have been proposed, for example the series of *Daubechies* and *Symlet* wavelets.

$$\int \varphi(t)dt = 1, \quad \int \psi(t)dt = 0$$

From these functions two new sets of functions are generated. These are indexed by the level index j and shift index s.

$$\varphi_{js}(t) = 2^{-j/2}\varphi\left(\frac{t - 2^j s}{2^j}\right), \quad \psi_{js}(t) = 2^{-j/2}\psi\left(\frac{t - 2^j s}{2^j}\right)$$

The original signal $\mathbf{x} = f(t)$ can be expressed as a linear combination of these

functions:

$$f(t) \approx \hat{f}(t) = \sum_s l_{js}\varphi_{js}(t) \quad + \sum_{1 \leqslant j \leqslant j_{max}} \sum_s h_{js}\psi_{js}(t)$$

The wavelet coefficients can be approximated by:

$$l_{js} \approx \int \varphi_{js}(t)f(t)dt, \quad h_{js} \approx \int \psi_{js}(t)f(t)dt$$

When using DWT as a dimensionality reducer the question of which coefficients to keep is not yet answered. Note that the resulting vector $\mathbf{y}$ still has k components. To reduce the dimensionality from k to d the most important coefficients in $\mathbf{y}$ must be found. One simple way to approximate these is to plot the signals during the decomposition and manually estimate the depth j and the interval where the mean curves of the healthy and the cancer samples differs the most in amplitude. Another way is to choose a threshold γ and keep the coefficients which absolute values exceeds γ and set all others to zero for every sample. The samples are then put together in a set and if a certain coefficient is greater than zero for a certain ratio r of the samples, it is chosen as a final coefficient [20]. The number of resulting coefficients d is decided by γ and r. It is possible to reconstruct the original signal $\mathbf{x}$ by using the inverse discrete wavelet transform (IDWT). How precise this reconstruction will be depends on the number of coefficients in $\mathbf{y}$ that are used. IDWT is not applied in this particular application however.

5 Classification methods

The aim of a classification method is to predict which class or category an observation (or a set of observations) belongs to. This is done by investigating a feature vector $\mathbf{x}$ with d elements that describes the properties of the observation or sample. Each of the d elements is a measure of a certain property of the observation. If the method is to be able to make successful predictions, it must have some knowledge about the magnitude of the properties for each class individually. The method gets this knowledge by studying a set of observations with known class belonging. This set is called a *training set* and should include observations from all c classes. This study sets a number of parameters for the method. The method together with its parameters is called a *classifier*. This report describes four different types of classification methods:

- Bayesian Discriminant Functions
- Fisher Linear Discriminant
- K Nearest Neighbours
- Neural Networks

The nature of the parameters varies between the methods. Examples of parameters are mean values and covariances in Bayesian Discriminant Functions, The number of neighbours in K Nearest Neighbour or the different weights in a Neural Net. When the parameters have been set, the next step is to use the classifier to classify unknown observations in a *test set*. The classification performance measured on this test set depends on the quality of the estimated parameters and how well adapted the method is for the specific application at hand. One method can perform very well under certain circumstances and very poorly under others. This is the reason why it is often interesting to try more than one method in a classification task.

The classes or categories in this project are the health status of the patients from which the samples are taken. These can be divided in two, three or four groups, depending on the information contained in the different data sets and what type of results one wants to obtain from the classifier. More about this can be found in the section *Implementation and testing*.

5.1 Bayesian Discriminant Functions

Bayesian discriminant functions is a way to classify a new observation when there exists a set of observations $\mathbf{X}$ for which the true classes are known. It can be shown that a Bayesian classifier minimizes the error in the classification step if the distributions of the d variables in the feature vectors $\mathbf{x}$ are completely known [9]. This is usually not the case. Instead the distributions are often approximated using the variables of the available observations. The most common assumption is that the variables have gaussian distributions. This idea has been used in this application as well.

Every class ω_i has a *prior probability* $P(\omega_i)$. $P(\omega_i)$ states the probability that a randomly selected sample belongs to class ω_i. Assuming that every sample belongs to some class, the sum of the prior probabilities is 1:

$$\sum_{i=1}^{c} P(\omega_i) = 1$$

where c is the number of classes. If nothing more than the prior probabilities are known, the unknown sample should be classified as belonging to the class with highest prior probability. In a classification problem, the feature vector $\mathbf{x}$ of the unknown sample is usually also known. The task is then to evaluate the *posterior probability* $P(\omega_i \mid \mathbf{x})$. This is the probability that the sample belongs to class ω_i given $\mathbf{x}$. The posterior probability can be expressed with Bayes' rule:

$$P(\omega_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_i)P(\omega_i)}{p(\mathbf{x})} \tag{2}$$

In addition to the prior probability, a probability density function $p(\mathbf{x} \mid \omega_i)$ of each class must be used when applying Bayes' rule. The value of $p(\mathbf{x})$ is the same for every class and can be discarded. The classification step is performed by using a discriminant function $g_i(\mathbf{x}) = p(\mathbf{x} \mid \omega_i)P(\omega_i)$ for each class. The classification error is minimized if the sample is classified as belonging to the class with the highest value of $g_i(\mathbf{x})$. In the case where $c = 2$, one single discriminant function can be used:

$$g(\mathbf{x}) = g_a(\mathbf{x}) - g_b(\mathbf{x})$$

where the class ω_a is chosen for the observation if $g(\mathbf{x}) > 0$. The effect of using discriminant functions on feature vectors is that $\mathbb{R}^d$ is partitioned into separate regions $R_1, \ldots, R_c \in \mathbb{R}^d$. The decision boundary between the classes a and b is the points in $\mathbb{R}^d$ where $g_a(\mathbf{x}) - g_b(\mathbf{x}) = 0$.

Collinear or Uncorrelated Variables

When $\mathbf{x}$ contains collinear or uncorrelated variables, the gaussian distribution for the d variables for class ω_i can be written as:

$$p(\mathbf{x} \mid \omega_i) = \frac{1}{(2\pi)^{d/2}|\mathbf{C}_i|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu_i)^T \mathbf{C}_i^{-1}(\mathbf{x}-\mu_i)}$$

where μ_i is a vector containing the mean values and $\mathbf{C}_i$ is the covariance matrix of class ω_i. It can be shown that the discriminant function $g_i(\mathbf{x})$ can be expressed as:

$$g_i(\mathbf{x}) = \mathbf{x}^T \mathbf{H}_i \mathbf{x} + \mathbf{h}_i^T \mathbf{x} + \omega_{i0}$$

where

$$\mathbf{H}_i = -\frac{1}{2}\mathbf{C}_i^{-1} \quad \mathbf{h}_i = \mathbf{C}_i^{-1}\mu_i \quad \omega_{i0} = -\frac{1}{2}\mu_i^T \mathbf{C}_i^{-1}\mu_i - \frac{1}{2}ln|\mathbf{C}_i| + lnP(\omega_i)$$

The part $lnP(\omega_i)$ can be discarded if the prior probabilities of the classes are equal. The expression $g_i(\mathbf{x})$ can be simplified further in some cases when the covariance matrices $\mathbf{C}_i$ have certain properties, for example when $\mathbf{C}$ is equal for every class. The expression above shows the general case however.

Independent variables

If the variables in $\mathbf{x}$ are statistically independent, the discriminant function can be evaluated with a more concise expression. The dimensionality reduction process ICA yields variables of this kind. The gaussian distribution of each variable $x_1, \ldots, x_d \in \mathbf{x}$ can then be written as:

$$p(x \mid \omega_i) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2}(\frac{x-\mu_i}{\sigma_i})^2}$$

where μ is the mean value and σ is the variance of the variable $x \in \mathbf{x}$. Assuming that the prior probability for each class is equal or unknown, the discriminant function for the class ω_i is calculated as follows in the independent case:

$$g_i(\mathbf{x}) = \prod_{m=1}^{d} p(x_m \mid \omega_i) \tag{3}$$

5.2 Fisher Linear Discriminant

The aim of the classification method Fisher Linear Discriminant (FLD) is to project the samples in the d-dimensional space onto a single line with a certain direction. This process reduces the dimensionality of the samples from d to 1. The projection can be expressed as:

$$y = \mathbf{w}^T \mathbf{x} \tag{4}$$

where $\mathbf{x}$ is a sample in d-dimensional space and $\mathbf{w}$ is a vector specifying the direction of the line. The goal is to find the direction of $\mathbf{w}$ that best discriminates the classes ω_a and ω_b. The projection of the samples belonging to ω_a should form

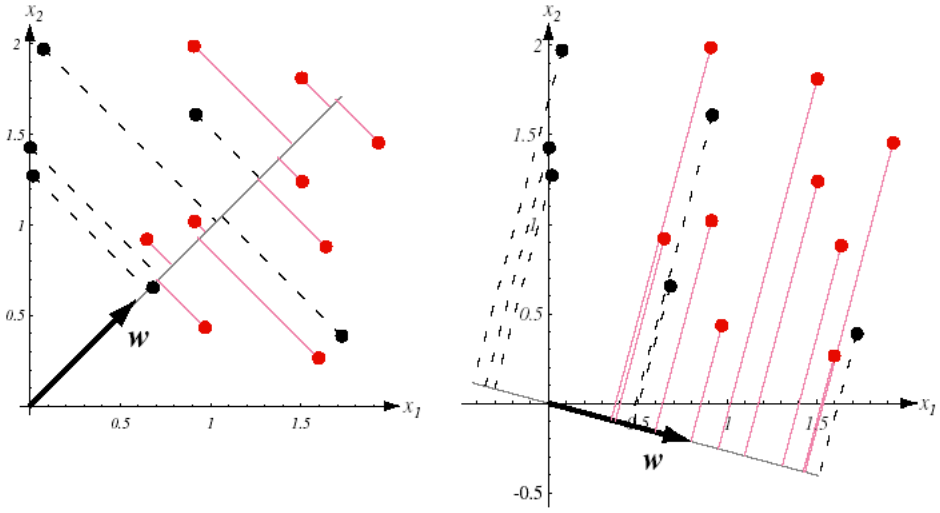


Figure 5: Projections of observations in two different directions. The figure to the right shows better discrimination than the left (The Picture is taken from [9]).

one cluster on the line and the samples belonging to ω_b should if possible form a separate, non-overlapping cluster. Figure 5 shows two different projections of observations with $d = 2$.

The generalization to the multiple-class case ($c > 2$) is described in [9]. The following steps apply to the two-class case ($c = 2$). The first step is to calculate the mean values for each class and for all samples put together:

$$\mu_a = \frac{1}{n_a}\sum \mathbf{x}_a \quad (\mathbf{x}_a \in \mathbf{X}_a), \quad \mu_b = \frac{1}{n_b}\sum \mathbf{x}_b \quad (\mathbf{x}_b \in \mathbf{X}_b)$$

$$\mu = \frac{1}{n}\sum \mathbf{x} \quad (\mathbf{x} \in \mathbf{X})$$

where n is the number of samples. $\mathbf{X} = \mathbf{X}_a \cup \mathbf{X}_b$ is the sample matrix and $\mathbf{x}$ is one sample. The indices a and b indicates the class belonging. One way to measure the separation of the projected points is to measure the *scatter* which is decided by the difference of mean values relative to the standard deviation for the classes. The scatter between the classes, $\mathbf{S}_B$ is decided by the difference of the class means and can be written as:

$$\mathbf{S}_B = n_a(\mu_a - \mu)(\mu_a - \mu)^T + \quad n_b(\mu_b - \mu)(\mu_b - \mu)^T$$

The scatter within the classes $\mathbf{S}_W$ depends on the variation of the variables for

each class individually and is calculated:

$$\mathbf{S}_W = (\mathbf{X}_a - \mu_a)(\mathbf{X}_a - \mu_a)^T + \quad (\mathbf{X}_b - \mu_b)(\mathbf{X}_b - \mu_b)^T$$

Intuitively, the *between-scatter* should be as high, and the *within-scatter* as low as possible for a well performed discrimination. This is indeed the case since it can be shown that the optimal vector $\mathbf{w}$ in the FLD-process is the one that maximizes the criterion function $J(\mathbf{w})$:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

and satisfies $\mathbf{S}_B\mathbf{w} = \lambda \mathbf{S}_W\mathbf{w}$, where the largest eigenvalue λ can be obtained by an eigendecomposition of $\mathbf{S}_W$ together with $\mathbf{S}_B$. The projection vector $\mathbf{w}$ is the eigenvector that corresponds to the eigenvalue λ. The matrices $\mathbf{S}_B$ and $\mathbf{S}_W$ have a rank of at most $n - c$. This means that d must be less than $n - c$ to avoid guaranteed singularity in the eigendecomposition. This is the reason why it is not possible to apply FLD directly on the raw data in this application [10]. The resulting points y on the line of the projected samples of class ω_a and ω_b can then be evaluated as:

$$y_a = \mathbf{w}^T \mathbf{x}_a \quad (\mathbf{x}_a \in \mathbf{X}_a)$$

$$y_b = \mathbf{w}^T \mathbf{x}_b \quad (\mathbf{x}_b \in \mathbf{X}_b)$$

When the vector $\mathbf{w}$ has been established, the classification step is straightforward. One simple way is to estimate a probability density function $p(y \mid \omega_i)$ for each class from the projected training set and then project the unknown sample onto the line. The sample can be classified as belonging to the class that has the highest value of the density function at that particular point on the line. Figure 6 shows an example of a classification with this technique.

5.3 K Nearest Neighbours

The k Nearest Neighbours method (KNN) is a widely used technique for solving classification problems. KNN is a *non-parametric* procedure. This means that it is not necessary to make assumptions about the underlying probability density functions of the d features in the feature vectors. This is a desirable property since in practise the parameters of density functions are often hard to obtain. KNN approximates the density function $p_n(x)$ of the features as:

$$p_n(\mathbf{x}) = \frac{k_n/n}{V_n}$$

where k_n is the number of observations within a certain volume V_n in the feature space with dimension d and n is the total number of observations. The size of V_n is chosen so that it contains k observations with the feature vector $\mathbf{x}$ as the center point. The approximation gets more accurate when the number of

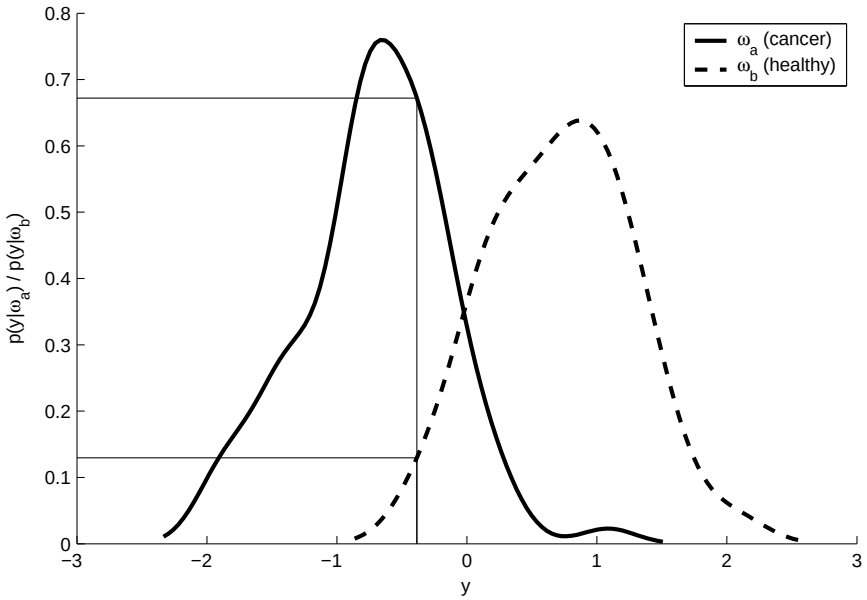


Figure 6: The patient is classified as having cancer since the probability of disease is higher than the probability of being healthy.

samples increases [9]. When an observation $\mathbf{x}$ is to be classified, a separate approximation is performed for each class ω_i:

$$p_n(\mathbf{x}, \omega_i) = \frac{k_i/n}{V}$$

k_i is the number of observations of the class ω_i within the volume V. The probability that an observation $\mathbf{x}$ belongs to a certain class $P_n(\omega_i \mid \mathbf{x})$ can then be calculated as:

$$P_n(\omega_i \mid \mathbf{x}) = \frac{p_n(\mathbf{x}, \omega_i)}{\sum_{j=1}^{c} p_n(\mathbf{x}, \omega_j)} = \frac{\frac{k_i/n}{V}}{\sum_{j=1}^{c} \frac{k_j/n}{V}} = k_i/k$$

where c is the number of classes. The probability $P_n(\omega_i \mid \mathbf{x})$ is evaluated for each class and $\mathbf{x}$ is classified as belonging to the class with the highest probability, i.e. maximizing the expression k_i/k. KNN can be implemented by calculating the distance from the observation $\mathbf{x}$ of unknown class to all the other n observations in the training set and then selecting the k observations with the shortest distance. $\mathbf{x}$ is classified as belonging to the class, which is most common among the k neighbours. In the two-class case the number k should be odd to avoid ties in the classification. The magnitude of k is chosen depending on the number of observations n. k should be increased when n is high and decreased when n

is low. The optimal value of k in specific application can be evaluated through testing.

Metrics

The distance between observations can be calculated with different metrics. A common metric is the *Euclidean distance*, the L_2 norm. The distance between the observations $\mathbf{x}$ and $\mathbf{y}$ is defined as:

$$L_2 = \sqrt{\sum_{m=1}^{d} (x_m - y_m)^2}$$

Other metrics are for example the *Minkowski metric* (the L_k norm) and the *Tanimoto metric*. The choice of metric can be based on the range of the data among the different axes in the d-dimensional space. If there is great difference in range, other metrics than the L_2 norm should be considered or the data should be scaled [9].

Computational Complexity

The implementation of KNN can be computational burdensome when the number of observations n is large and the value of d is high. *Pruning*, *Partial Distances* and *Search Trees* are three ways of dealing with this problem. In this application with relatively few samples, this has not been an important issue however. For more information, see [9].

5.4 Neural Networks

A Neural Network consists of a set of connected neurons. The neurons can be seen as nodes in a graph where the connections between them are the edges of the graph. In a Multilayer Neural Network, the neurons (also called units) are divided into at least three separate layers. These layers are called the input layer (denoted i), output layer (k) and the hidden layer (j) of the network. If the network has more than three layers, there is more than one hidden layer. All units in the input layer are connected to some or all of the nodes in the hidden layer and all units in the hidden layer are connected to some or all of the nodes in the output layer. The following theory describes a fully connected, feed-forward, three-layer network used for classification. Feed-forward means that the information in the network flows from the input to the output layer. For more information on this terminology, see [9].

All the edges in the network has a weight w associated with it. Each neuron sums the inputs to itself from all incoming edges. These inputs are calculated by multiplying the weight of the edge with the output from the neuron in the

previous layer. The sum of the j^{th} neuron in the hidden layer can be written as:

$$net_j = \sum_{i=0}^{d} x_i w_{ij} = \mathbf{w}_j^T \mathbf{x} \tag{5}$$

where there are d input units with values $x_i, \ldots, x_d$ ($x_0 = 1$ is called the *bias* of the unit). The neuron then creates it own output y by applying an activation function f_j to the sum net_j.

$$y_j = f_j(net_j)$$

The activation function must be non-linear but should be linear in the area where *net* is near zero to be able to approximate linear decision boundaries between the different classes. Two functions that meet these criterions are the *Tanh* and *Log-Sigmoid* functions. The sum of the kth neuron in the output layer can be written in the same way:

$$net_k = \sum_{i=0}^{h} y_j w_{jk} = \mathbf{w}_k^T \mathbf{y}$$

This sum is then activated:

$$z_k = f_k(net_k)$$

In a classification application such as this one, the neural networks normally have d units in the input layer, one for each variable in the feature vector $\mathbf{x}$. The output layer has c units, where c is the number of classes. The hidden layer has h neurons. As a rule of thumb, h should be chosen so that the total number of parameters in the network is less than the total number of observations n in the training set. The best value for h must be tested in each specific application. More on the magnitude of h can be found later in this chapter. Figure 7 shows an neural network of this kind.

Every output value z_k from units $1, \ldots, c$ can be interpreted as a discriminant function for one of the classes. A sample is classified as belonging to the class with the highest value for this function:

$$g_k(\mathbf{x}) = f_k\Big(\sum_{j=0}^{h} w_{jk} f_j\Big(\sum_{i=0}^{d} x_i w_{ij}\Big)\Big)$$

An neural network is a very powerful machine. In fact, it can be proven that a three-layer network can approximate all physical realizable functions if the

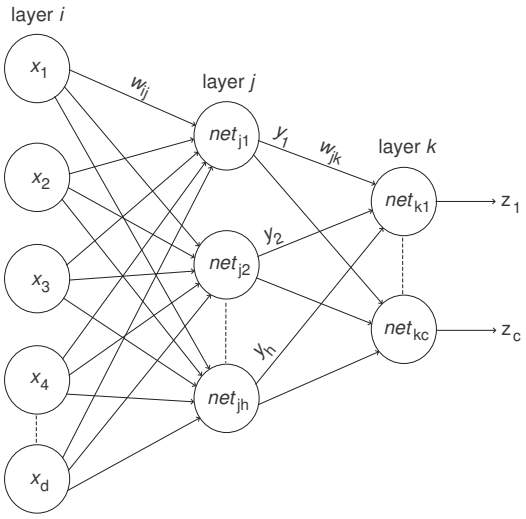


Figure 7: Working principle of a three-layer neural net

number of units in the hidden layer h is high enough. If the set of available observations is large enough, $g_i(\mathbf{x})$ will approximate $P(\omega_i \mid \mathbf{x})$ from equation (2).

To get the network to perform well in a classification task it must be trained with observations from a training set (network learning). Before this is done the weights w is initialised randomly to small non-zero values. One common way to train a network is to use the *back propagation* algorithm, which has three major steps. First, the output from one observation $\mathbf{x}$ is calculated as: $\mathbf{z} = g(\mathbf{x})$. Then the mean-squared error between the output and the target vector $\mathbf{t}$ is evaluated. The last step is to adjust the weights of the network so that the error decreases. These steps are then repeated through several iterations. The weight adjustments are propagated backwards from the output to the input layer. The back propagation algorithm uses a criterion function $J(\mathbf{w})$ and tries to find the weights $\mathbf{w}$ that minimizes this function:

$$J(\mathbf{w}) = \frac{1}{2}\sum_{k=1}^{c}(t_k - z_k)^2$$

One way to update the weights is to use *gradient descent*:

$$\mathbf{w}(m+1) = \mathbf{w}(m) + \Delta\mathbf{w}(m) \quad where \quad \Delta\mathbf{w} = -\eta\frac{\partial \mathbf{J}}{\partial \mathbf{w}}$$

where m denotes the iteration step in the update process. Details regarding

the weight adjustments in the different network layers can be found in [9]. The target vector $\mathbf{t}$ contains information about the true class of the observation $\mathbf{x}$. If there are three classes and $\mathbf{x} \in \omega_1$, then a possible $\mathbf{t}$ could be $[1\ 0\ 0]^T$. The *learning rate* η decides how fast the network learns. If η is too high, it will cause the algorithm to diverge, meaning that the global minima is not found. If it is too low it will converge but at a very slow rate (m will be high). There is an optimal value for η that will find the minima in just one step, but this value is not known in advance. Note that if a specific weight w_{ab} becomes zero in the update process, this is equal to removing the connection between node a and b.

As mentioned, a neural network can learn almost anything, having a large set of hidden units. This means that if all the observations in the training set are presented to the network through several iterations (one iteration is called an *epoch*), the network will be able to learn the training set perfectly. This usually results in poor generalization, i.e. the classification error of unknown samples will be high. One way of dealing with this problem is to use a method called *Early stopping*. In Early stopping, some of the observations in the training set are moved to a validation set. The remaining training set is used to train the network, and after each epoch the generalization error is calculated by classifying the observations in the validation set with the current trained network. The learning is stopped when the error in the validation set starts to increase.

It is important to normalize the d variables in the feature vectors before the observations are presented to the net. If for example,
$x_i >> (x_1, \ldots, x_{i-1}, x_{i+1}, x_d)$ then $x_i w_{ij}$ will dominate the expression in equation (5) and y_j will depend almost entirely on x_i. To avoid this problem, an appropriate normalization is often to the give each variable mean 0 and variance 1.

6 Back-Projection

In addition to classification of samples in order to decide if a patient is healthy or has cancer, it is also of much interest to determine the identities of the various biomolecules that contribute the most to the result of the classification. That is, to find the mass/charge (m/z) values in the initial spectra that contains most information about the difference between the classes (the following theory applies to the two class case). Finding these biomarkers will not make the classification system more accurate but is of great importance from a discovery perspective. The identities of the protein biomarkers are needed to understand the biological role of these proteins in the oncogenesis of prostate cancer. Knowing these identities will facilitate the production of antigen and antibody reagents for development of classic multiplex immunoassays [21]. This knowledge could lead to more specific and sensitive prognostication of cancer.

The use of the dimensionality reducing process *Principal Component Analysis* (PCA) in combination with the classification method *Fisher Linear Discriminant* (FLD) provides a straightforward way to identify the most important m/z values. This *back-projection* process has been proposed in [10]. The matrix $\mathbf{V}$ in equation (1) can be multiplied with the projection vector from FLD, $\mathbf{w}$ in equation (4):

$$\mathbf{w}' = \mathbf{V}\mathbf{w}$$

where $\mathbf{w}'$ is called the *spectral-space linear discriminant* with size $k \times 1$ (k is the total number of m/z values in the initial spectra). In order to determine the most important positions in $\mathbf{w}'$, the elements must be normalized. Doing this yields a significance vector $\mathbf{s}$ with the elements $s_1, \ldots, s_k$. The elements are calculated as:

$$s_i = \mid \mathbf{w}'_i(\mu_{d_i} - \mu_i) \mid$$

where μ_d is the average cancer spectra and μ is the all-class mean for the raw data. Each s_i represents the importance of the ith m/z value in the spectra. The elements in $\mathbf{s}$ with the highest values are the most significant. These can easily be obtained by sorting $\mathbf{s}$. When the back-projection process is performed it is important to take the lack of accuracy of the m/z values in the SELDI-process into consideration. The most significant elements in $\mathbf{s}$ can contain several values that lie very close to each other on the m/z-axis. This may indicate that it is in fact the same protein that has been identified.

It is possible to check the classification performance of the most important m/z values by applying equation (4) using the intensities corresponding these m/z values in the raw spectra (making the assumption that intensities of the chosen m/z values are statistically independent). This will in most cases result

in worse performance than a classification of the whole spectra (containing all information) but it will indicate how well these m/z values explains the class belonging of the samples.

7 Data sets

7.1 WIKSTRÖM et al.

CAPS

This sample set consists of 200 plasma samples from a large case-control study of prostate cancer in Sweden (CAPS). 100 of the samples were identified as coming from patients having prostate cancer and the other 100 (the control group) comes from a normal population, i.e. persons with no symptoms of prostate cancer. In fact, it is highly likely that the control group may include samples from persons with developing cancer, which had not been detected at the time of collection. 91 of the 100 cancer samples came from patients with organ-confined cancer and the remaining 9 came from patients where the cancer had spread to the seminal vesicle. The mean PSA level for the cancer samples was 19 ng/ml. The patients in the CAPS set did not fast before the samples were taken. In the north of Sweden, CAPS has around 80% coverage of all newly diagnosed prostate cancer cases. This set with a size of 200 samples, is only a small part of the total number of usable samples in CAPS and is used to establish a prognostic model prior to a larger study including 1500 + 1500 samples and controls. All samples were analysed twice with the SELDI system yielding a total of 400 mass spectrograms for each system setting.

The SELDI system was used with two different protein chip types, the CM10 and IMAC30/Cu^{2+} chips. CM10 is a cation exchanger and IMAC30/Cu^{2+} binds protein to the divalent cation Cu^{2+}. In addition, two different levels of laser intensity were applied to all samples, low and high. The samples in this experiment came from Heparinplasma. The Molecular Mass Range for the four different settings were set to:

- CM10 low: 3000-11000 Dalton
- CM10 high: 11000-200000 Dalton
- IMAC low: 2500-16000 Dalton
- IMAC high: 16000-200000 Dalton

PP1

The data set called *PP1* is a smaller set consisting of 25 cancer and 25 healthy samples collected from the Department of Urology at Norrlands Universitetssjukhus. 18 of the 25 cancer samples came from patients with organ-confined cancer and the remaining 7 came from patients where the cancer had spread to the seminal vesicle. The mean PSA level for the cancer samples was 18 ng/ml. The samples in the PP1 set came from fasting patients. The SELDI system was used with the same protein chips, laser intensities and other system settings as

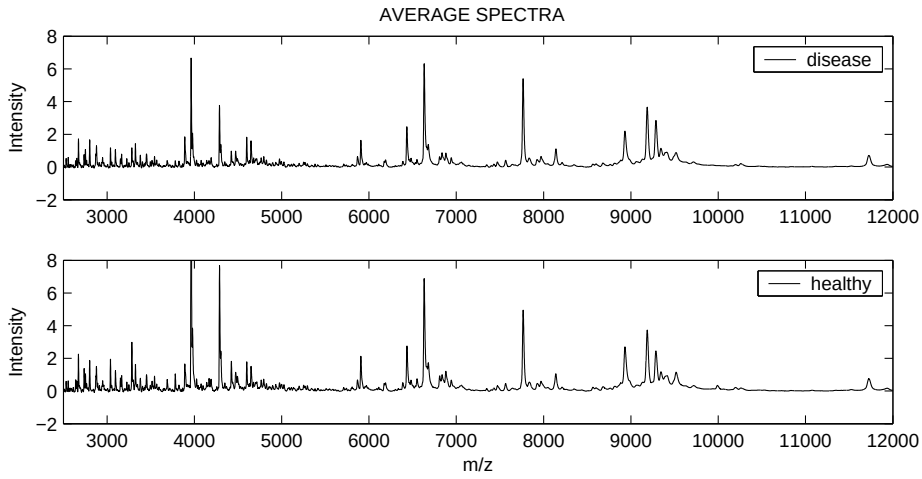


Figure 8: Average spectra from Wikström CAPS data set. Low Laser Intensity and the IMAC30 Chip.

the CAPS set. All samples were run twice through the SELDI process and the resulting average spectra was used in the analysis. The differences between the PP1 and CAPS sets are the source and the Molecular Mass Ranges used:

- CM10 low: 2500-17000 Dalton
- CM10 high: 11000-200000 Dalton
- IMAC low: 2500-17000 Dalton
- IMAC high: 3000-200000 Dalton

Figure 8 and 9 show the average spectra for two specific system settings used with the CAPS and the PP1 set respectively.

7.2 PETRICOIN et al.

In addition to the datasets provided by the department of medical bioscience, additional samples were used to test the performance of the algorithms described in this report. One such set of samples was downloaded from [12]. The SELDI method and a mass spectra analysis for this sample set have been performed by Petricoin et al. in [19]. The set consists of four subsets:

- 63 samples with no evidence of disease and PSA level less than 1 (ng/ml)
- 190 samples with benign prostate with PSA levels greater than 4
- 26 samples with prostate cancer with PSA levels 4 through 10
- 43 samples with prostate cancer with PSA levels greater than 10

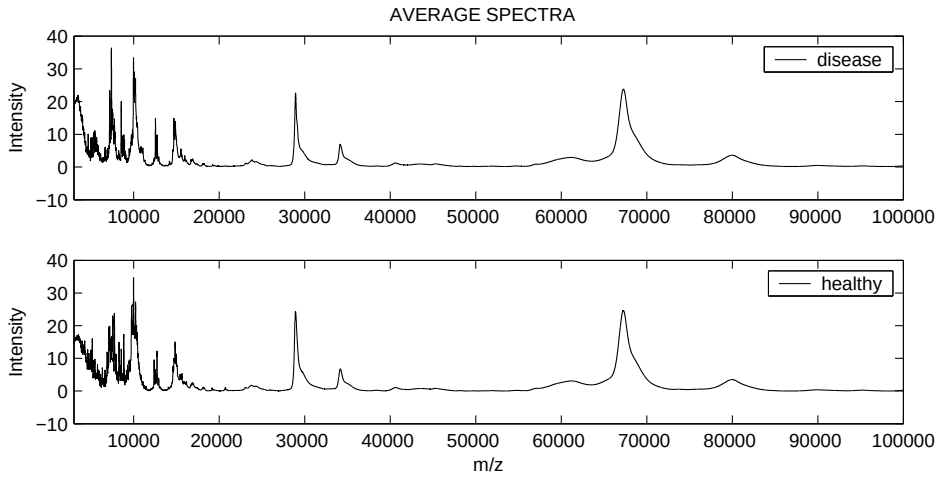


Figure 9: Average spectra from Wikström PP1 data set. High Laser Intensity and the IMAC30 Chip.

These serum samples were collected from men over age 50. Most of the 322 samples were collected in Santiago, Chile, in collaboration with the Catholic University of Chile. An additional 20 samples were collected at the National Cancer Institute and another 25 were obtained from the Simone Protective Cancer Institute Lawrenceville, NJ. In this study, the H4 (C16) hydrophobic interaction protein chip was used with the following specific settings for the SELDI system: Laser Intensity: 240, Detector Sensitivity: 10, Mass Focus: 6000, Position: 50, Molecular Mass Range of 0-20000 Daltons, 50-shot average per sample [12]. Figure 10 shows the average spectra according to the grouping in section *Implementation and testing*.

7.3 ADAM et al.

This dataset was downloaded from http://www.evms.edu/vpc/seldi/. It contains samples previously used in the work by the authors of [1]. The serum samples were obtained from the Virginia Prostate Center Tissue and Body Fluid Bank. The 327 different samples in this set were also divided into four separate categories:

- 81 samples with no evidence of disease and PSA level less than 4 ng/ml (mean 1,32)
- 78 samples with benign prostate with PSA levels between 4 and 10 (mean 4,60)
- 84 samples with organ-confined prostate cancer with PSA levels up to 95

(mean 10,10)
• 84 samples with non-organ-confined prostate cancer with PSA up to 8750 (mean 206.93)

In addition to the PSA level some other criterions were also used in the creation of the subsets: The non-disease patients were selected into the first subset if they had a normal digital rectal exam and no evidence of prostatic disease. A patient was selected into the benign group if they had gone through multiple negative biopsies. The blood samples from patients diagnosed with either prostate cancer or benign condition were procured from the department of Urology, Eastern Virginia Medical School. The samples from healthy patients came from free screening clinics open to the general public. The authors of [1] used the IMAC3 chip in their SELDI process. This chip is similar to the IMAC30 described above and is also a metal binding chip based on copper (Cu). The parameters of the SELDI system in this case was as follows: Laser Intensity: 220, Detector Sensitivity: 7, Focus lag time: 900 ns, Molecular Mass Range of 0-198000 Dalton and 192-shot average per chip [1]. Figure 11 shows the average spectra according to the grouping in section 8.

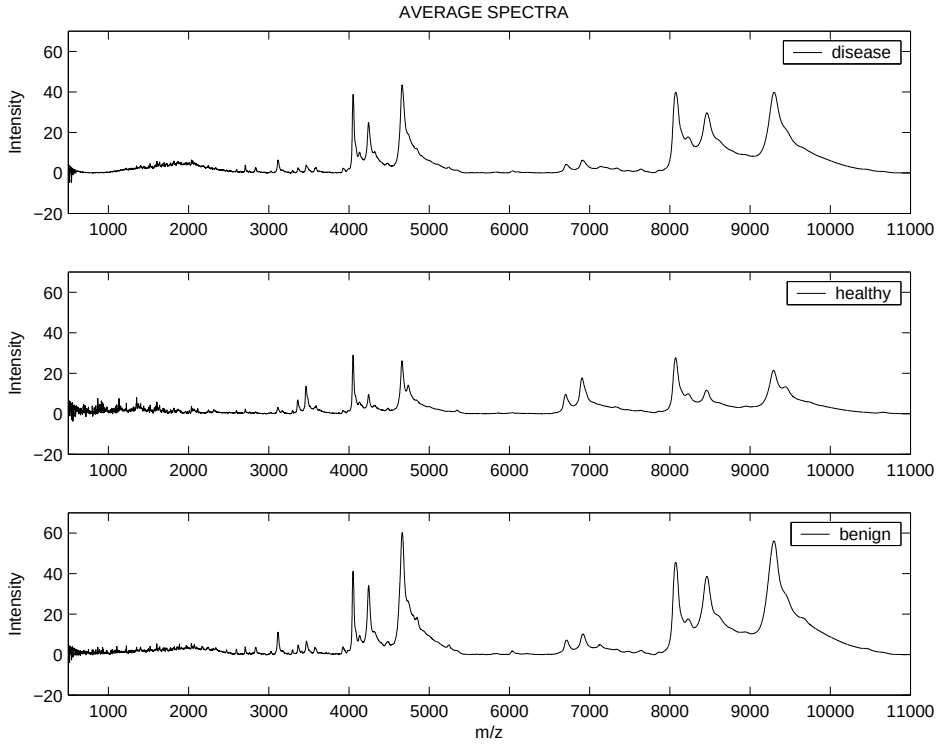


Figure 10: Average spectra from the Petricoin data set.

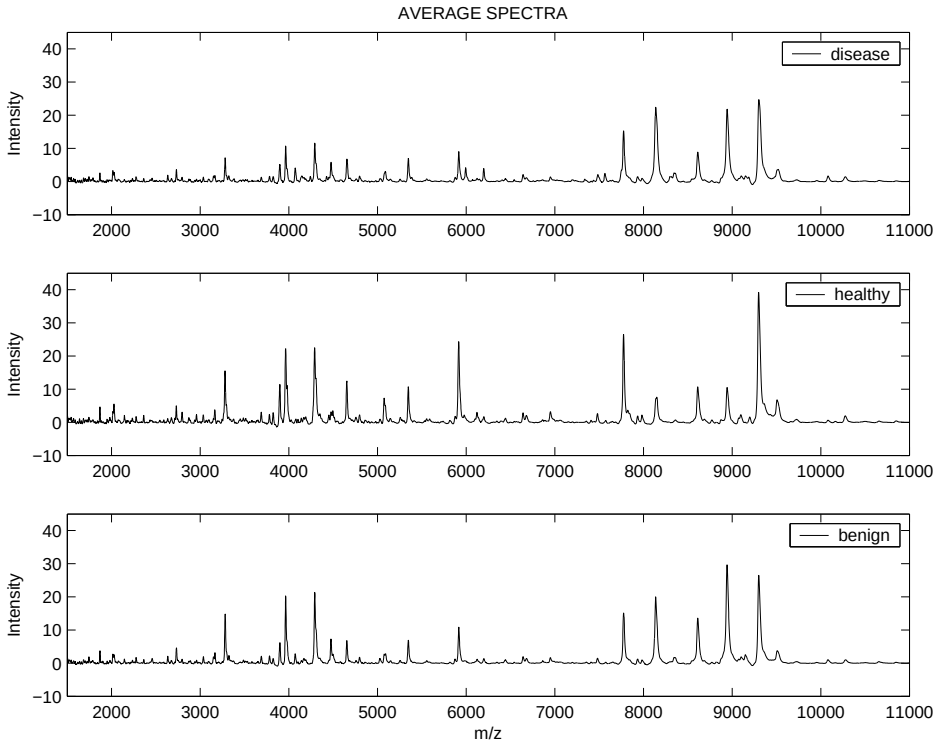


Figure 11: Average spectra from the Adam data set.

8 Implementation and Testing

This section includes a short description of the hardware and software used in the project together with information about how the samples in the different sets were categorized before the test runs. In addition, the section contains theory about *cross-validation* and specifies how the data were split into training and test sets. The latter depends on the classification method used in the separate test runs.

8.1 Equipment and Software

The dimensionality reduction algorithms and classification methods described in this report were implemented in MATLAB version 6.5 (Mathworks Inc). Some of the used code was downloaded from internet sites of other author [10, 2] but the main part of the programming work was done by the author of this report. The test runs were performed on a Pentium IV 2.6 GHz machine with operating system Windows XP equipped with 1 GB RAM.

8.2 Grouping of Samples

The samples set coming from the Department of Medical Bioscience, Pathology, Umeå University, (CAPS , PP1) consists of two subsets. These subsets are cancer and controls where the controls are presumed to come from healthy patients. The two subsets were named *disease* and *healthy* in the classification part.

The sample sets provided by the researchers of Petricoin et al. and Adam et al. both initially consisted of four subsets: healthy samples, samples with benign prostate condition and two sets with different types of prostate cancer. The two types of cancer sample sets were joined to form one single set called *disease* and the others were named *benign* and *healthy*. The classifiers used on the union of these three subsets classify the test samples into one of the three classes. For the Petricoin data set, 65 of the initial 190 benign samples were randomly selected to be used in the testruns. This was done to achieve an even number of samples for each category. In the Adam data set, 84 of the 168 cancer samples were randomly chosen for the same reason. See section 7 for more information about the separate sample sets.

8.3 Training and Test Sets

To get reliable and accurate results in a classification problem, it is important to split the set of observations into separate non-overlapping subsets. This is especially the case when the number of observations is limited and it is hard to obtain new ones. A common way is to create one training and one test set and then use the training set to estimate the parameters of the classification method and the test set to evaluate the classifier's performance. The reason for this separation is that we do not want information contained in the samples of

the test set to influence the parameters of the classifier. If this were the case, the classification of the samples in the test set would in most applications indicate better performance than if the test samples were completely unknown to the classifier. This is clearly not desirable. One way to separate the observations is *cross-validation* described below.

8.3.1 Cross-validation

In *cross-validation*, the n available observations in the set X are split into m separate subsets, each containing $\frac{n}{m}$ observations. The classification method is trained and tested m times, each time using one of the m subsets as the test set and all the remaining samples as the training set. The general performance of the classifier is estimated by evaluating the average performance of all the m test runs.

8.3.2 Training and Test Sets in this Project

All methods and algorithms described in this project with the exception of *Neural Nets*, uses a special form of cross-validation called *jack-knifing*. In jack-knifing, $m = n$ which means that the test set consists of only one sample at a time while all other data is used to train the classifier. This method is considered to produce an accurate estimation of the parameters but is more computationally complex than other commonly used methods [9].

8.4 Test Settings

Principal Component Analysis

When PCA was applied in combination with Bayesian Discriminant Functions, Fisher Linear Discriminants and K Nearest Neighbours, jack-knifing was used to divide the samples into training and test sets. A separate projection matrix was calculated for every training set and all samples were then projected to the d-dimensional space. All parameters of the classification methods were based only on information in the training sets.

Discrete Wavelet Transform

DWT with jack-knifing was used in combination with the same classification methods as PCA described above. Different number of decomposition steps (3-10) and wavelet coefficients (1-200) were tested and classification results were calculated for each setting. The importance of a wavelet coefficient was measured by how many samples s in the training set that had an absolute value above the 99^{th} percentile of that coefficient. The coefficient is considered more important if s is high. All parameters are based only on the training sets.

Independent Component Analysis

The ICA algorithm is computationally complex (in spite of the fact that the algorithm is called "Fast-ICA") and could not be run directly on the raw data in this application with the available hardware. The chosen solution was to first use discrete wavelet transform in 6 steps to reduce the dimensionality of the samples to 500 and then use ICA in a second reduction step to achieve a more manageable number of variables for classification. The maximum number of independent components was 60 for the Petricoin data set and 75 for the Adam data set. The DWT application used all samples to select the most important coefficients. This was done to reduce the time complexity of the problem and the introduced error is believed to be quite small (other tests have supported this theory). The only classification method that was tried in the ICA case was Bayesian Discriminant Functions. One reason for this was that the samples were projected with separate projection matrices for each class, a fact that made BDF the most suitable method in the classification step. Jack-knifing was used in the ICA case as well.

Neural Networks

Neural Networks were used in combination with both PCA and DWT in a series of separate test runs. In the PCA case, 60% of the samples formed the training set and the remaining 40% the test set. For DWT this ratio was set to 70/30. The separation into the two subsets was performed randomly before each test run. The projection matrices (PCA) and the wavelet coefficients (DWT) were calculated from information in the training set. The networks were trained in 50 epochs and after each epoch the classification performance was measured on both the training and test set. Different number of hidden nodes were tested to find the best setting for each data set and dimensionality reducer. The number of input nodes was chosen by studying the performance of other used classifiers (e.g. KNN, BDF) using the same reduction methods. The results presented in Appendix A are the average result of 50 different networks for each setting.

8.5 Proteins and Back-Projection

The Back-Projection algorithm was applied only to the cancer and healthy samples for each sample set. To evaluate the optimal number of principal components to use, the sets were first classified with PCA in combination with FLD. The best performance was achieved with 16 (Petricoin) and 92 (Adam) components respectively. To compensate for the lack of accuracy on the x-axis in the SELDI output and to avoid that the discovered proteins are highly collinear, a range limit of 30 was set. This means that all detected proteins have m/z variables that differs at least 30 units from all other proteins in the important set. In the classification step using only these proteins, the intensity values were treated as if they were independent by application of Bayesian discriminant functions. The goal of the back-projection algorithm is to detect the proteins and measure

the magnitude of the contained information not to make an exact evaluation of the classification performance. Jack-knifing was used during training of the Bayesian classifier.

8.6 Applied Algorithms

All different dimensionality reduction methods and classification algorithms were tested on the Petricoin and Adam data sets as described previously in this section. The accuracy of the data from Wikström et al. (CAPS and PP1) were more uncertain due to difficulties with the SELDI system and the pre-treatment of sample solutions. It wasn't decided until at a late stage in the project that the analysis of the data actually should be included in this report. Therefore, only the best reduction algorithm and classification methods were used on these two sets. The decision regarding the best algorithms was based on the results from the analysis of the Petricoin and Adam data sets. More about this issue can read in section *Discussion*. The Back-Projection algorithm was applied to all data sets except the CAPS sets with high laser intensity. These sets were excluded due to too much memory consumption during the test runs.

9 Summary of Tests

The results presented in this section are based mainly on the analysis of the Adam and Petricoin data sets. The data in the Wikström sets were more uncertain and all methods were not applied to this data. See section 10 for more information about the Wikström sets.

9.1 Classification Performance

The best overall classification performance for both the Adam and the Petricoin set was achieved using the combination of PCA and FLD. This combination seems to perform significantly better than PCA used together with other classification methods, for example KNN or BDF. To picture this, Figure 12 and 13 shows the classification performance $\pm$ one standard deviation of the error. The true distribution of the error is of course hard to obtain but Figure 12 and 13 clearly indicates that there is in fact a difference in performance between the methods. The best overall performance for the three classes is 0.970 for the Petricoin and 0.856 for the Adam data set. It is interesting to note that this difference does not exist when DWT is used instead of PCA for dimensionality reduction. In this case FLD is only slightly better than KNN and BDF for the Adam set while for the Petricoin set, KNN actually outperforms FLD. The performance level is 0.858 for FLD and 0.919 for KNN for the Petricoin set. One observation regarding the KNN method is that the optimal number of k seems to be 1 for nearly all different reduction methods and data sets.

Applying the data sets to neural networks strongly indicates that neural nets are not the most suitable method for this type of application, at least when the total number of samples is limited to about 200. For the Petricoin set the classification performance was acceptable, almost reaching the level of 0.8 for both the reduction methods PCA and DWT. The results for the Adam set were much worse, about 0.6 together with PCA and a very poor 0.4 used in combination with DWT. Section *Discussion* includes thoughts and comments around the lack of performance for the neural net approach.

The reduction method ICA performed significantly worse than PCA and DWT. ICA together with BDF gave an overall performance of 0.726 for the Petricoin set but the sensitivity was as low as 0.536 in this case. For the Adam set, ICA and BDF performed only slightly better than blind guessing. Other major disadvantages with ICA is that it can not be applied directly on the initial spectra due to the complexity and that it is stochastic, not yielding the same results every time it is used on the same data.

In general, all performance results from test runs with the Wikström data sets are at a lower level than for the other sets. It is important to note that the Wikström sets includes only two categories of samples instead of three which would normally be a much easier classification task, that is if the quality of the

data had been the same as for the other sets. The best performance for the CAPS set was 0.704, achieved with the IMAC30 protein chip using low laser intensity. PCA was used for dimensionality reduction and FLD for classification. The best result for the PP1 set was 0.820 with PCA and KNN. The IMAC30 chip was used in this case as well but with high laser intensity.

Appendix A includes complete result tables and plots for all methods applied to the data sets listed in this report.

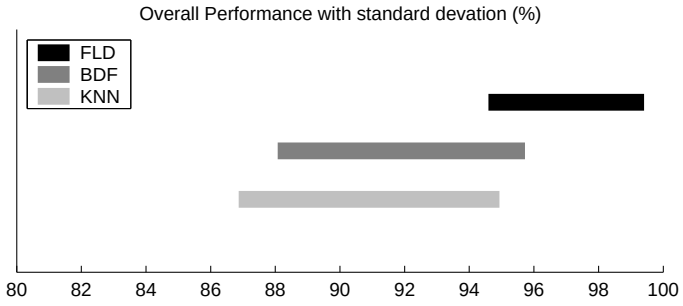


Figure 12: Classification performance ± one standard deviation for the Petricoin data set. Dimensionality reduction: PCA.

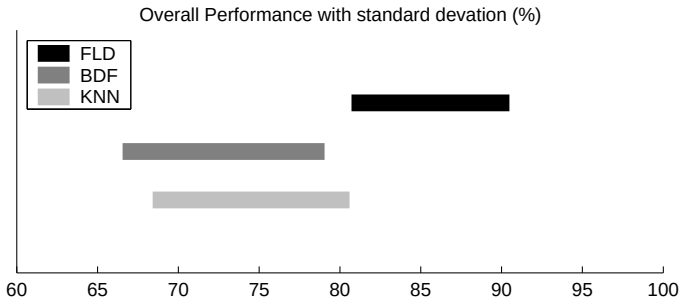


Figure 13: Classification performance ± one standard deviation for the Adam data set. Dimensionality reduction: PCA.

9.2 Important Proteins

Appendix B.1 contains a list of the ten most important proteins for each data set. It is obvious that the identities of the detected proteins vary between the data sets. These identities depend highly on the chip technology used in the SELDI-process. For example, a comparison between the list of proteins for the Petricoin and Adam set shows only two specific proteins that possibly appear in

both lists. This is the proteins with mass/charge values ≈ 4650 and ≈ 9300. It is also clear that a few proteins hold most of the information deciding the class belonging of the samples.

Appendix B.2 shows that using only the five best proteins, the performance of separating cancer from healthy patients is 0.932 for the Petricoin set. The corresponding result for the six best proteins in the Adam case is 0.818. BDF has been applied in these approximations. If the whole spectrum is used, the classification performance for the sets is 1 and 0.915 respectively as can be seen on the last row of the tables. The protein lists from the Wikström data sets is assumed to bee too uncertain to draw reliable conclusions. Looking at appendix B.2, the information content in these proteins appears very uneven and many proteins classify nearly all samples as belonging to the same class.

10 Conclusions and Discussion

Algorithms

The most promising algorithms in this work have been Principal Component Analysis used in combination with Fisher Linear Discriminant. PCA is a well known and often applied technique for various types of problems while the FLD perhaps is not so commonly used by researchers in this and similar research areas. FLD is a powerful tool for the separation of observations into categories. Figure 14 shows FLD used on all 197 samples in the Petricoin data set after reduction to 106 principal components. From the picture it might seem like all samples would be correctly classified but the impact of each sample is so strong that the use of the jack-knifing technique will rotate the directions of the model causing 6 of 197 samples to be misclassified. Figure 14 gives an example of what the method is capable of however.

Neural Networks did not perform as well as the other classification methods in this application. One probable reason for this could be that the number of parameters needed to build an accurate model is too large in comparison to the number of available samples. There are simply too many weights that must be set when the network for example contains 80 input, 30 hidden and 3 output nodes. Tests using only one output node separating two classes instead of three supports this theory. The performance appeared to be much better in the two-class case.

Combining Classifiers

One question that arises when studying the performance of the different methods is whether it is the same samples that are misclassified with every method. If this it not the case, could a combination of classifiers applied on the data sets improve the performance? To investigate this issue, the six methods that used jack-knifing were combined and applied on the three data sets. These methods were PCA and DWT, each in combination with BDF, FLD and KNN. All six classifiers individually were used on the samples (one at a time) and the result of the majority of the classifiers was interpreted as the combined result. In the case of a draw, the best individual classifier made the decision.

For the Petricoin data set, the performance did not improve when combining the classifiers. One reason for this is of course the fact that the performance using PCA and FLD was very high to begin with. The results for the Adam data set were more successful. The performance measures for all three classes increased compared to the best single classifier when the combination of classifiers was used. Table 1 shows the results. One interesting observation was that no sample of the 197 in the Petricoin set and only one of 243 in the Adam set was misclassified by all six classifiers (a benign sample that all classifiers inter-

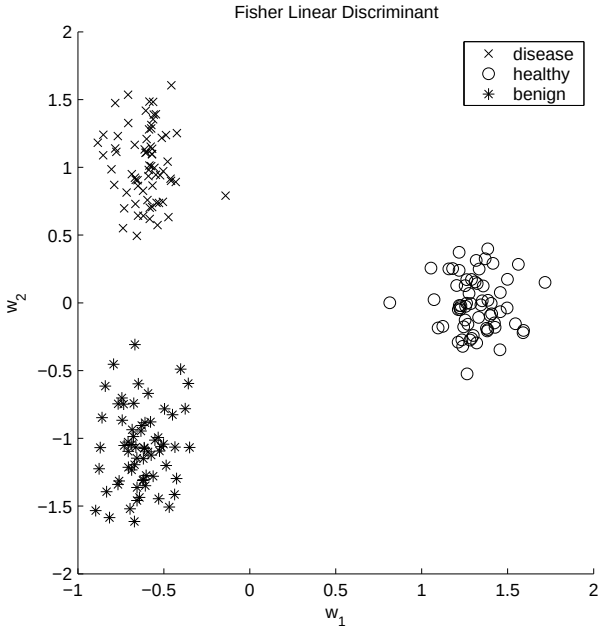


Figure 14: Fisher Linear Discriminat applied on all samples in the Petricoin set.

preted as disease). This indicates that a more sophisticated way of combining classifiers could be developed to increase the performance further (beyond the scope of this report).

Table 1: Combined performance for the Adam data set.

classification method	sensitivity	specificity	correct benign	overall performance
PCA/FLD	0.726	0.963	0.885	0.856
Combined voting	0.785	0.988	0.910	0.893

Classification Errors

Another interesting issue is to study the predicted class of the misclassified samples in relation to the true class belonging. In the Adam data set, only 1 of 18 misclassified cancer samples and 1 of 7 misclassified benign samples was predicted to be healthy. For the Petricoin set, the corresponding ratios were 1 of 2 and 1 of 4 respectively. Obviously, the proposed algorithms used on these sets have most difficulty in separating cancer and benign samples while the healthy samples are identified almost perfectly.

Different Types of Spectra

Looking at figures 10 and 11 showing the average spectra for the Petricoin and Adam data set, it is obvious that the intensity peaks in the Petricoin set are quite different from those in the Adam set. The peaks in the Petricoin set are much wider and this means that a larger part of the mass/charge variables holds important information about the class belonging in this set than in the Adam set. This could be a reason why the classification performance is generally better for the Petricoin set. The appearance of the Adam spectra has the advantage that it is easier to pinpoint the specific proteins corresponding to the peaks by just looking at the spectra. This can be a desirable property in situations where the main goal is to identify potential biomarkers. The appearance and positions of the peaks depend among other things on the chip technology used and the laser energy applied in the SELDI process.

Many authors in the referenced work have suggested that the initial part of the spectra ($\approx 0-2000$ Dalton) should be removed before the analysis, the reason being that it usually contains a lot of noise. The list of the most important proteins in the Petricoin set contradicts this assumption. Four of the ten most important proteins listed in appendix B.1 comes from this particular area. It is likely that the selected type of protein chip influences the information content in this part of the spectra.

Problems with the Wikström sets

The data sets from Wikström et al. (CAPS and PP1) were considered to be uncertain due to difficulties with the SELDI system and the pre-treatment of sample solutions. The coefficient of variation was high for these data sets and the resulting spectra clearly contained a lot of noise. This is a well known problem with the SELDI technology. Another issue was that the sample solutions in the CAPS set had been handled in a way that caused some proteins to degenerate to different degrees in the separate samples. This was of course another source of uncertainty for this particular set.

Another fact that probably contributes to the low classification performance for these sets is the grouping of the samples. Unlike the Adam and Petricoin data sets, the cancer groups in CAPS and PP1 included very few samples with non-organ-confined cancer. These types of samples are in general easier to classify correctly than samples with organ-confined cancer since their spectra are more dissimilar to spectra coming from healthy patients. In addition, the healthy groups are likely to contain both benign and undetected cancer samples and this of course makes it more difficult to build an accurate model, especially since benign spectra appear to be more similar to cancer spectra than to healthy in the other data sets used.

The researchers at the Department of Medical Bioscience, Umeå University,

have been trying to determine the best system settings, sample treatment and chip technology to achieve more reliable results. The intention was to produce new data sets of higher quality and to analyse these instead of the CAPS and PP1 sets in this project. These new data sets were not ready in time however and could not be included in this work. It was then decided that the CAPS and PP1 sets should actually be analysed and that this report should contain the results of the analysis.

Final words

This work have showed that the SELDI process in combination with dimensionality reduction methods and classification algorithms is an effective way to make highly reliable prognostications of patients possibly suffering from prostate cancer. The SELDI technology can be used to produce the same types of spectra from patients with other forms of cancer, e.g. ovarian and breast cancer and the different methods described in this report can be applied on these types of spectra as well.

Acknowledgements

I would like to thank my supervisor at the Department of Computing Science, Fredrik Georgsson, for many practical tips during this master thesis project and for interesting discussions around the methods and results presented in this work and others. I thank Pernilla Wikström for introducing me to the working principle and difficulties of the SELDI system and for providing me with articles and data related to the contents of this report. Finally thanks also to my fiancé Malin for her support.

References

[1] Bao-Ling Adam, Yinsheng Qu, John W. Davis, Michael D. Ward, Mary Ann Clements, Lisa H. Cazares, O. John Semmens, Paul F. Schnellhammer, Yutaka Yasui, Ziding Feng, and George L. Wright Jr. Serum protein fingerprinting coupled with a pattern-matching algorithm distinguishes prostate cancer from benign prostate hyperplasia and healthy men. *Cancer Research*, 62:3609–3614, July 2002.

[2] A.Hyvärinen and E.Oja. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Tr. on Neural Networks*, 10(3):626–634, 1999.

[3] Issac Alphonso. Independent component analysis. *ECE 8990 - Special Topics in ECE Pattern Recognition*, March 2001.

[4] G. Ball, S. Mian, F. Holding, R. O. Allibone, J. Lowe, S. Ali, G. Li, S. McCardle, I. O. Ellis, C. Creaser, and R. C. Rees. An integrated approach utilizing artificial neural networks and seldi mass spectrometry for the classification of human tumours and rapid identification of potential biomarkers. *Bioinformatics*, 18(3):395–404, 2002.

[5] Asa Ben-Hur, Olivier Chapelle, Rene´ Doursat, Andre´Elisseeff, Isabelle Guyon, and Jason Weston. Application of support vector machines to the classification of proteinchip system mass spectral data of prostate cancer serum. Technical report, BIOwulf Technologies, 2001. BIOwulf Technologies Berkeley New York Savannah.

[6] Marco Bressan, David Guillamet, and Jordi Vitria. Using an ica representation of high dimensional data for object recognition and classification. *ISBN 0-7695-1272-0/01*, 2001. Centre de Visio per Computador (CVC), Universitat Autonoma de Barcelona.

[7] Mark Carpenter, Sreelatha Melath, Sijian Zhang, and William E. Grizzle. Statistical processing and analysis of proteomic and genomic data. *Statistics & Pharmacokinetics*, (106), 2003. Medical Statistics Section and Department of Pathology, University of Alabama-Birmingham, Birmingham, Alabama.

[8] Kevin R. Coombes, Herbert A. Fritsche Jr., Charlotte Clarke, Jeng-Neng Chen, Keith A. Baggerly, Jeffery S. Morris, Lian-Chun Xiao, Mien-Chie Hung, and Henry M. Kuerer. Quality control and peak finding for proteomics data collected from nipple aspirate fluid by surface-enhanced laser desorption and ionization. *Clinical Chemistry*, 49(10):1615–1623, 2003.

[9] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. ISBN 0-471-05669-3. John Wiley & Sons, Inc., second edition, 2001.

[10] Ryan H, Lilien Hany Farid, and Bruce R. Donald. Probabilistic disease classification of expression-dependent proteomic data from mass spectrometry of human serum. *Journal of Computational Biology*, February 2003.

[11] http://home.od.ua/ relayer/algo/dsp/surfingw/wbasic. Wavelet basics. February 2004.

[12] http://ncifdaproteomics.com/. Proteomic analysis, February 2004.

[13] http://www.evms.edu/vpc/seldi/seldiprocess/index.html. Overview of the seldi system, April 2004.

[14] Emanuel F. Petricoin III, Ali M Ardekani, Ben A. Hitt, Peter J Levine, Vincent A Fusaro, Seth M. Steinberg, Gordon B Mills, Charles Simone, David A Fishman, Elise C Kohn, and Lance A Liotta. Use of proteomic patterns in serum to identify ovarian cancer. *The Lancet*, 359:572–577, February 2002.

[15] S. Lehrer, J. Roboz an H. Ding, S. Zhao, E.J. Diamond, J.F. Holland, N.N. Stone, M.J. Droller, and R.G. Stock. Putative protein markers in the sera of men with prostatic neoplasms. *British Journal of Urology*, 92:223–225, April 2003. Department of Radiation Oncology.

[16] L.Eriksson, E.Johansson, N.Kettaneh-Wold, and S.Wold. *Introduction to Multi- and Megavariate Data Analysis using Projection Methods (PCA and PLS).* UMETRICS AB, UMETRICS AB Box 7960 SE90719 Umeå, June 1999.

[17] Jinong Li, Zhen Zhang, Jason Rosenzweig, Young Y. Wang, and Daniel W. Chan. Proteomics and bioinformatics approaches for identification of serum biomarkers to detect breast cancer. *Clinical Chemistry*, 48(8):1296–1304, 2002.

[18] M.Lennon, G.Mercier, M.C.Mouchot, and L.Hubert-Moy. Independent component analysis as a tool for the dimensionality reduction and the representation of hyperspectral images. *International Geoscience And Remote Sensing Symposium*, July 2001. Ecole Nationale Superieure des Telecommunications de Bretagne - Department ITI (France).

[19] Emanuel F. Petricoin, David K. Ornstein, Cloud P. Paweletz, Ali Ardekani, Paul S. Hackett, Ben A. Hitt, Alfredo Velassco, Christian Trucco, Laura Wiegand, Kamillah Wood, Charles B. Simone, Peter J. Levine, W. Marston Linehan, Michael R. Emmert-Buck, Seth M. Steinberg, Elise C. Kohn, and Lance A. Liotta. Serum proteomic patterns for detection of prostate cancer. *Journal of the National Cancer Institute*, 94(20):1576–1578, October 2002.

[20] Yinsheng Qu, Bao-Ling Adam, Mark Thornquist, John D. Potter, Mary Lou Thompson, Yutaka Yasui, John Davies, Paul F. Schnellhammer, Lisa Cazares, MaryAnn Clements, George L. Wright Jr., and Ziding Feng. Data reduction using a discrete wavelet transform in discriminant analysis of very high dimensionality data. *Biometrics*, 59:143–151, March 2003.

[21] Yinsheng Qu, Bao-Ling Adam, Yutaka Yasui, Michael D. Ward, Liza H. Cazares, Paul F. Schellhammer, Ziding Feng, O. John Semmes, and George L. Wright JR. Boosted decision tree analysis of surface-enhanced laser desorption/ionization mass spectral serum profiles discriminates prostate cancer from noncancer patients. *Clinical Chemistry*, 48(10):1835–1843, July 2002.

[22] C. Valens. A really friendly guide to wavelets. Technical report, July 1999.

[23] Yutaka Yasui, Margaret Pepe, Mary Lou Thompson, Bao-Ling Adam, George L. Wright Jr., Yinsheng Qu, John D. Potter, Marcy Winget, Mark Thornquist, and Ziding Feng. A data-analytic strategy for protein biomarker discovery: profiling of high-dimensional proteomic data for cancer detection. *Biostatistics*, 4(3):449–463, 2003.

[24] Tatyana A. Zhukov, Roy A. Johanson, Alan B. Cantor, Robert A. Clark, and Melvyn S. Tockman. Discovery of distiction protein profiles specific fot lung tumors and pre-malignant lung lesions by seldi mass spectrometry. *Lung Cancer*, 40:267–279, January 2003.

Appendices

A Classification Results

A.1 Petricoin Data Set

A.1.1 Principal Component Analysis

classification method	principal components	sensitivity	specificity healthy	specificity benign	overall performance
BDF	38	0.899	0.952	0.908	0.919
FLD	106	0.971	1	0.938	0.970
KNN 1	19	0.884	0.968	0.877	0.909
KNN 3	44	0.942	0.952	0.831	0.909
KNN 5	14	0.870	0.968	0.831	0.888
KNN 7	38	0.957	0.952	0.785	0.898
KNN 9	32	0.928	0.968	0.738	0.878
KNN 11	27	0.928	0.968	0.754	0.883
KNN 13	35	0.942	0.968	0.723	0.878
KNN 15	32	0.957	0.952	0.723	0.878

NEURAL NETWORKS						
principal components	hidden nodes	number of epochs	sensitivity	specificity healthy	specificity benign	overall performance
30	15	12	0.735	0.888	0.695	0.764
30	20	46	0.712	0.888	0.733	0.767
30	25	43	0.751	0.863	0.765	0.786
30	30	49	0.766	0.875	0.713	0.780
30	35	47	0.757	0.869	0.680	0.763

A.1.2 Discrete Wavelet Transform

classification method	decomp. steps	wavelet coeff.	sensitivity	specificity healthy	specificity benign	overall performance
BDF	7	43	0.826	0.952	0.708	0.827
FLD	6	59	0.855	0.921	0.800	0.858
KNN 1	7	171	0.971	0.905	0.877	0.919
KNN 3	8	169	1	0.937	0.800	0.914
KNN 5	6	90	1	0.921	0.785	0.888
KNN 7	5	182	1	0.905	0.738	0.883
KNN 9	5	171	1	0.921	0.692	0.873
KNN 11	9	128	1	0.905	0.662	0.858
KNN 13	7	53	0.957	0.921	0.677	0.853
KNN 15	5	114	1	0.857	0.646	0.838

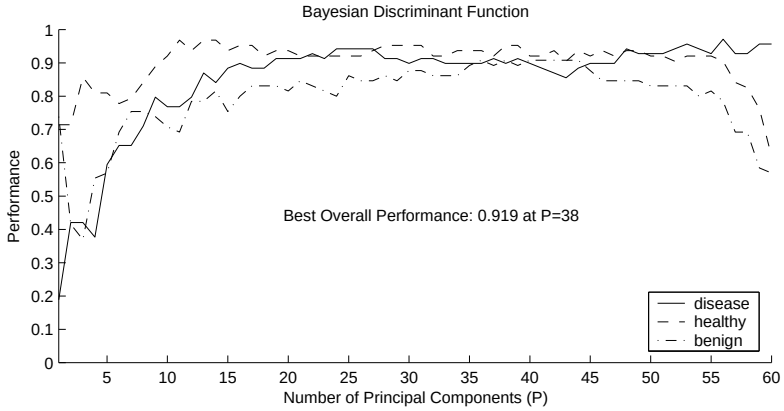
NEURAL NETWORKS						
wavelet coefficients	hidden nodes	number of epochs	sensitivity	specificity healthy	specificity benign	overall performance
80	10	15	0.740	0.786	0.716	0.743
80	15	27	0.797	0.849	0.740	0.792
80	20	31	0.766	0.820	0.811	0.793
80	25	31	0.745	0.834	0.749	0.768
80	30	38	0.742	0.786	0.704	0.741

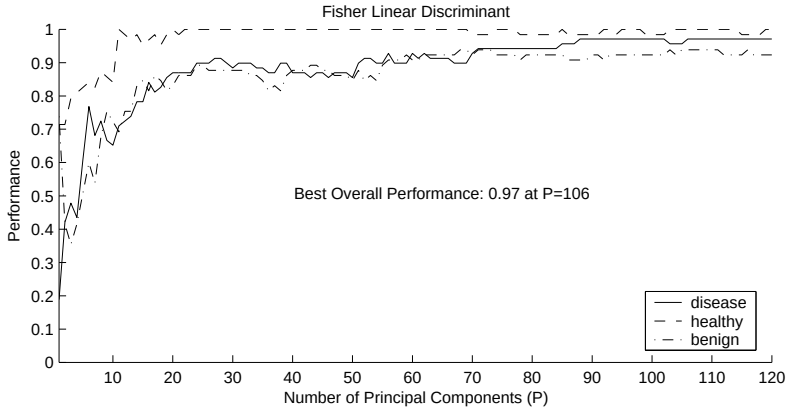
A.1.3 Independent Component Analysis

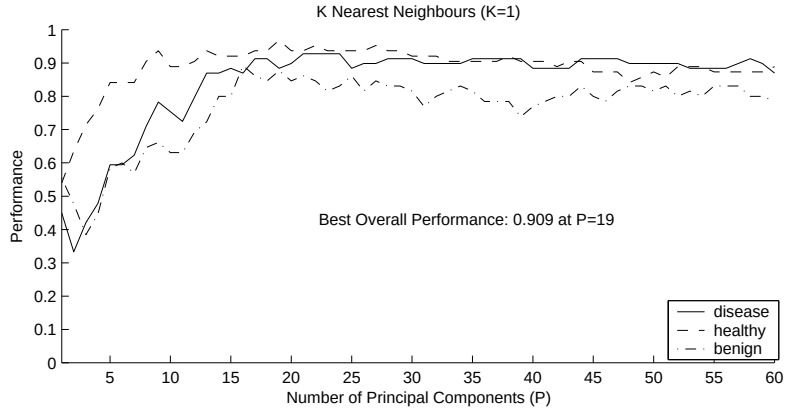
Pretreatment of data: DWT with 6 steps and 500 coefficients
Classification Method: Bayesian Discriminant Functions

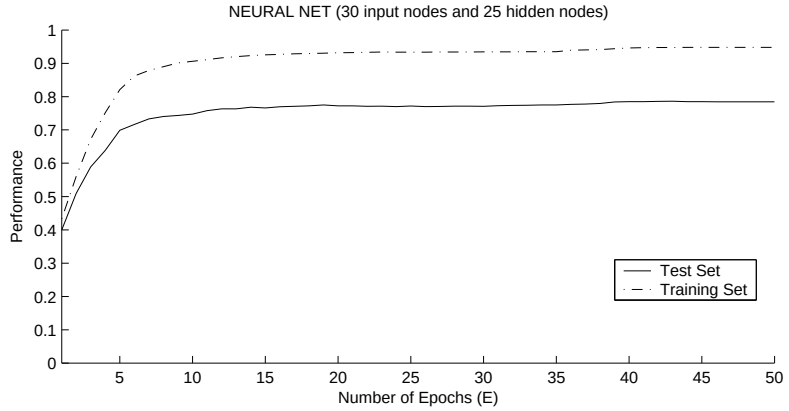
independent components	sensitivity	specificity healthy	specificity benign	overall performance
1	0.551	0.159	0.385	0.371
5	0.435	0.651	0.338	0.472
10	0.464	0.667	0.585	0.569
15	0.333	0.683	0.646	0.548
20	0.464	0.698	0.600	0.584
25	0.536	0.794	0.677	0.665
30	0.478	0.810	0.738	0.670
35	0.522	0.762	0.677	0.650
40	0.420	0.810	0.677	0.629
45	0.507	0.857	0.708	0.685
50	0.449	0.841	0.754	0.675
55	0.536	0.857	0.800	0.726
60	0.464	0.905	0.769	0.706

A.1.4 Plots PCA

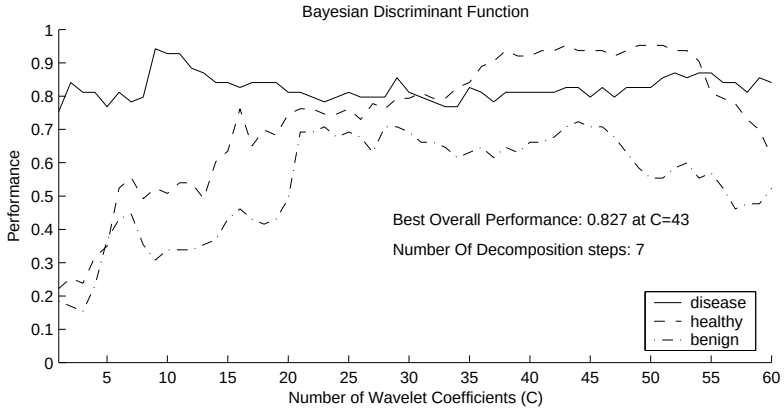


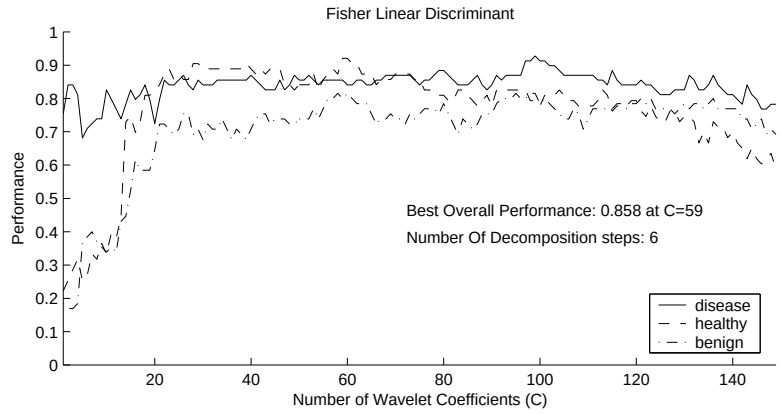

Fisher Linear Discriminant
Performance
Number of Principal Components (P)
Best Overall Performance: 0.97 at P=106
disease
healthy
benign

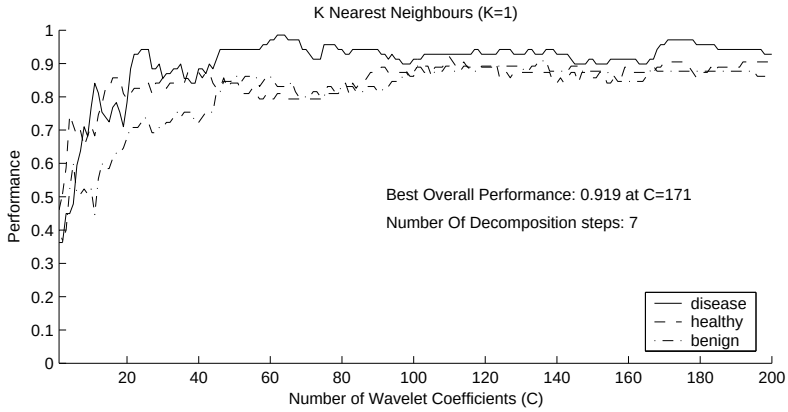

K Nearest Neighbours (K=1)
Performance
Number of Principal Components (P)
Best Overall Performance: 0.909 at P=19
disease
healthy
benign

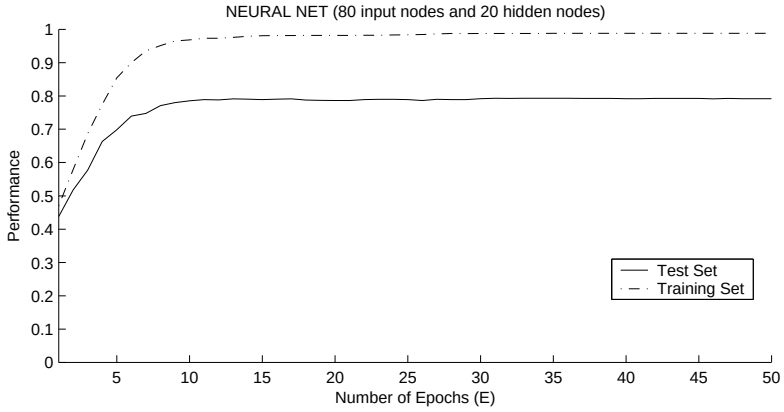

NEURAL NET (30 input nodes and 25 hidden nodes)
Performance
Number of Epochs (E)
Test Set
Training Set

A.1.5 Plots DWT








NEURAL NET (80 input nodes and 20 hidden nodes)
Performance
Number of Epochs (E)
Test Set
Training Set
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
5
10
15
20
25
30
35
40
45
50

A.2 Adam Data Set

A.2.1 Principal Component Analysis

classification method	principal components	sensitivity	specificity healthy	specificity benign	overall performance
BDF	21	0.405	0.963	0.833	0.728
FLD	170	0.726	0.963	0.885	0.856
KNN 1	45	0.667	0.790	0.782	0.745
KNN 3	18	0.583	0.889	0.731	0.733
KNN 5	19	0.548	0.840	0.808	0.728
KNN 7	20	0.524	0.840	0.795	0.716
KNN 9	19	0.488	0.852	0.808	0.712
KNN 11	16	0.524	0.815	0.795	0.708
KNN 13	16	0.488	0.815	0.808	0.700
KNN 15	20	0.440	0.790	0.808	0.675

NEURAL NETWORKS						
principal components	hidden nodes	number of epochs	sensitivity	specificity healthy	specificity benign	overall performance
30	15	7	0.540	0.696	0.580	0.603
30	20	30	0.508	0.685	0.541	0.576
30	25	30	0.507	0.689	0.554	0.581
30	30	12	0.469	0.753	0.611	0.603
30	35	19	0.517	0.706	0.587	0.599

A.2.2 Discrete Wavelet Transform

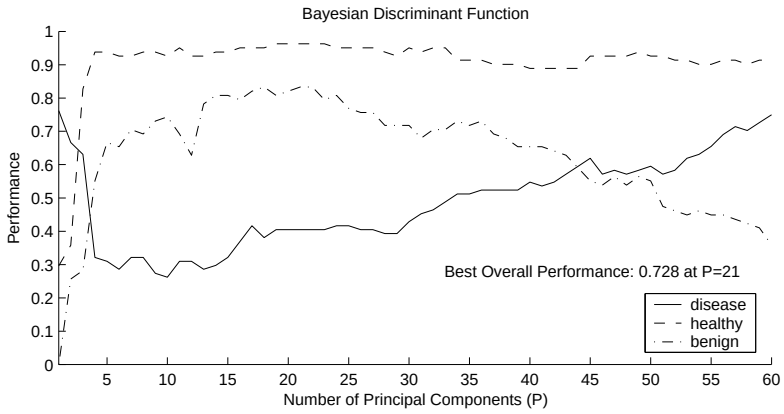
classification method	decomp. steps	wavelet coeff.	sensitivity	specificity healthy	specificity benign	overall performance
BDF	4	56	0.905	0.840	0.513	0.757
FLD	8	90	0.726	0.889	0.756	0.790
KNN 1	9	117	0.643	0.914	0.782	0.778
KNN 3	5	198	0.548	0.914	0.846	0.765
KNN 5	9	88	0.583	0.926	0.731	0.745
KNN 7	6	151	0.571	0.901	0.795	0.753
KNN 9	6	111	0.524	0.926	0.795	0.745
KNN 11	6	66	0.571	0.889	0.705	0.720
KNN 13	5	159	0.452	0.840	0.859	0.712
KNN 15	10	120	0.417	0.914	0.846	0.720

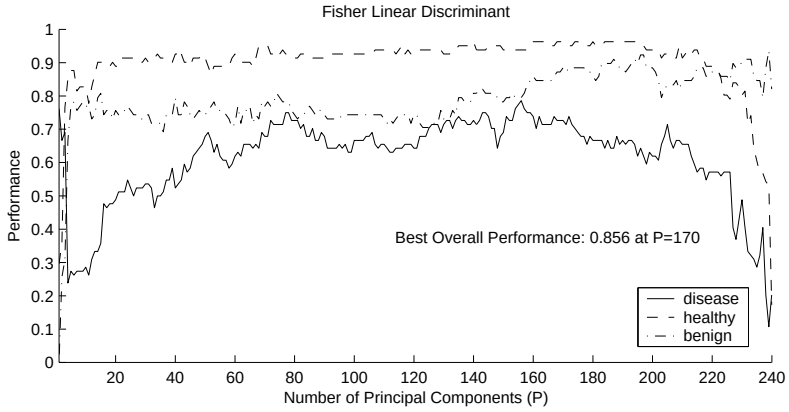
NEURAL NETWORKS						
wavelet coefficients	hidden nodes	number of epochs	sensitivity	specificity healthy	specificity benign	overall performance
80	10	48	0.372	0.380	0.423	0.386
80	15	14	0.385	0.404	0.403	0.396
80	20	19	0.392	0.360	0.400	0.383
80	25	10	0.378	0.400	0.420	0.395
80	30	8	0.387	0.382	0.453	0.400

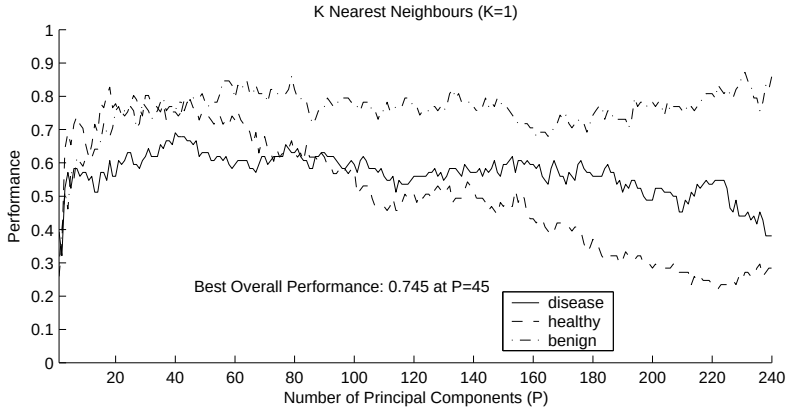
A.2.3 Independent Component Analysis

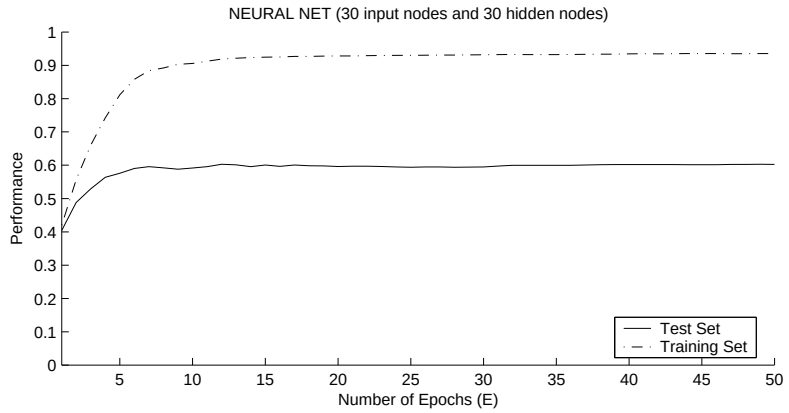
independent components	sensitivity	specificity healthy	specificity benign	overall performance
1	0.512	0.259	0.051	0.280
5	0.417	0.519	0.205	0.383
10	0.274	0.630	0.128	0.346
15	0.298	0.605	0.141	0.350
20	0.238	0.667	0.205	0.370
25	0.298	0.679	0.141	0.374
30	0.238	0.704	0.141	0.362
35	0.202	0.765	0.128	0.366
40	0.214	0.889	0.154	0.420
45	0.190	0.778	0.115	0.362
50	0.202	0.790	0.103	0.366
55	0.155	0.840	0.077	0.358
60	0.131	0.840	0.090	0.354
65	0.167	0.827	0.141	0.379
70	0.143	0.802	0.077	0.342
75	0.107	0.877	0.103	0.362

A.2.4 Plots PCA

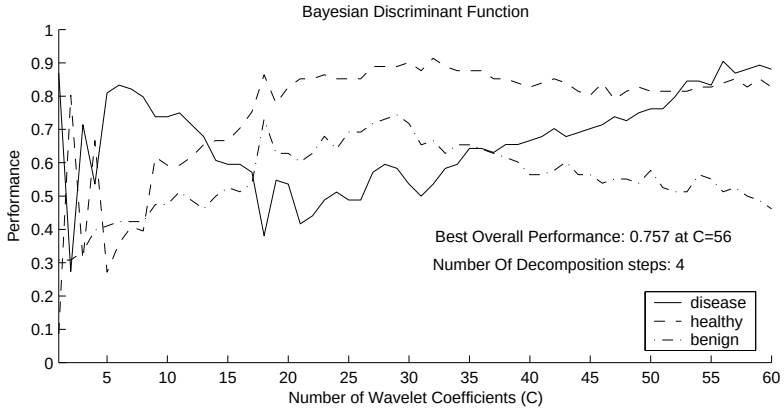


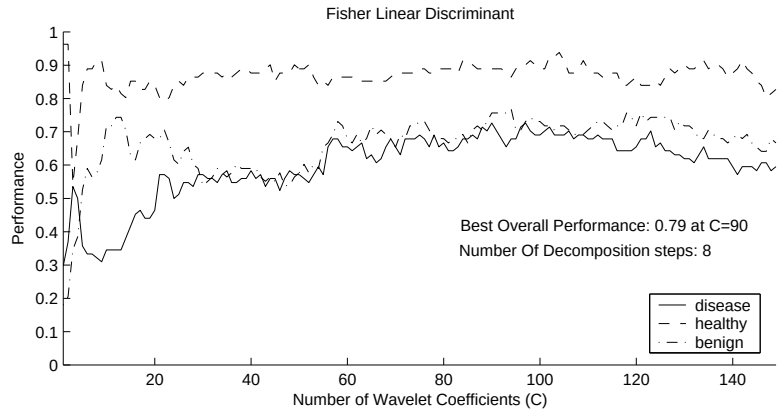

Fisher Linear Discriminant
Performance
Number of Principal Components (P)
Best Overall Performance: 0.856 at P=170
disease
healthy
benign

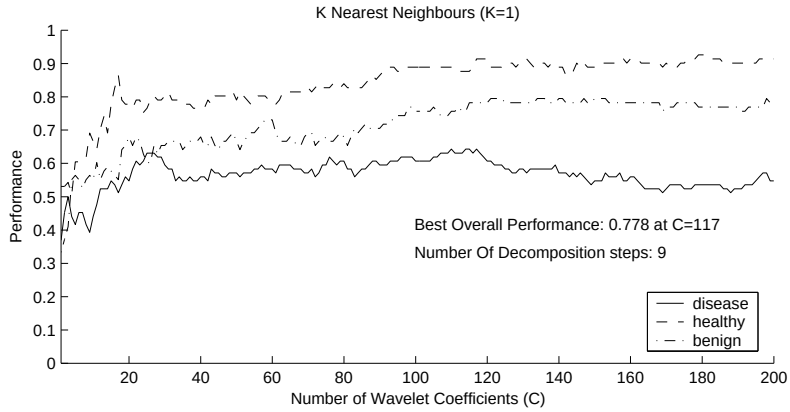

K Nearest Neighbours (K=1)
Performance
Number of Principal Components (P)
Best Overall Performance: 0.745 at P=45
disease
healthy
benign

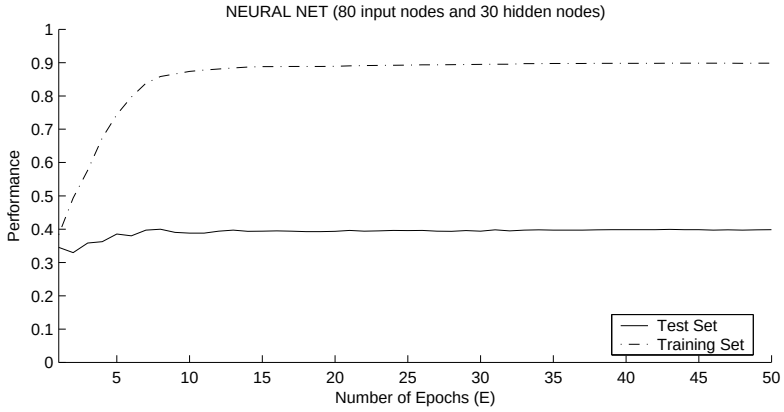

NEURAL NET (30 input nodes and 30 hidden nodes)
Performance
Number of Epochs (E)
Test Set
Training Set

A.2.5 Plots DWT








NEURAL NET (80 input nodes and 30 hidden nodes)
Performance
Number of Epochs (E)
Test Set
Training Set
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
5
10
15
20
25
30
35
40
45
50

A.3 Wikström Data Set - PP1

A.3.1 Principal Component Analysis

Chip: IMAC30 Laser Intensity: high				
classification method	principal components	sensitivity	specificity	overall performance
BDF	17	0.680	0.760	0.720
FLD	27	0.720	0.840	0.780
KNN 1	41	0.960	0.680	0.820
KNN 3	3	0.800	0.560	0.680
KNN 5	18	0.960	0.400	0.680
KNN 7	4	0.920	0.440	0.680

Chip: IMAC30 Laser Intensity: low				
classification method	principal components	sensitivity	specificity	overall performance
BDF	11	0.880	0.680	0.780
FLD	25	0.800	0.720	0.760
KNN 1	7	0.640	0.720	0.680
KNN 3	22	1	0.360	0.680
KNN 5	21	1	0.280	0.640
KNN 7	9	0.880	0.400	0.640

Chip: CM10 Laser Intensity: high				
classification method	principal components	sensitivity	specificity	overall performance
BDF	31	0.640	0.680	0.660
FLD	42	0.680	0.720	0.700
KNN 1	9	0.640	0.640	0.640
KNN 3	4	0.640	0.600	0.620
KNN 5	30	0.960	0.200	0.580
KNN 7	3	0.720	0.400	0.560

Chip: CM10 Laser Intensity: low				
classification method	principal components	sensitivity	specificity	overall performance
BDF	4	0.880	0.600	0.740
FLD	6	0.880	0.440	0.660
KNN 1	7	0.680	0.680	0.680
KNN 3	5	0.760	0.520	0.640
KNN 5	5	0.840	0.400	0.620
KNN 7	3	0.640	0.520	0.580

A.4 Wikström Data Set - CAPS

A.4.1 Principal Component Analysis

Chip: IMAC30 Laser Intensity: high				
classification method	principal components	sensitivity	specificity	overall performance
BDF	4	0.640	0.687	0.663
FLD	14	0.640	0.687	0.663
KNN 1	10	0.590	0.626	0.608
KNN 3	21	0.611	0.606	0.608
KNN 5	21	0.680	0.596	0.638
KNN 7	8	0.629	0.626	0.628

Chip: IMAC30 Laser Intensity: low				
classification method	principal components	sensitivity	specificity	overall performance
BDF	40	0.750	0.525	0.638
FLD	95	0.680	0.727	0.704
KNN 1	43	0.730	0.646	0.688
KNN 3	37	0.750	0.495	0.623
KNN 5	10	0.650	0.586	0.618
KNN 7	1	0.740	0.505	0.623

Chip: CM10 Laser Intensity: high				
classification method	principal components	sensitivity	specificity	overall performance
BDF	13	0.549	0.616	0.583
FLD	2	0.580	0.646	0.613
KNN 1	9	0.540	0.646	0.593
KNN 3	5	0.591	0.586	0.588
KNN 5	7	0.611	0.586	0.598
KNN 7	5	0.592	0.566	0.578

Chip: CM10 Laser Intensity: low				
classification method	principal components	sensitivity	specificity	overall performance
BDF	52	0.580	0.629	0.604
FLD	13	0.710	0.577	0.645
KNN 1	22	0.610	0.629	0.619
KNN 3	187	0.510	0.711	0.609
KNN 5	4	0.640	0.660	0.650
KNN 7	4	0.640	0.608	0.624

B Proteins

B.1 Most Important Proteins

Dataset:	Petricoin	Adam	Wikström PP1				Wikström CAPS	
Chip:	H4	IMAC3	IMAC30		CM10		IMAC30	CM10
Protein no.			low	high	low	high	low	low
1	21	9301	4289	9774	6630	11072	3963	9289
2	116	8135	3978	9743	3167	11645	4287	4650
3	317	8943	6630	10070	4170	12024	3283	3902
4	6907	5906	4423	9930	3909	11138	2671	8991
5	9308	4304	3281	10039	3325	11103	3038	8602
6	9278	7769	3324	9805	9283	12082	8934	4158
7	9339	3279	7763	9865	6433	15542	2738	4481
8	4668	3968	3478	8286	4650	11036	4168	3171
9	269	4651	9177	7376	4206	11293	3534	4750
10	3465	3893	9342	4948	6677	15639	4428	8933

B.2 Information Content

number of used proteins	Performance Petricoin Data Set			Performance Adam Data Set		
	sensitivity	specificity	overall	sensitivity	specificity	overall
1	0.971	0.413	0.705	0.821	0.580	0.703
2	0.928	0.508	0.727	0.476	0.938	0.703
3	0.957	0.889	0.924	0.536	0.926	0.727
4	0.942	0.889	0.917	0.595	0.901	0.745
5	0.957	0.905	0.932	0.738	0.815	0.776
6	0.942	0.889	0.917	0.821	0.815	0.818
7	0.942	0.921	0.932	0.869	0.704	0.788
8	0.928	0.889	0.909	0.869	0.716	0.794
9	0.913	0.873	0.894	0.881	0.667	0.776
10	0.957	0.841	0.902	0.869	0.741	0.806
Full Spectra	1	1	1	0.905	0.926	0.915

number of used proteins	Performance Wikström PP1 IMAC30 low			Performance Wikström PP1 IMAC30 high		
	sensitivity	specificity	overall	sensitivity	specificity	overall
1	0.960	0.200	0.580	0.920	0.320	0.620
2	0.920	0.200	0.560	0.960	0.400	0.680
3	0.920	0.200	0.560	0.960	0.440	0.700
4	0.920	0.200	0.560	0.920	0.480	0.700
5	0.920	0.200	0.560	1	0.480	0.740
6	0.920	0.200	0.560	1	0.400	0.700
7	0.920	0.200	0.560	1	0.400	0.700
8	0.960	0.200	0.580	0.920	0.400	0.660
9	0.960	0.200	0.580	0.920	0.400	0.660
10	0.960	0.200	0.580	0.920	0.480	0.700
Full Spectra	0.800	0.720	0.760	0.720	0.840	0.780

number of used proteins	Performance Wikström PP1 CM10 low			Performance Wikström PP1 CM10 high		
	sensitivity	specificity	overall	sensitivity	specificity	overall
1	0.680	0.560	0.620	0.400	0.960	0.680
2	0.920	0.160	0.540	0.520	0.880	0.700
3	0.880	0.320	0.600	0.480	0.880	0.680
4	0.960	0.360	0.660	0.600	0.880	0.740
5	0.960	0.360	0.660	0.480	0.840	0.660
6	0.880	0.360	0.620	0.600	0.880	0.740
7	0.880	0.360	0.620	0.520	0.880	0.700
8	0.880	0.360	0.620	0.520	0.880	0.700
9	0.920	0.360	0.640	0.520	0.880	0.700
10	0.920	0.360	0.640	0.600	0.880	0.740
Full Spectra	0.880	0.440	0.660	0.680	0.720	0.700

number of used proteins	Performance Wikström CAPS IMAC30 low			Performance Wikström CAPS CM10 low		
	sensitivity	specificity	overall	sensitivity	specificity	overall
1	0.870	0.303	0.588	0.280	0.866	0.569
2	0.870	0.303	0.588	0.280	0.835	0.553
3	0.890	0.283	0.588	0.270	0.856	0.558
4	0.870	0.313	0.593	0.630	0.557	0.594
5	0.850	0.313	0.583	0.330	0.773	0.548
6	0.850	0.333	0.593	0.370	0.814	0.589
7	0.850	0.303	0.578	0.350	0.794	0.569
8	0.860	0.303	0.583	0.360	0.763	0.558
9	0.870	0.333	0.603	0.330	0.814	0.569
10	0.860	0.343	0.603	0.320	0.835	0.574
Full Spectra	0.680	0.727	0.704	0.710	0.577	0.645