Various Kernel Methods with Applications

Kornél Kovács

researcher

Research Group on Applied Intelligence

May 2008

A thesis submitted for the degree of doctor of philosophy
of the University of Szeged

University of Szeged
Doctoral School in Mathematics and Computer Science
Ph.D. Program in Informatics

Preface

Model building is probably the most significant component of scientific thinking baring abstract similarities in all branches of science. We build models and examine how they function, we apply modifications to them as long as we reach our desired goal. Artificial intelligence is a special branch of science, which provides an endless horizon to the lovers of computational model building. All my life, I have developed my abilities in a direction that allows me to create models that are able to simulate certain components of human intelligence. The present dissertation is the anthology of my endeavors to create methods grouped around a certain task group, the elaboration of which has given me high-flown intellectual experiences.

Naturally, to reach the current phase of my scientific career would not have been possible without the support of my immediate surrounding. I find it very important to list the people who have given me this support.

First of all, I owe the most to my supervisor András Kocsor, who has lead me in mastering scientific thinking. He introduced me and made me devote myself to model building. I would also like to thank Csaba Szepesvári, who has helped my work in the examination of models from a mathematical point of view. Further, I thank Dóra Csendes and David Curley for scrutinizing and correcting this thesis from a linguistic point of view. Last, but not least, my heart-felt thanks goes to my wife Judit for providing a secure family background during the time spent writing this work.

Szeged, Hungary Kornél Kovács

List of Figures

List of Tables

Contents

To my wife,
Judit

Chapter 1

Introduction

Machine learning forms a branch of artificial intelligence [59]. It comprises a set of algorithms that enable computers to learn. The concept of learning usually builds on two different approaches: inductive and deductive learning. My thesis follows the inductive learning approach, which retrieves rules or descriptive patterns from massive data sets. Presently, the main focus of machine learning is the complex task of automatic information extraction. Further important application possibilities lie in natural language processing, syntactic pattern recognition, the improvement of search engines, medical diagnosis, bio-informatics, speech recognition, object recognition, and enhancing computer games with humanlike intelligence - only to mention a few fields. Certain machine learning methods attempt to eliminate human resource from the process of data analysis, while others try to make human-machine interaction more human.

Considering the current level of technological advances, machine learning has become the most researched and most intensively developing field of artificial intelligence. The present dissertation focuses on the examination and elaboration of the most novel approach in machine learning, namely that of kernel methods.

This chapter of the thesis contains the following. First, I introduce the basic technique of kernel methods. Following that, I summarize in separate sections the results gained by applying kernel methods to various scientific problems. Finally, I close the chapter with overviewing and evaluating the results.

1.1 Kernel Methods

Kernel methods (KM) are a family of pattern recognition algorithms [61], whose most significant member is the Support Vector Machine (SVM) [69]. The general task of pattern recognition is to identify and examine representative correlations (e.g., clusters, classification decisions, etc.) on general data (e.g., vectors, documents, sequences,

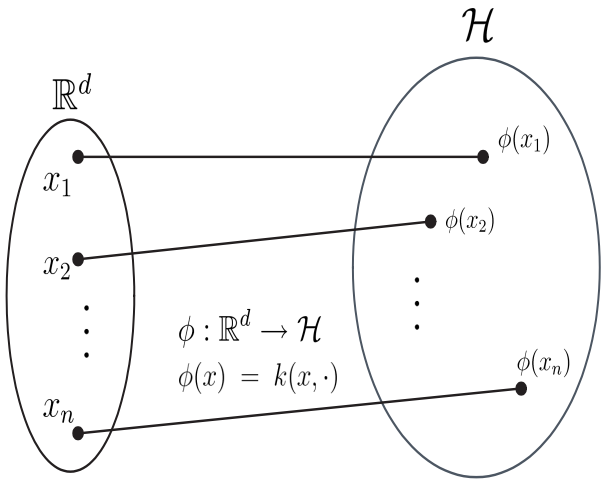

Figure 1.1: An illustration of the kernel mapping idea. The inner product in Hilbert space $\mathcal{H}$ is defined $\langle k(x_1, \cdot), k(x_2, \cdot) \rangle = k(x_1, x_2)$.

pictures, etc.). The KM approach was named after kernel functions, which work in a derived feature space where the real coordinates of patterns never have to be calculated. These methods only rely on the dot product of paired sample points, which are calculated implicitly by applying the kernel functions. Apart from SVM, there are a number of other algorithms belonging to the family of KM: e.g., various regression methods, Fisher's linear discriminant analysis (LDA) [25], principal components analysis (PCA) [29], canonical correlation analysis (CCA) [2], ridge regression [49], spectral clustering [48], and many others. Generally speaking, the majority of kernel methods lead to effectively soluble problems, convex optimisation or eigenproblems.

1.1.1 Inroduction to the kernel mapping idea

Let X, y be the training data, where $X = (x_1, \ldots, x_n)$ are the input patterns ($x_j \in \mathbb{R}^d$) and $y \in \{-1, +1\}^n$ are the corresponding target labels for classification tasks or alternatively $y \in \mathbb{R}$ for regression problems. We will assume that (x_i, y_i), $i = 1, \ldots, n$ are independent, identically distributed random variables.

It is often the case that the problem of classification, regression or relevant feature extraction can be made substantially easier when the data is mapped into an appropriate high dimensional space by some non-linear mapping ϕ and linear methods are applied to the transformed data. If the algorithm is expressible in terms of dot products and if the non-linear mapping

$$\phi : \mathbb{R}^d \to \mathcal{H} \tag{1.1}$$

is such that the dot products of the images of any two points x and y under ϕ can be

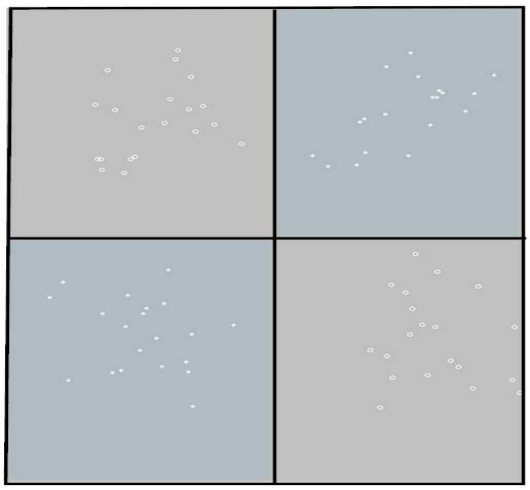

Figure 1.2: The Checkerboard example.

computed as a function of x and y only and in $\mathrm{poly}(d)$-time without explicitly calculating $\phi(x)$ or $\phi(y)$ then the algorithm remains tractable, regardless of the dimensionality of $\mathcal{H}$. This allows us to consider very high or even infinite dimensional image spaces $\mathcal{H}$. We may as well start by choosing a symmetric positive definite function

$$k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}, \tag{1.2}$$

called the kernel function (see e.g. [10]). Then the closure of the linear span of the set

$$\{\, k(x, \cdot) \,|\, x \in \mathbb{R}^d \,\} \tag{1.3}$$

gives rise to a Hilbert space $\mathcal{H}$ where the inner product is defined such that it satisfies

$$\langle k(x_1, \cdot), k(x_2, \cdot) \rangle = k(x_1, x_2) \tag{1.4}$$

for all points $x_1, x_2 \in \mathbb{R}^d$ [45]. The choice of k automatically gives rise to the mapping

$$\phi : \mathbb{R}^d \to \mathcal{H} \tag{1.5}$$

defined by $\phi(x) = k(x, \cdot)$. This is called the kernel mapping idea (cf. Fig. 1.1 and [1; 51; 68]).

1.1.2 Artificial Example

I demonstrate the application of kernel mapping by solving a typical classification problem. Suppose we have a 2 by 2 checkerboard on which we spread sample points

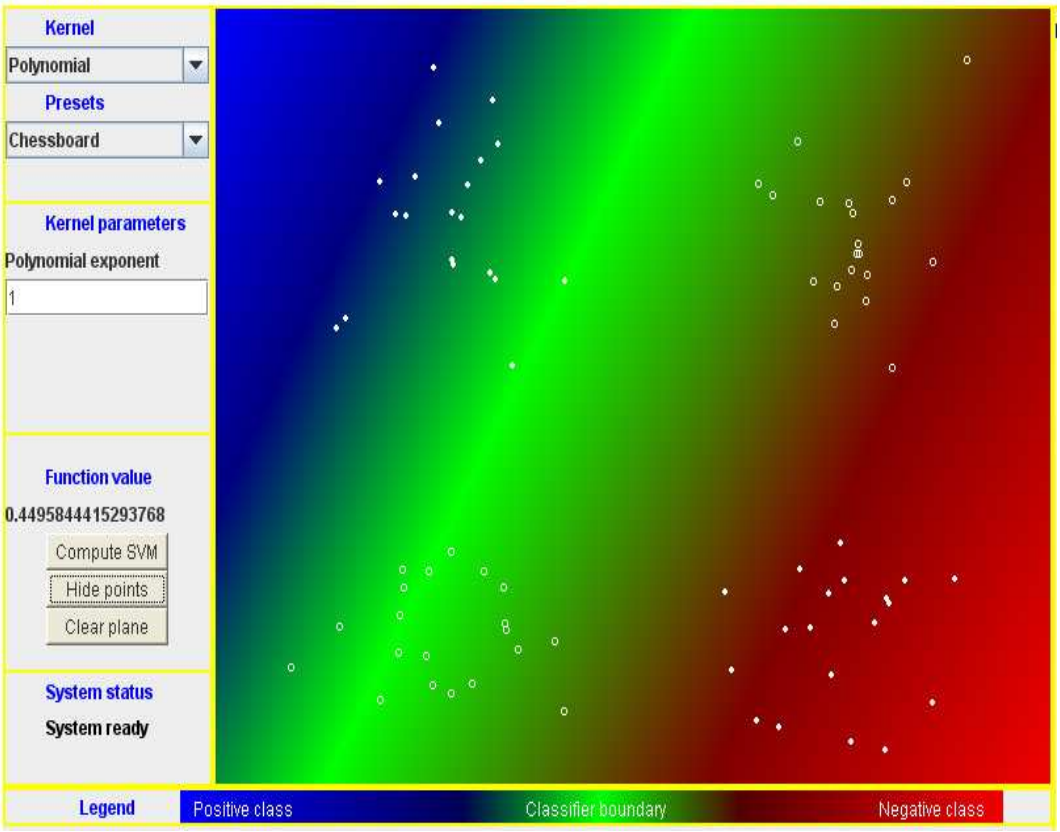

Figure 1.3: The Checkerboard example. The sample points are classified using a linear kernel function.

accidentally. Let us generate positive samples on fields (1,1) and (2,2), while fields (1,2) and (2,1) cover negative samples. Figure 1.2 illustrates a generated two-class set following the above defined pattern. Positive examples are marked with a circle, while negative ones are marked with dots on the figure. The machine learning task is to define a bordering surface, which separates the positive examples from the negative ones. For demonstrational purposes, I use Guillaume Caron's SVM applet:

http://www.site.uottawa.ca/ gcaron/SVMApplet/SVMApplet.html,

which endeavors to separate the positive and negative samples by applying the SVM method. On Figure 1.3, we can see the result of the linear kernel separation. It is instantly visible that with defining a plain linear decision surface (see the light green line appearing in the middle of the green zone) the separation is not possible. The applied kernel function in this case is:

$$k(x, z) = \langle x, z \rangle. \tag{1.6}$$

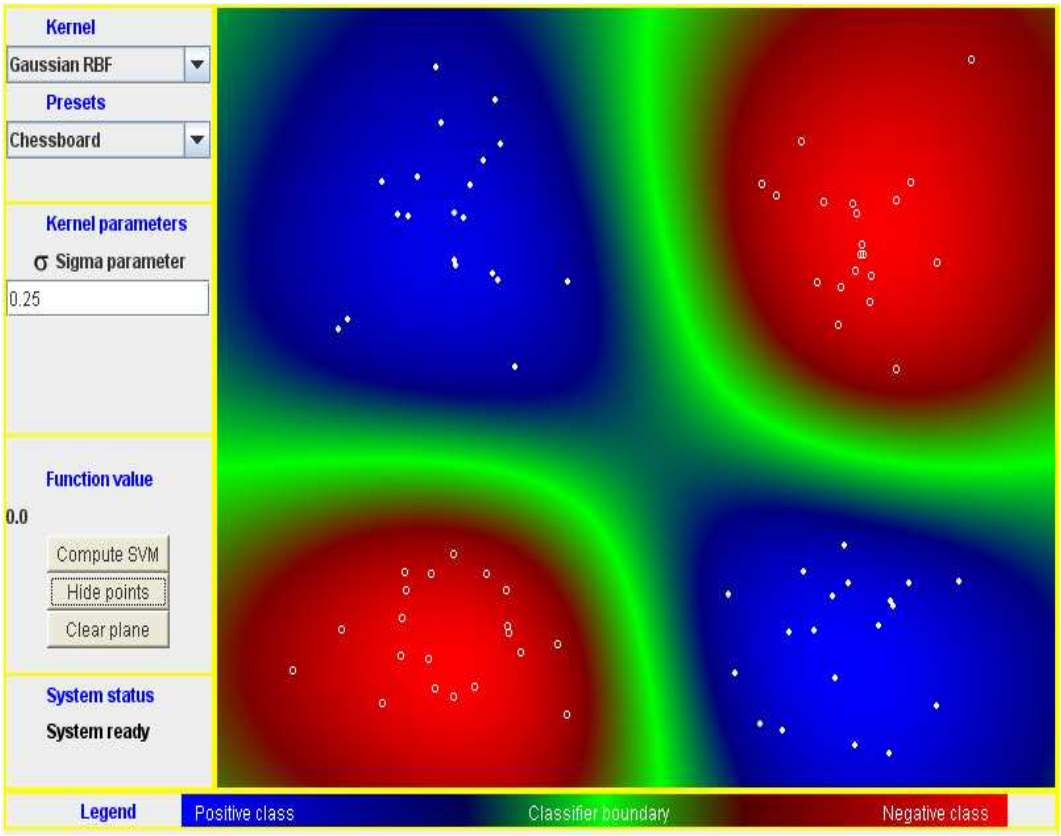

Figure 1.4: The Checkerboard example. The sample points are classified using a non-linear kernel function.

If we replace the linear kernel with the following

$$k(x, z) = \exp\left(\frac{- \parallel x - z \parallel}{0.25}\right), \tag{1.7}$$

so called RBF kernel function, then the hyperplane separation in space $\mathcal{H}$ results in a non-linear decision surface in the original space. The decision surface can be seen in Figure 1.4. The bright green contour imitates the separation lines of the checkerboard. Thus, the success of the separation of the sample points with this kernel method is 100%.

1.2 Structure of the Thesis

Following the introduction of kernel methods, I present the results of the thesis chapter by chapter. Finally, I close the dissertation by summarizing the results.

1.2.1 Summary by Chapters

The thesis comprises two main parts. In the first part (Chapter 2 and 3), I introduce feature extraction methods that are designed to support the efficient solution of classification and regression tasks. In the second part (Chapter 4 and 5), I define two novel branches of kernel-based learning methods.

In Chapter 2, I propose a new feature extraction method called Margin Maximizing Discriminant Analysis (MMDA) [31], which seeks to extract features suitable for classification tasks. MMDA is based on the principle that an ideal feature should convey the maximum information about the class labels and it should depend only on the geometry of the optimal decision boundary and not on those parts of the distribution of the input data that do not participate in shaping this boundary. Further, distinct feature components should convey unrelated information about the data. Two feature extraction methods are proposed for calculating the parameters of such a projection that are shown to yield equivalent results. The kernel mapping idea is used to derive non-linear versions. Experiments with several real-world, publicly available data sets demonstrate that the method yields competitive results.

Face recognition is a highly non-trivial classification problem since the input is high-dimensional and there are many classes with just a few examples per class. In the Chapter, I also propose using Maximum Margin Discriminant Analysis to solve face recognition problems. I show that MMDA is capable of finding good features in face recognition and performs very well provided it is preceded by an appropriate pre-processing phase [37].

In Chapter 3, I consider feature extraction in a regression context. Feature extraction can also be thought of as a way to reduce the dimensionality of the input data whilst keeping important information intact. Principal component analysis (PCA) [29] is such an unsupervised method that seeks to represent the input patterns in a lower dimensional subspace so that the expected squared reconstruction error is minimized. Since PCA is not meant for classification tasks the average regression error may become arbitrarily bad after the data is projected onto the first few principal components. This is what has motivated me in elaborating two new kernel-based feature extraction methods introduced in the Chapter, each of which especially designed to support regression methods. The first approach follows the line underlying the MMDA method, yet in this case I propose to stick to the 'ambiguity decomposition' theory when defining the method. The technique thus created is called Decorrelated Learning Regression (DLR) [65]. The second approach is a variation of the so called Average Derivative Estimation (see statistics), which - in this case - uses kernel functions [65]. The method searches for the relevant sub-space of the feature space, which allows for the more precise solution of the regression problem, i.e. we suppose that the unknown regression function f can

be defined as follows: $f(x) = f_0(Bx)$, where the B matrix maps the d dimensional data to an m dimensional sub-space. With this, I intend to show that using kernel methods in regression problems ensures the selection of relevant sub-spaces from the point of view of regression efficiency.

In Chapter 4, I elaborate on the application of hyperplane-based kernel methods in classification [34]. Practically, I enhance the geometrical theory represented by the Support Vector Machine (SVM) [61]. With extension of the input space's dimension, the bias term used in SVMs can be eliminated. The defined formulae provide for the use of different loss functions. A unique application of the regression formalism makes it possible to develop a new hyperplane-based classification method. In this case, we integrate the elements of the output space into the formula of the hyperplane approach. Finally, I introduce the eigen-analysis technique of Minor Component Analysis. This technique separates the positive and the negative sample points based on the eigen vector pertaining to the smallest eigen value, where the eigen vector is a normal vector defining the hyperplane. In the Chapter, I evaluate the efficiency of the defined methods on standard data bases. Considering the test results, we can see that the efficiency proves to be impressive.

Combinations of basis functions are applied in Chapter 5 to generate and solve a convex reformulation of several well-known machine learning algorithms like certain variants of boosting methods and Support Vector Machines [35]. I call such a reformulation a Convex Machines (CM) approach. The non-linear Gauss-Seidel iteration process for solving the CM problem converges globally and fast [8; 35]. The sparsity of the CM solution can effectively be controlled by some novel heuristics. The proposed techniques [35; 36] are inspired by the methods from linear algebra and Feature Selection (FS). Numerical results and comparisons demonstrate the effectiveness of the methods on publicly available datasets. The FS-based ones perform better than the general purpose selection techniques, but methods from the latter group can be employed on large-sized datasets. The CM approach can perform learning tasks using far fewer basis functions and generate sparse solutions.

The final Chapter provides a short summary of the thesis. Lastly, I round off the work with an Appendix containing a brief summary of the principal results of the thesis.

1.2.2 Summary by Results

The theses of the present dissertation can be differentiated in two ways and can be separated into two different groups. In one reading, the results acquired by the author fall into the topic of kernel-based feature extraction and classification methods within the field of machine learning. In an other reading, we can learn about algorithmic constructions and practical applications. Hereunder, following the structure of the

dissertation, the results are introduced according to the first approach. It is important to note that the list of results below enumerate only those parts of the novelties to which the author has contributed in major part.

The author's kernel-based feature extraction methods form the first group of results. These results are described in detail in Chapter 2 and 3.

I/1. The author has defined the direct version of the MMDA algorithm [31; 37]. This method employs a feature extraction technique that increases the efficiency of classification methods. The author managed to prove the feasibility of the defined method by applying it on several examples of the UCI machine learning database [9].

I/2. The MMDA algorithm is apt by nature to reduce high dimensional feature spaces in order to increase classification efficiency. The author has developed a modified version of this algorithm for face recognition. With the help of the FERET gold standard face recognition database [17], he managed to prove the usability of the introduced method. He also managed to surpass the state-of-the-art results in the field [37].

I/3. Beyond classification, regression problems may also form the focus of feature extraction. The author has also developed a version of the MMDA algorithm for solving regression tasks, with a focus on the retrieval of correlation-free features. The name of the method is Kernel Decorrelated Learning Regression (KDLR) [65]. Based on tests performed over standard regression problems we can state that the approach leads to more efficient regression in practice.

I/4. The author proposed the combination of the statistics-based average derivative estimation method on the one hand, and kernel functions on the other hand. The aim of this novel method called Kernel Average Derivative Estimation (KADE) is the identification of sub-spaces that are relevant from a regression's point of view [65]. By testing on artificial data and comparing relating algorithms the author proved that the identified sub-spaces enable more effective regression in a good number of cases.

Novel kernel-based classification algorithms form the second group of results. These results are detailed in Chapter 4 and 5.

II/1. The author defined a family of hyperplane-based classification methods [34]. He proposed three modifications, each following traditional geometrical concepts: i) he used various loss functions in hyperplane-based classification; ii) he applied linear regression in a unique way to improve classification; iii) he embedded the

Thesis Topics	[31]	[34]	[35]	[36]	[37]	[65]	*Chapter*	*Type*
MMDA	•						2	Feature Extraction
MMDA FACE version	•				•		2	Feature Extraction
KDLR	•					•	3	Feature Extraction
KADE						•	3	Feature Extraction
Hyperplane Classifiers		•					4	Classification
Convex Networks			•	•			5	Classification
Basic Basis Selection Methods			•				5	Basis Selection
Complex Basis Selection Methods				•			5	Basis Selection

Table 1.1: The relation between the thesis topics and the corresponding publications.

output space into the input space and he developed the Minor Component Classifier (MCC) method, which defines a classification hyperplane with the help of an eigenvector pertaining to the smallest eigenvalue of a sample point matrix. The author formed the testing environment of the methods and performed demonstrational tests. Results prove that the methods developed by the author are comparable to results performed by SVM [61].

II/2. The author constructed a classification scheme called Convex Machine Technique, which applies a rare combination basis functions. The method contains a number of machine learning techniques, such as: Support Vector Machine (SVM) [61], Smooth Support Vector Machine (SSVM) [39], Least Square Support Vector Machine (LSVM) [64], Kernel Logistic Regression (KLR) [21], just to mention a few. Inspired by basic numeric mathematical methods, the author also developed three base function selection techniques (RANDOM, MGRAMM, CORR) [35], which he tested on certain elements of the UCI data repository [9].

II/3. He further developed three complex base function selection techniques (SFS, SFFS, PTA) in order to improve the efficiency of classification on the one hand, and to decrease the size complexity of the classification model on the other hand. The defined methods build on the analogy of state-of-the-art feature space selection techniques. Base on test results, it can be declared that these methods support effective classification [65].

Finally, Table 1.1 summarizes which publication covers which method of the thesis.

Part I

Kernel-Based Feature Extraction

Chapter 2

Maximum Margin Discriminant Analysis

This chapter considers feature extraction in a classification context. Feature extraction can be used for data visualization, e.g. plotting data in the coordinate system defined by the principal components of the data covariance matrix. Visualization may help us to find outliers, or meaningful clusters. Another beneficial use of feature extraction is in noise reduction. In classification the goal is to suppress irrelevant information in order to make the classification task using the transformed data easier and simpler. The author describes here a new feature extraction method called Margin Maximizing Discriminant Analysis (MMDA) which seeks to extract features suitable for classification tasks. MMDA is based on the principle that an ideal feature should convey the maximum information about the class labels and it should depend only on the geometry of the optimal decision boundary and not on those parts of the distribution of the input data that do not participate in shaping this boundary.

2.1 Introduction

Feature extraction is the process of transforming the input patterns either by means of a linear or a non-linear transformation. Linear transformations are more amenable to mathematical analysis, while non-linear transformations are more powerful. When linear methods are applied to non-linearly transformed data, the full method becomes non-linear. One important case is when the linear method uses only dot-products of the data. In this case the kernel mapping idea [1; 51; 68] can be used to obtain an efficient implementation whose run time does not depend on the dimensionality of the non-linear map's image space. This 'kernel mapping' idea applies to many well-known feature extraction methods like principal component analysis and linear discriminant

analysis. In classification, the best known example utilizing this idea is the support vector machine (SVM) [68].

Principal component analysis (PCA) [29] is one of the most widely-known linear feature extraction method used. It is an unsupervised method that seeks to represent the input patterns in a lower dimensional subspace such that the expected squared reconstruction error is minimized. By its very nature PCA is not meant for classification tasks. So, in the worst case, the Bayes error rate may become arbitrarily bad after the data is projected onto the first few principal components even if the untransformed data was perfectly classifiable. We shall call this phenomenon a *filtering disaster*. PCA can still be very useful e.g. for suppressing "small noise" which corrupts the input patterns regardless of the class labels. PCA has been generalized to KPCA [62] by using the kernel mapping idea.

Classical linear discriminant analysis (LDA) [25] searches for directions that allow optimal discrimination between the classes provided that the input patterns are normally distributed for all classes $j = 1, \ldots, m$ and share the same covariance matrix. If these assumptions are violated LDA becomes suboptimal and a filtering disaster may occur. Recently, LDA has been generalized using the kernel mapping technique [5; 46; 57] as well.

Discriminant analysis as a broader subject addresses the problem of finding a transformation of the input patterns such that classification using the transformed data set becomes easier (e.g. by suppressing irrelevant components, or noise). More recent methods in discriminant analysis include the "Springy Discriminant Analysis" (SDA) (and its non-linear kernelized counterpart, KSDA), which was derived using a mechanical analogy [30; 32] or, in a special case, as a method for maximizing the between-class average margin itself averaged for all pairs of distinct classes [40]. The goal of the algorithm proposed in [24] is to find a linear transformation of the input patterns such that the statistical relationship between the input and output variables is preserved. The authors of this article use reproducing kernel Hilbert spaces (RKHS) to derive an appropriate contrast function. One distinctive feature of their approach is that the method is completely distribution free. There are many other methods available but we will not discuss them here due to lack of space.

In this chapter we propose a new linear feature extraction method that we will call Margin Maximizing Discriminant Analysis (MMDA). MMDA projects input patterns onto the subspace spanned by the normals of a set of pairwise orthogonal margin maximizing hyperplanes. The method can be regarded as a non-parametric extension of LDA which makes no normality assumptions on the data but, instead, uses the principle that the separating hyperplane employed should depend on the decision boundary only. A deflation technique is proposed to complement this principle to extract a sequence of orthogonal projection directions. A corresponding non-linear feature extraction method

is derived using the kernel mapping technique. The performance of the proposed methods is examined on several real-world datasets. Our findings show that the new method performs quite well and, depending on the dataset may sometimes perform better than any of the other methods tested, resulting in an increase in classification accuracy.

2.2 Principles of MMDA

MMDA makes use of the principal idea underlying LDA: projecting the input data onto the normal of a given hyperplane which separates the two classes best and provides all the information a decision maker needs to classify the input patterns. However, at this point LDA places normality assumptions on the data, whereas we make no such assumptions, but propose to employ margin maximizing hyperplanes instead.

This choice was motivated by the following desirable properties of such hyperplanes [7; 10; 27]: (i) without any additional information they are likely to provide good generalization on future data; (ii) these hyperplanes are insensitive to small perturbations of correctly classified patterns lying further away from the separating hyperplane; moreover, (iii) they are insensitive to small variations in their parameters. In addition to these properties, margin maximizing hyperplanes are insensitive to the actual *probability distribution* of patterns lying further away from the decision boundary. Hence when a large mass of the data lies far away from the ideal decision boundary we can expect the new method to win against those methods that minimize some form of average loss/cost since those methods necessarily take into account the full distribution of the input patterns. An example of such a situation is depicted in Figure 2.1. Note that such situations are expected to be quite common in practical applications like character recognition and text categorization. Actually, the original motivation of MMDA stems from a character recognition problem. Suppose that there are two character classes. Suppose also that the input space is the space of character images. Then, let us concentrate only on two pixels. Specifically, let us assume that pixel 1 is such that for characters in class 1, it can be 'on' or 'off', but in the majority of cases it is 'on'. Further, let us assume that pixel 1 is never 'on' for characters in class 2. Suppose too that pixel 2 is such that it is always 'off' for characters in class 1 and it is always 'on' for characters in class 2. Admittedly, these are strong simplifying assumptions, but similar cases do occur in real-world character recognition tasks. In this simplified case, the ideal feature extractor should actually work like a feature selection method: since the two classes are well separated by using pixel 2, it should project the 2D space of the two pixels onto the second coordinate. Now notice that this is the situation depicted in Figure 2.1. LDA and PCA fail to find such a projection, but MMDA succeeds in doing so.

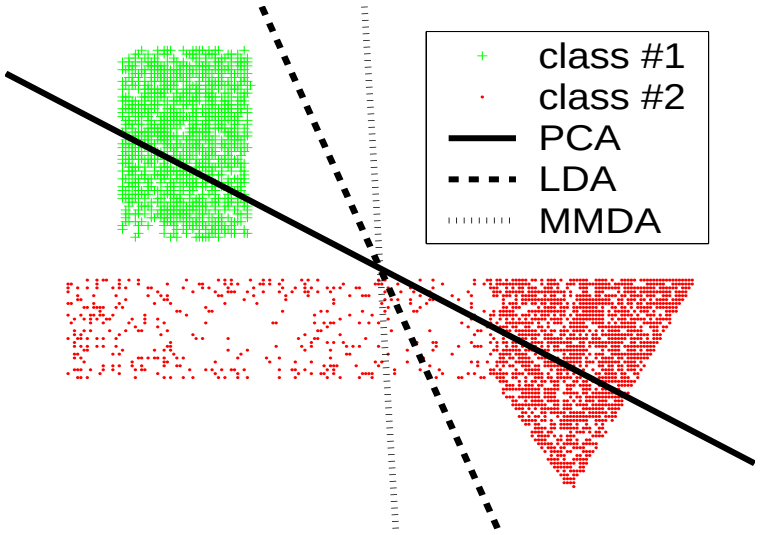

Figure 2.1: An illustration of the behavior of PCA, LDA and MMDA for a binary classification problem. The figure shows the one-dimensional subspace represented by a hyperplane that PCA, LDA and MMDA project the data onto. Although the data is linearly separable, PCA and LDA fail to find a subspace such that the data when projected onto the subspace remains linearly separable. MMDA avoids this problem by projecting onto the normal of a separating hyperplane when such a hyperplane exists.

We supplement the idea of projecting onto the space spanned by the normal of a margin maximizing hyperplane by a deflation technique which guarantees that all subsequent hyperplanes (and all subsequent normals) are orthogonal to each other. As a result each successive feature extraction step extracts "new" information unrelated to information extracted in the previous steps.

Deflation can be incorporated as a step to transform the data covariance matrix. However, we can also incorporate a suitable orthogonality criterion in the equations defining the margin maximizing hyperplane. We will show the equivalence of these two approaches in the next section and discuss their relative merits.

2.3 Linear Feature Extraction

2.3.1 The Deflation Approach

Let X, y be the training data, where $X = (x_1, \ldots, x_n)$ are the input patterns ($x_j \in \mathbb{R}^d$) and $y \in \{-1, +1\}^n$ are the corresponding target labels. We will assume that (x_i, y_i), $i = 1, \ldots, n$ are independent, identically distributed random variables.

Assuming that the data (X, y) is separable, the maximum margin separating hyperplane can be found as a solution of a quadratic programming problem [10]. When the data is not separable the maximum margin separation problem is modified to simultane-

ously maximize the margin and minimize the error [68]. This still results in a quadratic programming problem. In order to introduce the corresponding equations formally, let us fix a positive real number C that we will use to weight the misclassification cost. Then the maximum margin separation (MMS) problem is defined as follows: given (X, y, C) find $w \in \mathbb{R}^d$, $b \in \mathbb{R}$ and

$$\xi = (\xi_1, \ldots, \xi_n)^T \in \mathbb{R}^n \tag{2.1}$$

such that[1]

$$\begin{aligned} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n} \xi_i \to \min \text{ s.t.} \\ y_i(w^T x_i + b) \geq 1 - \xi_i, \\ \xi_i \geq 0, \quad i = 1, \ldots, n. \end{aligned} \tag{2.2}$$

MMDA now proceeds as follows: given (X, y, C), find the solution of the MMS problem (X, y, C). Let this solution be (w_1, b_1). The first extracted feature component is

$$f_1(x) = w_1^T x. \tag{2.3}$$

Now transform the data by projecting it onto a space orthogonal to w_1. For simplicity, assume that w_1 is normalized so $\|w_1\|_2 = 1$. Then the projected data is given by

$$x_i' = x_i - (w_1^T x_i) w_1. \tag{2.4}$$

Let X' denote the matrix $(x_1', \ldots, x_n')$ and let (w_2, b_2) be the solution of the MMS problem (X', y, C). Then the second extracted feature component is

$$f_2(x) = w_2^T x', \tag{2.5}$$

where

$$x' = x - (w_1^T x) w_1. \tag{2.6}$$

This procedure can be repeated as many times as desired. The following proposition shows that w_1 and w_2 are orthogonal.

Proposition

Let (w_1, b_1) be the solution of the MMS problem (X, y, C) and (w_2, b_2) be the solution of the MMS problem (X', y, C), where $X' = (x_1', \ldots, x_n')$ with x_i' defined by (2.4). Then the vectors w_1 and w_2 are orthogonal. □

[1] Here $\|w\|_2$ denotes the ℓ^2 norm of w.

Proof

First, let us split vector w_1 to the sum of two vectors. $w_1 := w_x + w_y$, where

$$w_x \in \ span(x_1, \ldots, x_n) \tag{2.7}$$

and $w_y \perp w_x$. Then in the target function of Eq. (2.2)

$$\|w_1\|_2^2 = \|w_x + w_y\|_2^2 = \|w_x\|_2^2 + \|w_y\|_2^2, \tag{2.8}$$

while $w_x \perp w_y$.

Note that in Eq. (2.2)

$$y_i(w_1^T x_i + b) \geq 1 - \xi_i \Longleftrightarrow y_i(w_x^T x_i + b) \geq 1 - \xi_i, \tag{2.9}$$

which implies in the optimal solution w_y equals the zero vector. Otherwise, based on Eq. (2.8), if w_x is not in $span(x_1, \ldots, x_n)$ cutting off the section outside the linear span would lead to a better solution.

Now taking into account that $x_i' = x_i - (w_1^T x_i)w_1$ for all i, and similarly to w_1, the w_2 vector is also in $span(x_1,' \ldots, x_n')$. We have that w_1 is orthogonal to w_2.

□

A corollary of this proposition is that

$$f_2(x) = w_2^T(x - (w_1^T x)w_1) = w_2^T x. \tag{2.10}$$

Similarly, if $w_3, \ldots, w_r$ ($r \leq d$) are the normals extracted up to step r then $w_1, \ldots, w_r$ are pairwise orthogonal and the ith feature value $f_i(x)$ can be computed via:

$$f_i(x) = w_i^T x. \tag{2.11}$$

In order to derive our first practical algorithm let us note that the solution of the MMS problem is typically obtained via the Langrangian dual of (2.2):

$$\begin{aligned} -\frac{1}{2}\alpha^T R\alpha + \alpha^T 1 \rightarrow \max \\ \text{such that} \quad y^T\alpha = 0, \quad 0 \leq \alpha \leq C1, \end{aligned} \tag{2.12}$$

where the matrix R is defined by

$$R = YX^T XY \tag{2.13}$$

and

$$Y = \mathrm{diag}(y_1, \ldots, y_n) \tag{2.14}$$

and $\alpha \in \mathbb{R}^n$ [68]. Here $C1 = (C, \ldots, C)^\top \in \mathbb{R}^d$ and the comparison of vectors is made one component at a time. Given α, the solution of (2.12), the solution of the MMS problem (X, y, C) is recovered through

$$w = X\alpha \quad \text{and} \quad b = 1^T\alpha. \tag{2.15}$$

We shall call (2.12) the dual MMS problem parameterized by (R, y, C).

Let X' be defined as before. Notice that (2.12) depends on the data vector X only through the matrix R. Hence, the Langrangian dual defined for the transformed data X' takes the form in (2.12), but R needs to be recalculated. The next proposition shows how to do this in the general case when the data is projected onto a subspace spanned by an orthonormal system:

Proposition

Let X' be the data X projected onto a space orthogonal to the orthonormal system $W = (w_1, \ldots, w_r)$. Then

$$R' = Y(X')^T X' Y = Y\left(X^T X - V^T V\right) Y, \tag{2.16}$$

where we define V by $V = W^T X$. In particular, if $W = XA$ for some matrix A then R' can be calculated by

$$R' = Y\left(K - (KA)(KA)^T\right) Y, \tag{2.17}$$

where $K = X^T X$. □

Proof

Since $x_i' = x_i - (w_1^T x_i) w_1$ for all i we can say that

$$X' = X - WW^T X = (I - WW^T)X. \tag{2.18}$$

Thereby using the orthogonality of matrix W, we get that

$$(X')^T X = X^T (I - WW^T)^T (I - WW^T) X \tag{2.19}$$

$$= X^T (I - 2WW^T + WW^T WW^T) X = X^T (I - WW^T) X \tag{2.20}$$

$$= X^T X - V^T V. \tag{2.21}$$

A special case of the proposition is that when $W = XA$ is a result af simple algebraic

conversions. □

The significance of this result is that it shows it is possible to use existing SVM code to extract a sequence of orthogonal margin maximizing hyperplanes just by transforming the matrix R. This proposition is given extra weights as it shows that it is possible to apply the kernel mapping idea to MMDA. This will be considered in more detail in Section 2.4.

2.3.2 The Direct Method

The deflation approach requires $O(n^2)$ calculations when calculating the transformed matrix R'. The method we consider in this section avoids this at the price of slightly increasing the dimensionality of the quadratic programming problem.

Let us define the maximum margin separation problem with orthogonality constraint (MMSO problem) as follows. Let u be a d-dimensional vector: $u \in \mathbb{R}^d$. The MMSO problem parameterized by (X, y, C, u) is to find $w \in \mathbb{R}^d$, $b \in \mathbb{R}$ and $\xi = (\xi_1, \ldots, \xi_n) \in \mathbb{R}^n$ such that

$$\begin{aligned} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n}\xi_i \rightarrow \min \text{ s.t.} \\ y_i(w^T x_i + b) \geq 1 - \xi_i, \\ \xi_i \geq 0, \quad i = 1, \ldots, n, \\ u^T w = 0. \end{aligned} \tag{2.22}$$

Let $H = H_{(u,0)}$ be a hyperplane with normal $u \in \mathbb{R}^d$ (we assume $\|u\|_2 = 1$ as before) and bias 0. Let $X' = (x'_1, \ldots, x'_n)$ be the matrix whose columns are composed of the x_i vectors projected onto

$$H: \; x'_i = x_i - (u^T x_i)u \tag{2.23}$$

as before. The following proposition shows the equivalence of the solutions of MMSO problem and the solutions obtained using the deflation approach:

Proposition

Let C have a fixed positive value. Given the data (X, y) and the hyperplane H with normal u satisfying $\|u\|_2 = 1$ and bias 0, the following holds: let X' denote the data projected onto the hyperplane H. Then the solutions of the MMS problem (X', y, C) and the MMSO problem (X, y, C, u) coincide. □

Proof

The proof is trivial, while the $u^T w = 0$ condition in Eq. (2.22) is equivalent with the optimization over a hyperplane with normal u . □

According to this last proposition, we obtain equivalent solutions to those gotten using the deflation approach when orthogonality constraints are added to the MMS problem. It is readily seen that the proposition remains true when the number of orthogonality constraints – r, say – is bigger than one. The corresponding MMSO problem will be denoted by (X, y, C, U), where $U = (u_1, \ldots, u_r)$ is the matrix of vectors that are used to define the orthogonality constraints.

It is not difficult to prove that the solution of an MMSO problem (X, y, C, U) may be obtained by solving the following dual quadratic programming problem:

$$-\frac{1}{2}\left(\alpha^\top Y K Y \alpha + \gamma^\top U^T U \gamma\right) + \alpha^\top 1 + \gamma^\top U^T X Y \alpha \to \max$$
$$\text{such that} \quad y^T \alpha = 0, \quad 0 \leq \alpha \leq C1. \tag{2.24}$$

Since the number of columns of U is r, the dimensionality of γ will also be r, and hence the number of variables in the above quadratic programming problem will be $n + r$.

The direct method works as follows: Given the data (X, y, C), let (w_1, b_1) be the solution of the MMS problem (X, y, C). Assuming that the solution vectors $(w_1, b_1), \ldots, (w_{r-1}, b_{r-1})$ have already been computed, (w_r, b_r) is obtained as the solution of the MMSO problem (X, y, C, W_{r-1}), where $W_{r-1} = (w_1, \ldots, w_{r-1})$.

Now we will show (i) that the dual MMSO optimization problem (X, y, C, W_r) can be put into a form where the dependence on X is only through the dot product matrix

$$K = X^T X, \tag{2.25}$$

and (ii) that the matrices involved in the dual MMSO optimization problem can be computed in an incremental manner in time $O(mn)$, where m is the number of non-zero elements of $\alpha^{(r)}$. We know that the vectors in W_r lie in the span of

$$X: \; w_i = X\alpha^{(i)}. \tag{2.26}$$

Therefore,

$$W_r = X A_r, \tag{2.27}$$

where $A_r = (\alpha^{(1)}, \ldots, \alpha^{(r)})$. Hence,

$$W_r^T W_r = A_r^T K A_r \;\text{ and }\; W_r^T X Y = A_r^T K Y. \tag{2.28}$$

So

$$W_r^T W_r = [A_{r-1},\, \alpha^{(r)}]^T K [A_{r-1},\, \alpha^{(r)}], \tag{2.29}$$

where the subblocks can be computed by

$$A_{r-1}^T K A_{r-1},\;\; A_{r-1}^T \alpha^{(r)},\;\; (\alpha^{(r)})^T A_{r-1} \;\; \text{and} \;\; (\alpha^{(r)})^T K \alpha^{(r)}, \tag{2.30}$$

respectively. Further,

$$W_r^T XY = A_r^T KY \tag{2.31}$$

and hence,

$$(W_r^T XY)^T = (YKA_{r-1},\, YK\alpha^{(r)}). \tag{2.32}$$

Thus the direct method may be computationally cheaper than the deflation approach when the value of m (the number of support vectors) obtained in step r is much smaller than the number of data points.

2.4 Non-linear Feature Extraction

It is clear that the kernel mapping idea can be used to obtain an efficient non-linear version of MMDA too: Firstly, the MMS problem at the heart of MMDA is actually the problem solved by SVMs, which itself builds on the kernel mapping idea [10]. It is well known that the MMS problem can be efficiently solved in the $\mathcal{H}$ feature space. However, for the sake of completeness, we shall briefly describe how to 'kernelize' the MMS problem. The input patterns X appear in Equation (2.12) only through the dot product matrix $X^T X$. Hence, defining the matrix K by

$$K_{ij} = k(x_i, x_j), \quad 1 \leq i, j \leq n \tag{2.33}$$

and replacing $X^T X$ in Equation (2.12) by K, we obtain a quadratic programming problem such that (if α denotes its solution)

$$w(\cdot) = \sum_{i=1}^{n} \alpha_i k(x_i, \cdot) \;\; \text{and} \;\; b = \alpha^T 1 \tag{2.34}$$

is the solution of the MMS problem (Φ, y, C), where

$$\Phi = (\phi(x_1), \ldots, \phi(x_n)). \tag{2.35}$$

Now assuming that r directions

$$W = (w_1, \ldots, w_r) \tag{2.36}$$

have already been determined, the $(r+1)$th direction can be computed as the solution of the dual MMS problem with R replaced by R', where R' is defined in Proposition 2. Since

$$W = \Phi A \tag{2.37}$$

for an appropriate matrix A (the jth column of A is the solution of the jth dual subproblem) and $\Phi = (\phi(x_1), \ldots, \phi(x_n))$, R' can be computed by Proposition 2, where K is now defined by (2.33). It was also found that the dual of the MMSO problem can be expressed in terms of $X^T X$ when W lies in the span of X. Hence the dual of the MMSO problem can also be expressed using K only, and thus it can be solved efficiently, regardless of the dimensionality of $\mathcal{H}$.

Finally, rewriting (2.11) in terms of the kernel function we find that the ith component of the feature extraction mapping can be evaluated using

$$f_i(x) = \sum_{j=1}^{n} \alpha_j^{(i)} k(x_i, x), \tag{2.38}$$

where $\alpha^{(i)}$ is the solution of the ith dual subproblem. Eq. (2.38) follows directly from

$$w_i(\cdot) = \sum_{j=1}^{n} \alpha_j^{(i)} k(x_i, \cdot) \tag{2.39}$$

and the fact that

$$f_i(x) = \langle w_i, \phi(x) \rangle = \sum_{j=1}^{n} \alpha_j^{(i)} \langle \phi(x_i), \phi(x) \rangle. \tag{2.40}$$

Dataset	# classes	# attribs	# train	# test
Bupa	2	7	699	*
Pima	2	8	768	*
Iono	2	34	351	*
Heart	2	13	303	*
DNA	3	181	2000	1186
Satimage	6	36	4435	2000
Optdigits	10	64	3823	1797

Table 2.1: The characteristics of datasets used in the experiments. In the cases marked by * 10-fold class-balanced cross-validation was used to measure performances.

2.5 Experiments on UCI Datasets

2.5.1 Visualisation Experiments

In the first experiment we sought to demonstrate the visualization capability of MMDA. We used the *Wine* dataset from the UCI machine learning repository [9] which has 13 continuous attributes, 3 classes and 178 instances. We applied PCA, LDA and MMDA to these data sets. Two dimensional projections of the data are shown in Figure 2.2. In the case of PCA and LDA, the data is projected onto the eigenvectors corresponding to the two largest eigenvalues. Since MMDA is defined for binary classification problems, with multi-class problems we need to group certain classes together. In this example the first direction is obtained by grouping classes 2 and 3 together into a single class, while the second direction is obtained by grouping classes 1 and 3 together. It can be seen that for both LDA and MMDA the data became separable in the projection space. It was also noticed that the margin of separation is larger for the case of MMDA, as expected. Note that the size of the margin can be controlled to some extent by the parameter C. For this figure we used $C = 1$ and the data was centered and scaled to have unit variance (this transformation was applied in all of our other experiments as well). Actually, the data is not linearly separable in the case of the PCA projection.

2.5.2 Subspace Selection Experiments

Next we investigate whether MMDA can estimate useful subspaces that preserve information necessary for the classification task. For this we ran MMDA on a number of binary classification problems. We changed the number of dimensions of the estimated subspace and measured the classification accuracy that could be achieved by projecting data on the extracted subspace. This experiment was run with both the linear and kernelized versions of MMDA. Since there is obviously no optimal classifier we decided to estimate the quality of the extracted subspace by training an artificial neural network

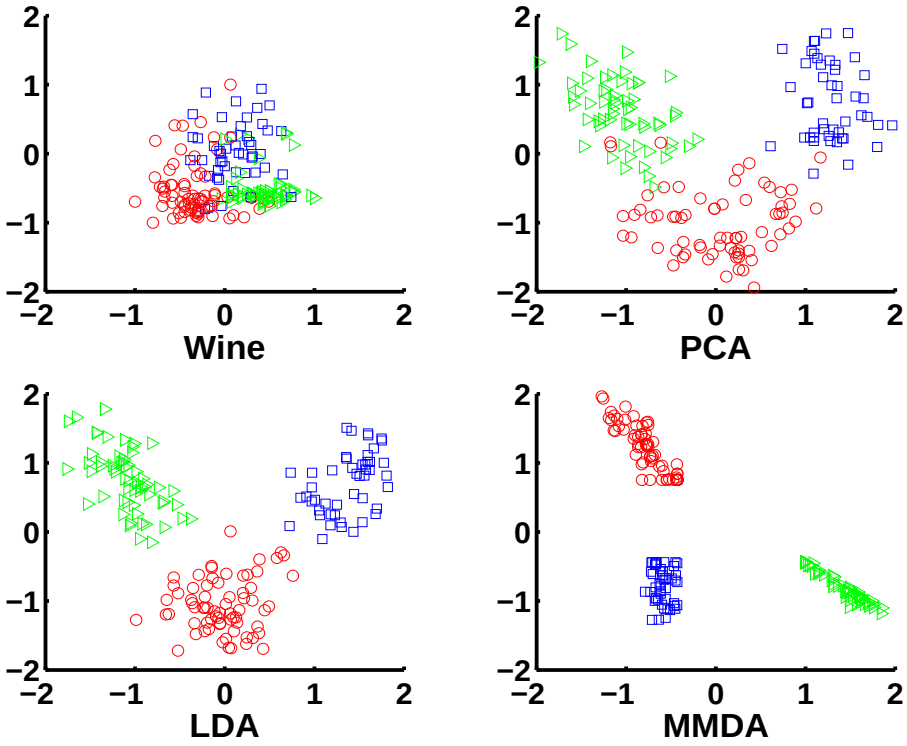

Figure 2.2: Scatter plot of wine data projected onto a two-dimensional subspace. The upper left subfigure shows the projection onto the first two attributes, while the other three show the results of a PCA, LDA and MMDA transformation, respectively.

(ANN) classifier on the projected data. The ANN was trained for a fixed number of iterations using batch gradient descent with a constant learning rate. There is one hidden layer and the number of hidden nodes is three times the number of inputs. Our experiments showed that, on the datasets used, this method is competitive with the results of SVMs. We chose to combine ANNs with linear feature extraction since (i) we wanted to keep the algorithms simple and since (ii) the test speed of the resulting composite classifier was then usually very high. High classification speed is important for some applications like OCR. SVMs need special postprocessing to achieve comparably high speeds, therefore we decided to use ANNs.

The characteristics of the datasets used in this experiment are shown in Table 2.1, while the results are presented in Figures 2.3 and 2.4. The results labeled 'original' were obtained using an ANN trained on the original, untransformed data. It may be seen that for a number of datasets very good classification rates are achieved with only a few features. Also, in certain cases performance drops when the dimensionality of the subspace is increased.

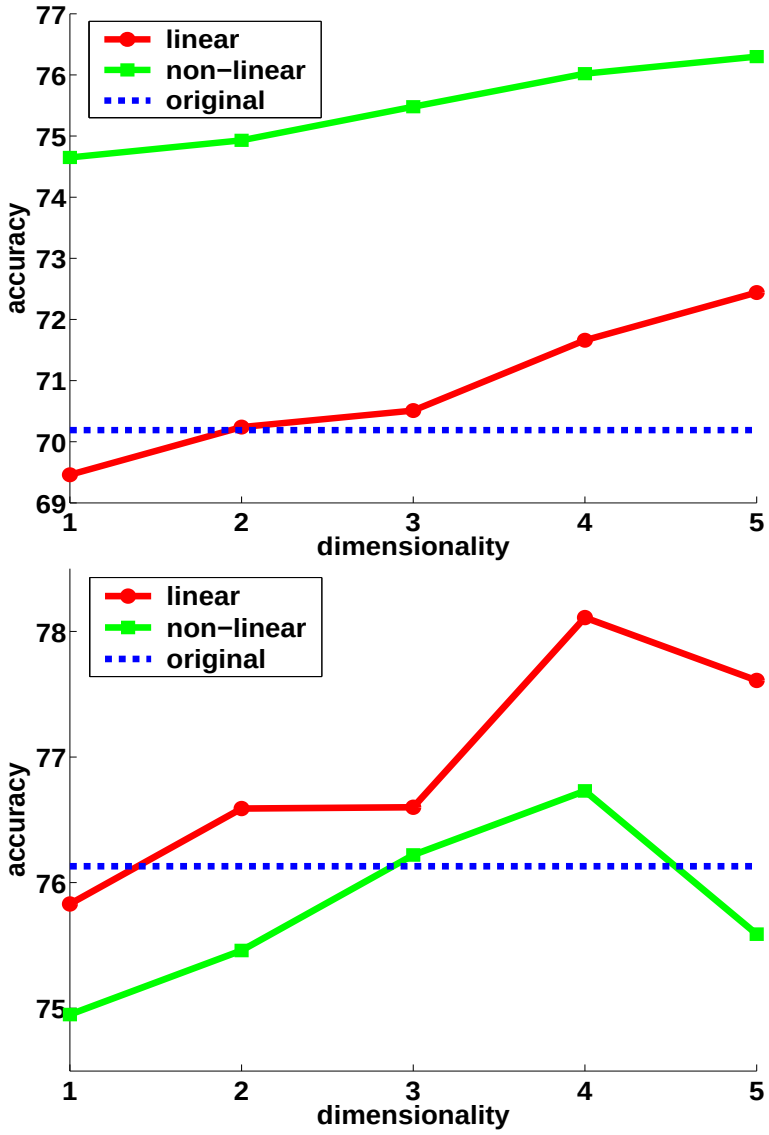

Figure 2.3: Accuracies achieved by training a neural network on the subspace extracted by (K-)MMDA shown for datasets called Bupa and Pima.

2.5.3 Multi-class Classification Tests

Next, we tested the performance of the method on a number of larger multi-class problems. For multi-class problems we used the "one vs. all" approach: basically when the number of classes is m we ran (K-)MMDA m times with one class against all the others. We chose this approach for its simplicity. This is probably a suboptimal approach, though our initial experiments with output-coding suggests that accuracies obtained this way are quite good.[2] In this case we tested the interaction of MMDA with several classifiers. These were the ANN introduced earlier, support vector machines with the

[2]Note that we lose pairwise orthogonality (directions extracted for different subproblems are not necessarily orthogonal). In the future we plan to investigate the case when pairwise orthogonality is enforced. Note here that as a kernel for K-MMDA fourth order cosine kernels were used.

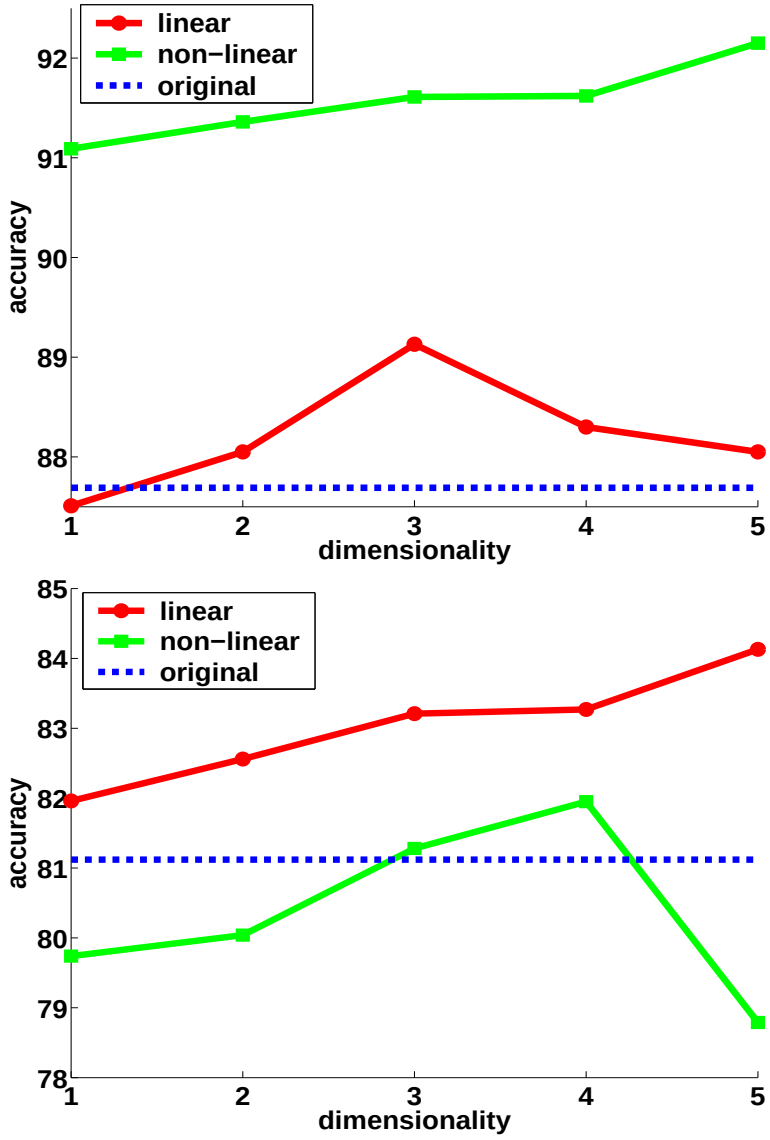

Figure 2.4: Accuracies achieved by training a neural network on the subspace extracted by (K-)MMDA shown for datasets called Ionosphere and Heart Disease.

linear kernel and C4.5 [54]. The results for the three datasets are shown in Figure 2.5. For comparision we also included the results obtained with 'no feature extraction', PCA and LDA. For the datasets DNA and Optdigits we got competitive results, but for Satimage the result for the tested cases were worse than those obtained with the other methods tested. In particular, in the case of Satimage all feature extractors yielded worse results than those using *no* feature extractor. We conjecture that the optimal subspace for Satimage might be just the (untransformed) space of input patterns.[3]

[3]In these experiments K-MMDA was implemented using a Gauss-Seidel iteration (or the Adatron) and (without loss of generality) we set $b = 0$.

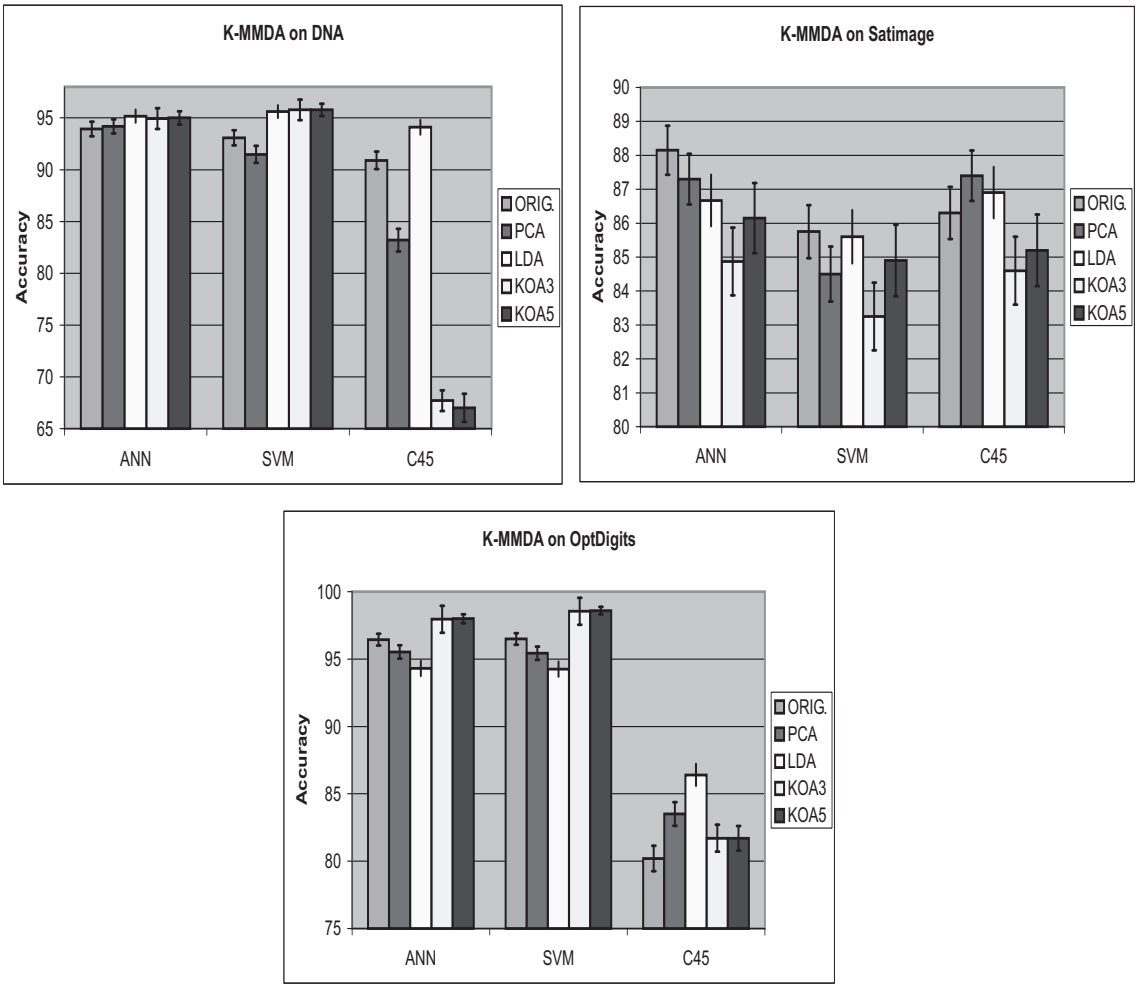

Figure 2.5: The interaction of classifiers and feature extractors. The results for K-MMDA are shown. The label KOA$< i >$ means that K-MMDA was run in a one-vs-all manner, and for each subproblem where i is the number of directions extracted per subproblem.

2.6 MMDA for Face Recognition

Human face recognition is a special classification problem where the number of classes is high, there are only a few samples per class and the input space is high-dimensional. These properties make face recognition an especially challenging classification problem.

2.6.1 Motivation

Successful approaches to face recognition must exploit the inherent regularity of face images: a good classifier has to suppress within-class (intra-personal) differences while enhancing between-class (or extra-personal) differences. This is the basic idea underlying some subspace methods, the best known examples of which include the Fisherface method and Moghaddam and Pentland's Bayesian Face Recognition (BFR) methods [6; 47]. These methods work by projecting the space of images into a lower-dimensional

space where classification is typically done by resorting to 1-nearest neighbour classification with an appropriately defined distance function. The difficulty is that since there are only a very few examples per class, the information of what needs to be suppressed/enhanced cannot be class specific still, the features should maximize the amount of information kept about the class labels.

In this chapter it was already shown that MMDA can produce high quality features. The performance of classifiers built on the top of these features often exceeds the performance of other state-of-the-art methods. Hence it seems worthwhile to apply MMDA to the problem of extracting features in face recognition.

2.6.2 MMDA Face Recognition Method

Since MMDA is defined for binary classification problems, with multi-class problems one needs to group certain classes together. In [31] it was suggested that MMDA should be used following a "one-vs.-all" approach: basically when the number of classes is m, MMDA is run m times with one class against all the others. This is a simple approach and in [31] it was suggested that although it is likely to be suboptimal, it can yield a sufficiently good performance even when compared with the more involved approach based on output-coding.

It should be mentioned that the one-vs.-all approach cannot produce useful features in face recognition tasks as here all the m subproblems are seriously skewed with one class having only a few elements and the other has lots. Such skewed distributions will yield highly correlated features for the independent subproblems since the subproblems are "well aligned" (it is easy to see that forcing independent features to be decorrelated does not help either due to the large overlap of the problems). The same conclusion holds for the features obtained using the output-coding approach since there classes are grouped together without taking into account their relations in the input space. In a typical subtask persons with very different faces could be grouped together while persons with similar faces might be assigned to different classes.

This gives us the idea of using the available data to create the binary classification subproblems for MMDA. The particular approach suggested here is to use information in the images to create the subproblems. One approach for doing just this is the following. Some method (like LDA or PCA) is used to generate a number of unrelated features. For each feature, the projections of training images on a selected feature are computed. This produces a number of points on the real-line. Next, the persons in the training set are grouped into two groups so as to minimize the total within-group distortion between the previously calculated points on the real-line (this is a special case of k-means clustering and can be implemented efficiently). MMDA is then run on this binary classification problem (on the original, untransformed images) and the

Algorithm 1 Feature Extraction by MMDA for Face Recognition

input: $(m, (x_1, y_1) \ldots, (x_N, y_N))$ // no. of subjects, list of face-image, person id pairs

$F := ()$; $X^i := \{x_j | y_j = i\}$, $i = 1, \ldots, m$; // images of person i

$(w_1, \ldots, w_n) := \mathrm{FE}((x_1, y_1) \ldots, (x_N, y_N))$; // extract n features using method FE

for $i \in \{1, \ldots, n\}$ **do**

$z_j := w_i^T x_j$, $j = 1, \ldots, N$; // project images

$Z^i := \{z_j | y_j = i\}$, $i = 1, \ldots, m$; // collect projected images of person i

Find $(v_1, \ldots, v_m) \in \{-1, 1\}^m$ such that

$$\sum_{\substack{v_i=-1, v_j=-1 \\ i \neq j}} \sum_{\substack{z \in Z^i \\ z' \in Z^j}} (z - z')^2 + \sum_{\substack{v_i=1, v_j=1 \\ i \neq j}} \sum_{\substack{z \in Z^i \\ z' \in Z^j}} (z - z')^2$$

is minimized.

$F_0 := MMDA(\cup_{v_i=-1} X^i, \cup_{v_j=+1} X^j)$; // extract features using MMDA

append$(F, F0)$; // append features to the list of features extracted so far

end for

return F

corresponding features are then saved. The process is continued with the next feature. The union of features extracted this way defines the extracted feature space. The proposed algorithm is listed above.

2.7 Face Recognition Experiments

We used the CSU Face Identification Evaluation System (CSU FIES) to evaluate the performance of our algorithm on the FERET database [17].

The FERET dataset includes only raw face data collections. Examples of variations of collections can be seen in Fig. 2.6 and tipical set of images collected in one sitting in Fig. 2.7. CSU (Colorado State University) has developed a usefull preprocessing tool that performs same basic pre-processing steps (normalisation, equalization, face cutting). We used the CSU FIES environment for performing our experiments with MMDA.

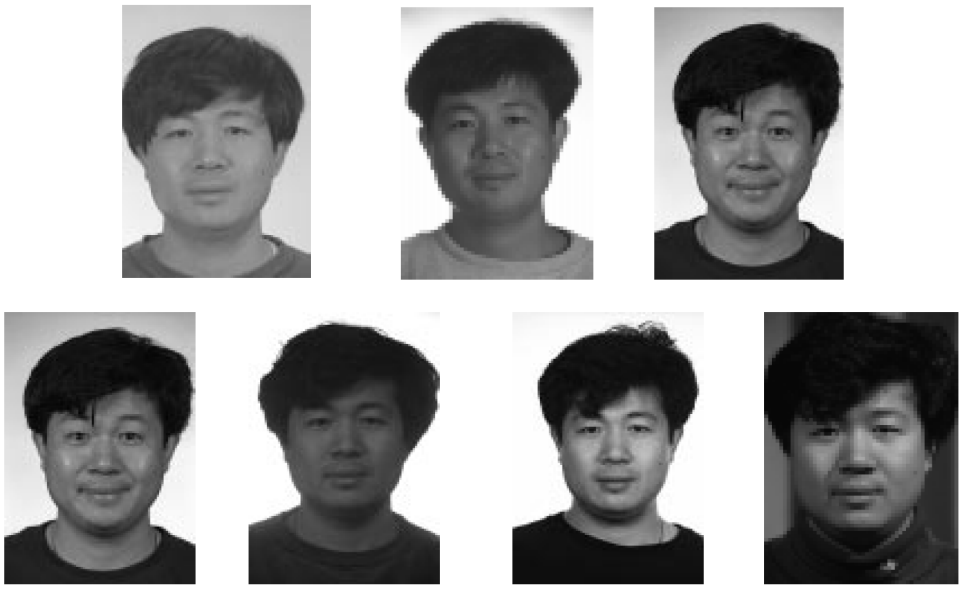

Figure 2.6: Examples of variations of collections.

2.7.1 Experimental Setup

Since the idea can be applied to other supervised feature extraction algorithms, we also decided to test it using LDA in place of MMDA, so that we could see the effect of the data-grouping procedure and the effect of MMDA separately. In addition, both PCA and LDA were used as the underlying feature-extraction methods. This gave rise to the algorithms LDA(LDA), LDA(PCA), MMDA(LDA) and MMDA(PCA). The CSU Toolkit allows one to choose a number of distance functions. We tried out a large number of choices, but only the results for the best distance functions are shown here. For the algorithms mentioned so far this was the so-called covariance distance function. The two other algorithms tested were PCA and a combination of LDA and MMDA method. For PCA the best results were obtained using the distance called MahCosine in the CSU Toolkit (MahCosine is the cosine distance measured in the Mahalanobis space; for more details see [17]), while the combined approach, which contains the original LDA direction itself beside the derived MMDA one, employs also the above mentioned covariance distance.

2.7.2 Experiments with the standard probs sets

Figures 2.8-11 show the Cumulative Match Curve of recognition rate versus recognition rank that was obtained for the standard probe sets FAFB, FAFC, DUP1 and DUP2. For a description of these probe sets the reader should see Section 4.1 of [4]. In brief, FAFB contains images that were taken at the same time as the training (gallery) images, but the subjects were asked to assume a different facial expression than those in the gallery

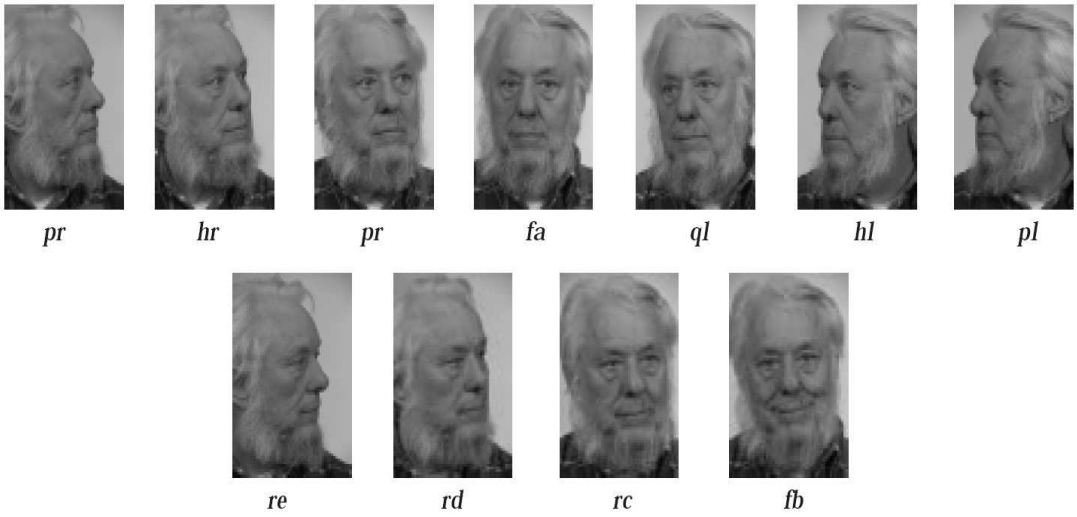

Figure 2.7: Tipical set of images collected in one sitting.

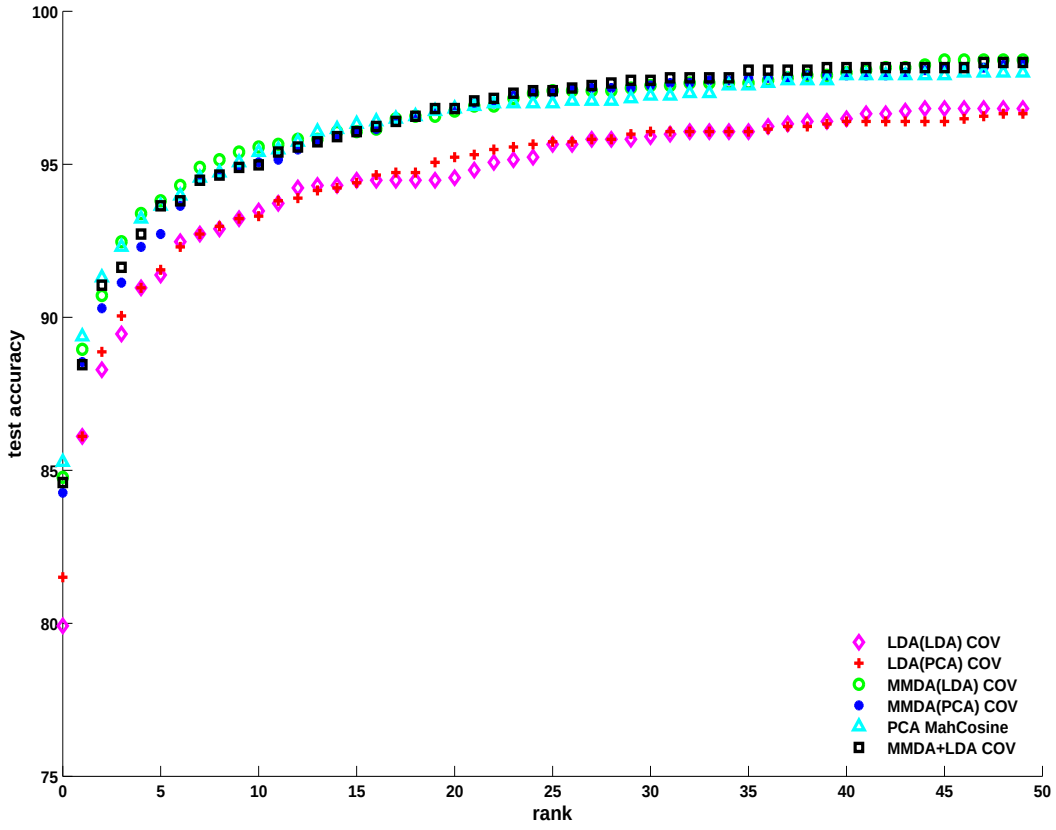

Figure 2.8: Results obtained for the standard probe set FAFB of FERET.

images. FAFC contains the images of subjects under significantly different lighting conditions (this is a smaller set than FAFB). DUP1 contains images taken between one minute and 1,031 days after the gallery image was taken, while DUP2 is a subset of DUP1 where the probe image was taken at least 18 months after the probe image.

According to the figure, MMDA based methods with the COV distance consistently

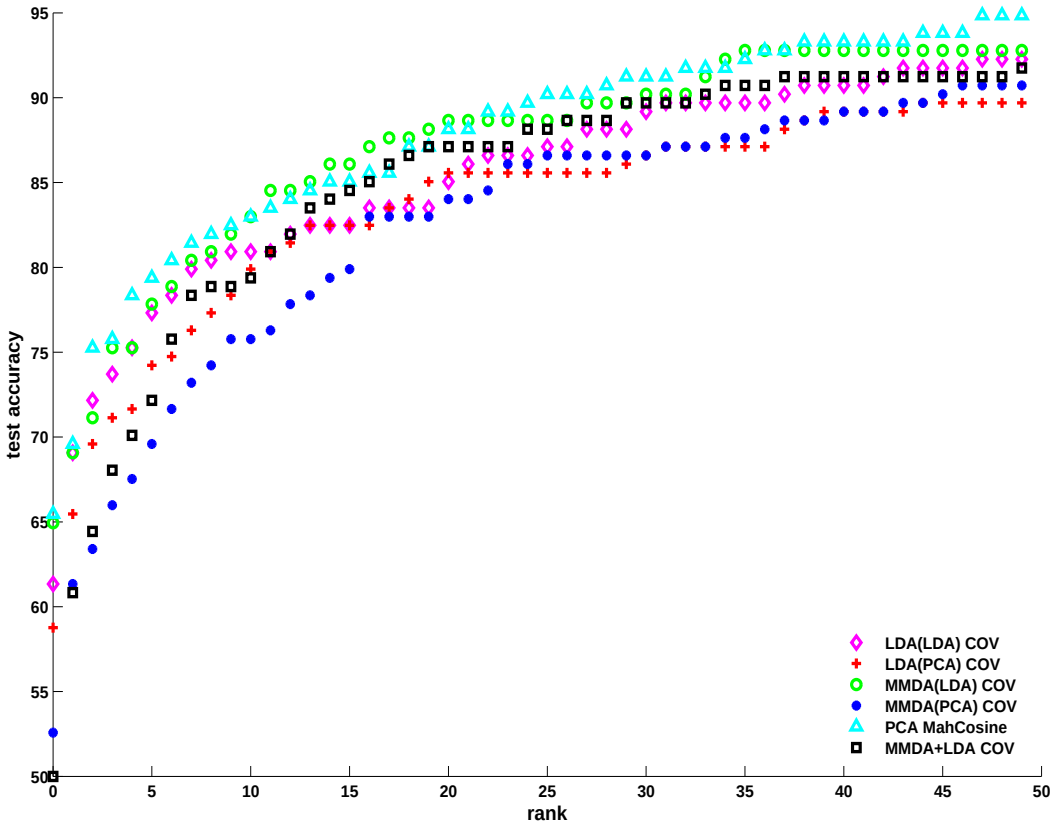

Figure 2.9: Results obtained for the standard probe set FAFC of FERET.

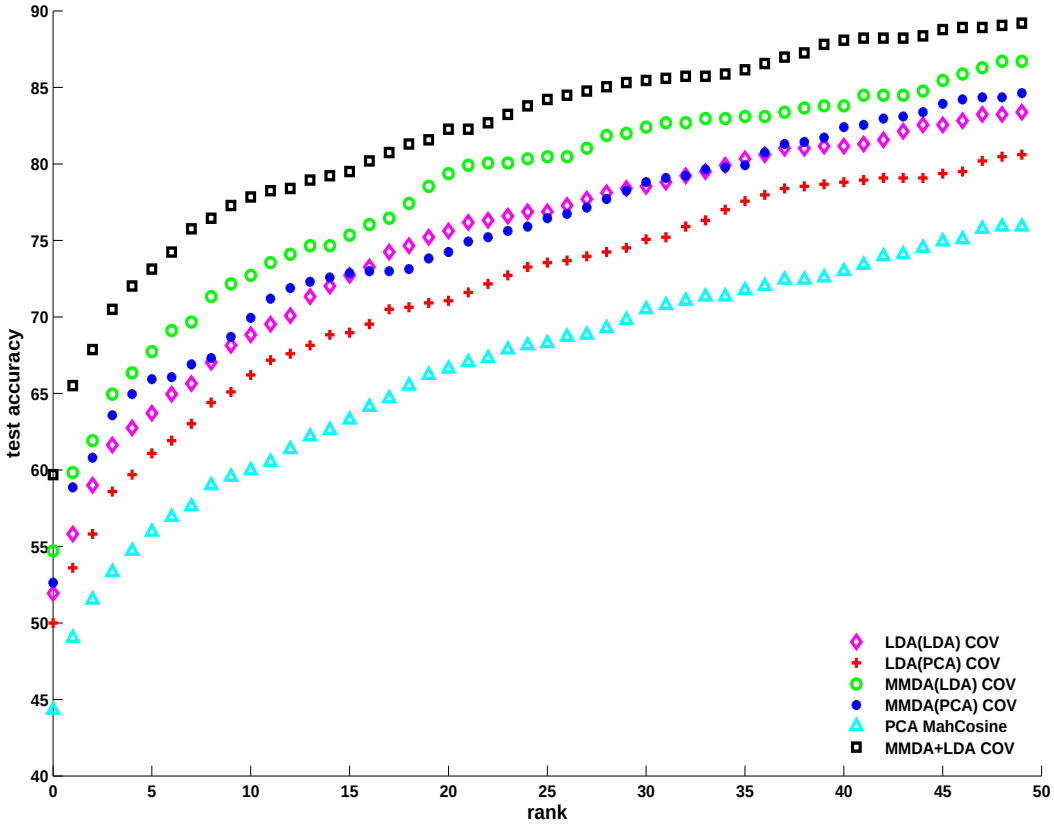

Figure 2.10: Results obtained for the standard probe set DUP1 of FERET.

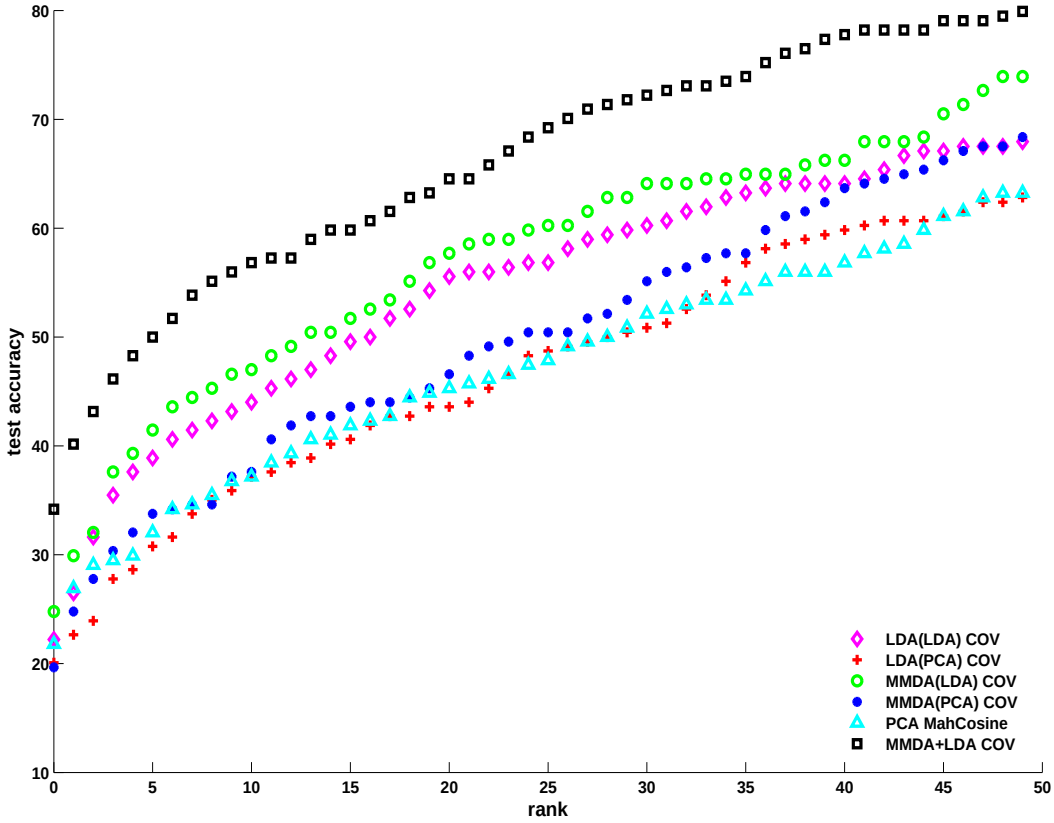

Figure 2.11: Results obtained for the standard probe set DUP2 of FERET.

achieved the best performance on all the tests. On FAFB all the algorithms achieved similar performances. The performance of MMDA(PCA) is significantly worse than that of MMDA(LDA) (especially for the harder probe sets) – in line with our expectation that LDA should yield better individual groupings than PCA. On FAFB and FAFC the best performance next to MMDA(LDA) was achieved by PCA MahCosine (i.e. a special Eigenface method), which is still hard to compete with on these test sets. Note, however, the serious degradation of the performance of this method on the "harder" test sets DUP1 and DUP2. It appears that rejecting noise (the objective of PCA) is the most useful when images are taken close to each other in time (and hence possibly lie in a more concise subspace). Quite surprisingly the performance of LDA(LDA) comes closest to the performance of the MMDA algorithms. Still, MMDA-based algorithms have a considerable advantage over LDA(LDA) on all the datasets.

2.7.3 Experiments with the one image per person test set.

In face recognition one particularly critical issue is the number of images per person available in the training set. In a typical application only a few images might be available for each person. In addition we should not expect to rerun the feature extraction part when a new person is incorporated into the database. Hence we ran further tests in

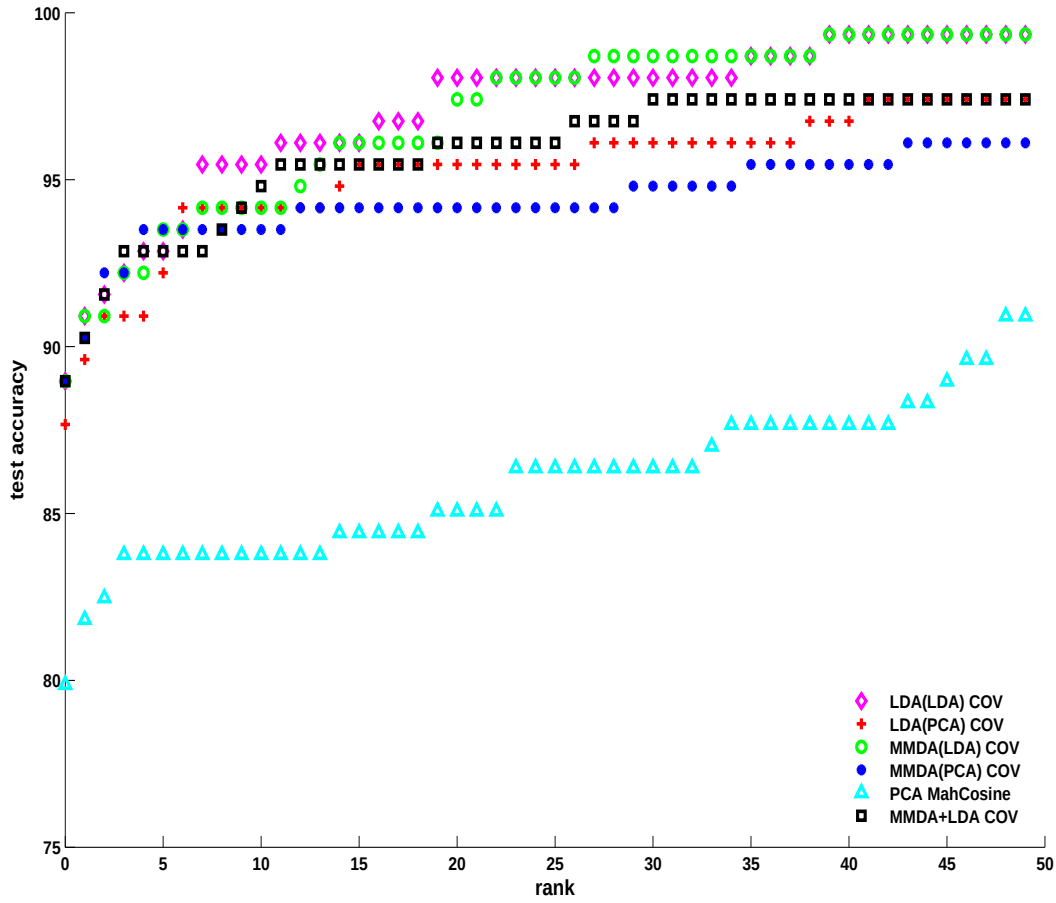

Figure 2.12: Results obtained for the one-image-per person test.

order to evaluate whether our algorithm can indeed suppress intra-personal changes while enhancing extra-personal differences under such stringent conditions. We took 200 persons and divided them into two disjoint sets: a set of training images (this set contained 70% of the images) and a set used for testing (this one had one image in the gallery per person, and as many images as the person had in the probe set). The results obtained are shown in Fig. 2.12 above. Here, MMDA(LDA) COV still came out the best, followed by LDA(LDA) COV.

2.8 Conclusions

One common feature of PCA, LDA and SDA is that they require finding a number of principal eigenvectors of a matrix whose dimension scales with the dimensionality of the input space in the linear case, and scales with the number of input patterns in the non-linear case. MMDA requires the solution of constrained quadratic optimization problems. As a result, in the case of non-linear feature extraction our method extracts sparse solutions, whilst the kernelized versions of PCA, LDA and SDA extract dense solutions (when no additional 'tricks' are used). The maximum number of features

derived using LDA is actually the minimum of the dimensionality of the space and the number of classes minus one. For high dimensional spaces with a few classes this limits the use of LDA [40]. Unlike the standard LDA with (linear) MMDA we can extract as many features as the dimensionality of the pattern space (feature space) allows.

Non-linear MMDA should benefit more from the margin maximizing idea than the linear version as the non-linear version typically works in very high dimensional (sometimes infinite dimensional) feature spaces. This was partially confirmed by our experiments where, for certain datasets, K-MMDA was shown to give excellent results.

In summary, our experiments so far have shown that MMDA can indeed compete with other alternative feature extraction methods. One nice aspect of MMDA is that it can be implemented on top of existing SVM software. Therefore we believe that the proposed method will be a useful tool for researchers using machine learning. In the future we plan to investigate the properties of MMDA more thoroughly (e.g. in multi-class problems). Extensions that make use of different norms penalizing w may also be of interest.

In this chapter we described the application of MMDA together with a clever preprocessing method to face recognition tasks. MMDA is a method that is most suitable for binary classification problems where many samples are available for each of the classes. Since face recognition lies on the opposite end of the spectrum of classification problems (many classes, few samples per class) MMDA cannot be used directly with face recognition. The preprocessing method proposed is capable of creating a sufficiently large number of independent groupings of subjects such that subjects within the same group have similar images and the corresponding images can be fed into MMDA that returns some features.

In the face recognition experiments we found that the performance of MMDA rivals that of the best alternative methods on all the tests that we tried. Of particular interest is the result of a test where each person in the training set had a single image and persons in the test set had no images in the training set used to tune the features. It was found that MMDA performed the best on this test – the significance of this result is that situations like the tested one are likely to be found in practice.

We think that the results obtained so far are encouraging. However, there remain some important open questions of course. It would be important to examine further features such as noise resistance of the results – this could be done using existing tools from the CSU toolkit. Moreover, the second experiment involved only 200 persons due to limited time and resources. It would be useful to know whether the advantages of using MMDA(LDA) are retained if the number of samples in the training set were to be increased. Also, there are a number of other ways to create (balanced) binary problems, e.g. by clustering the image space and then combining nearest neighbour clusters. It would be interesting to find out what the performance of MMDA/LDA is with such

groupings (see [76] for such an approach applied to LDA). Then as argued in [43] a better input representation should have a significant impact on the performance of the algorithms as well. It would also be interesting to try the method in other domains like text classification that share some of the characteristics of face recognition problems.

Chapter 3

Feature Extraction for Regression Problems

In this chapter we consider two novel kernel machine based feature extraction algorithms in a regression settings. The first method is derived from the principles underlying the Maximum Margin Discimination Analysis (MMDA) algorithm. However, here it is shown that the orthogonalization principle employed by the original MMDA algorithm can be motivated by using the well-known ambiguity decomposition, thus providing a firm ground for the good performance of the algorithm. The second algorithm combines kernel machines with average derivative estimation and is derived from the assumption that the true regressor function depends only on a subspace of the original input space.

3.1 Introduction

Let us consider regression problems, where the data (X_i, Y_i) are independent, identically distributed random variables, L is loss function such as e.g. quadratic loss function

$$L(y, z) = (y - z)^2, \tag{3.1}$$

and we seek to determine the regressor

$$f(x) = \underset{y}{\operatorname{argmin}}\, E[L(Y, y)|X = x]. \tag{3.2}$$

In the case of the quadratic loss function

$$f(x) = E[Y|X = x]. \tag{3.3}$$

Here

$$X \sim X_i,\ Y \sim Y_i. \tag{3.4}$$

Let us first consider the model

$$Y = \sum_i \beta_i g_i(X) + \epsilon, \tag{3.5}$$

where

$$g_i : X \to \mathbb{R} \tag{3.6}$$

are unknown functions, and ϵ is noise variable, independent of Y, X. We shall consider estimating g_i by means of an iterative procedure. One view of the model is then to treat the

$$Y = \beta^T \gamma + \epsilon \tag{3.7}$$

as a linear regression problem, where $\gamma = (g_1, \ldots, g_m)$.

3.2 Feature Extraction Based on Ambiguity decomposition

In this section – based on the Ambiguity decomposition formalism – we propose a feature extraction method called 'decorrelated learning', which is defined directly for regression.

3.2.1 Ambiguity decomposition

Now we shall assume that the vector β (cf. Eq. 3.7) is such that $0 \leq \beta \leq 1$, $\beta^T e = 1$, where $e = (1, 1, \ldots, 1)^T$, i.e., the output can be obtained as a noisy convex combination of the 'features' $g_1(X), \ldots, g_m(X)$. We shall further assume that the loss function is the quadratic loss.
Let $g = \sum_i \beta_i g_i$, f arbitrary. Then, it is not hard to see that

$$\begin{aligned}(g(x) - f(x))^2 = &\sum_i \beta_i (g_i(x) - f(x))^2 \\ &- \sum_i \beta_i (g_i(x) - g(x))^2.\end{aligned} \tag{3.8}$$

Therefore,

$$\mathrm{Loss}(g) = \sum_i \beta_i \,\mathrm{Loss}(g_i) - \sum_i \beta_i E[(g_i(X) - g(X))^2] \tag{3.9}$$

and

$$\mathrm{Loss}(g) = E[(g(X) - f(X))^2]. \tag{3.10}$$

This formula, first given in [38] is called "ambiguity decomposition" (AD), since

$$\sum_i \beta_i E[(g_i(X) - g(X))^2] \tag{3.11}$$

can be viewed as the "ambiguity" of the ensemble $g_1(X), \ldots, g_m(X)$.

3.2.2 Decorrelated Learning

According to AD the ensemble loss can be decreased if the ambiguity of the ensemble is maximized whilst keeping the loss of the individual members low.

Now, we obtain easily

$$\sum_i \beta_i E[(g_i(X) - g(X))^2] = \sum_i (\beta_i^2 - \beta_i)\Big(E[g_i(X)]^2 + \mathrm{Var}[g_i(X)]\Big) - \sum_{i \neq j} \beta_i \beta_j \,\mathrm{Cov}(g_i(X), g_j(X)).$$

Therefore, given two ensembles (g_i), $(\hat{g}_i)$ satisfying

$$E[g_i(X)] = E[\hat{g}_i(X)] \tag{3.12}$$

and

$$\mathrm{Var}[g_i(X)] = \mathrm{Var}[\hat{g}_i(X)], \tag{3.13}$$

if

$$\sum_{i \neq j} \beta_i \beta_j E[g_i(X)\, g_j(X)] < \sum_{i \neq j} \beta_i \beta_j E[\hat{g}_i(X)\, \hat{g}_j(X)] \tag{3.14}$$

then

$$\mathrm{Loss}(g) < \mathrm{Loss}(\hat{g}). \tag{3.15}$$

The assumption of equal expected values and variances is motivated by assuming that each g_i should match the regressor function f as closely as it is possible and hence the expected value and variance of $g_i(X)$ are controlled by this desire.

As a conclusion, we have that a small ensemble loss is to enforce orthogonality:

$$E[g_i(X)g_j(X)] = 0, \quad i \neq j. \tag{3.16}$$

In what follows we call methods that enforce this condition *decorrelated learning* (DL) methods.

3.2.3 Kernel Machines

Now, let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a positive definite kernel, $\mathcal{H}$ be the RKHS corresponding to k. Let $\{(x_i, y_i)\}_{i=1}^n$ denote the observed data (again, x_i, y_i are i.i.d.) and let $L(y, z)$ be a loss function, e.g.

$$L(y, z) = (y - z)^2, \; f \in \mathcal{H}. \tag{3.17}$$

Define

$$R(f) = \frac{1}{n} \sum_{i=1}^{n} L(f(x_i), y_i) + \lambda \|f\|_2, \tag{3.18}$$

where $\|f\|_2$ is in the norm of $\mathcal{H}$ (i.e. this is ridge regression in the case of the quadratic loss). By the "Representer Theorem" of Wahba [71] $f \in \operatorname{span}(\Phi)$, where

$$\Phi = (\phi_1, \ldots, \phi_n) \tag{3.19}$$

and $\phi_i : \mathbb{R}^d \to \mathbb{R}$ is defined by

$$\phi_i(x) = k(x_i, x). \tag{3.20}$$

E.g. assume $f = \Phi\alpha$ for some $\alpha \in \mathbb{R}^n$. Equation (3.18) can be solved by

$$R(\alpha; X; k) = \frac{1}{n} \sum_{i=1}^{n} L((\Phi\alpha)(x_i), y_i) + \lambda \alpha^T K \alpha, \tag{3.21}$$

where

$$K_{ij} = k(x_i, x_j) \quad \text{and} \quad X = (x_1, \ldots, x_n). \tag{3.22}$$

When the dataset X and the kernel k are fixed we will often write $R(\alpha)$ instead of $R(\alpha; X; k)$. Similarly, when the kernel is fixed we will use $R(\alpha; X)$.

Now assume $g_i = \Phi\alpha_i$, $g_j = \Phi\alpha_j$. How does (3.16) look like in this case? By replacing the expectation operation with the empirical mean we obtain

$$0 = \sum_{k=1}^{n} g_i(x_k)g_j(x_k) = \sum_{k=1}^{n}\sum_{i,j} \alpha_{ki}\alpha_{kj}\phi_i(x_k)\phi_j(x_k)$$
$$= \sum_{i,j}\sum_{k=1}^{n} \alpha_{ki}\alpha_{kj}k(x_i,x_k)k(x_j,x_k) = \alpha_i^T K^2 \alpha_j.$$

Therefore an iterative procedure that optimizes $R(\alpha)$ and respects the orthogonality criterion (3.16) is as follows: Given $\alpha_1, \ldots, \alpha_i$, let

$$\alpha_{i+1} = \underset{\alpha}{\operatorname{argmin}}\{\, R(\alpha) \,|\, \alpha_j^T K^2 \alpha = 0,\, 1 \leq j \leq i \,\}. \tag{3.23}$$

Once $\alpha_1, \ldots, \alpha_k$ are computed for some $k > 0$, one may estimate the optimal mixing coefficients β_i by e.g. ordinary or regularized (linear) least squares.

Observe that this is just the algorithm originally proposed for classification in [31] under the name MMDA with the modification that the orthogonality criterion

$$\alpha_j^T K \alpha = 0 \tag{3.24}$$

used by MMDA is replaced with

$$\alpha_j^T K^2 \alpha = 0. \tag{3.25}$$

If $k(x,y) = x^T y$ then $g_i(x) = u_i^T x$, for $u_i = \sum_i \alpha_i x_i$. Then $\alpha_j^T K \alpha = 0$ is equivalent to $u_j^T w = 0$ and $\alpha_j^T K^2 \alpha = 0$ is equivalent to $u_j^T \hat{C} u$, where $\hat{C}$ is the empirical covariance matrix:

$$(n-1)\hat{C} = \sum_{k=1}^{n} x_k x_k^T. \tag{3.26}$$

Therefore, by replacing the ad-hoc Euclidean metric of [31] we get a more principled version of MMDA. When the data covariance matrix is the unit matrix then the two approaches are equivalent. This may explain the success of MMDA on some of the problems it were tested on in [31].

In what follows we call the method obtained by solving (3.23) together with the method used to obtain the mixing coefficients β_i, *decorrelated learning regression* (DLR).

3.2.4 Algorithms

Now let us look at the algorithmic aspect of the optimization problem (3.23). There are two equivalent ways to solve this problem (the appropriate results of [31] extend to the case discussed here with minor changes). The first method is to successively deflate the input data. Let us look at this in the linear case (i.e., $k(x, y) = x^T y$). Let the input data be collected into the matrix $X = (x_1, \ldots, x_n)$. Assume that $w = X\alpha$ is the parameter vector obtained in the last step of the algorithm. Then the deflation of the data X can be done using

$$X' = X - ww^T CX/(w^T Cw). \tag{3.27}$$

Indeed, for each i $(1 \leq i \leq n)$,

$$w^T Cx_i' = 0, \tag{3.28}$$

which follows immediately since

$$\begin{aligned} w^T CX' &= w^T CX - (w^T Cw)\,(w^T CX)/(w^T Cw) \\ &= w^T CX - w^T CX \\ &= 0. \end{aligned} \tag{3.29}$$

In the non-linear case it can be shown that the solution of $R_X(\alpha)$ depends on the data X only through the kernel matrix

$$K = \Phi^T \Phi. \tag{3.30}$$

Therefore, an algorithm corresponding to the deflation approach in the non-linear case can be obtained by looking at how K is transformed by the deflation now carried out in the feature space.

The non-linear counterpart of (3.27) is obtained by replacing w with $\Phi\alpha$, X with Φ, and using

$$C \propto \Phi\Phi^T \tag{3.31}$$

we have that

$$\begin{aligned} \Phi' &= \Phi - \Phi\alpha\alpha^T \Phi^T C\Phi/(\alpha^T \Phi^T C\Phi\alpha) \\ &= \Phi - \Phi\alpha\alpha^T K^2/(\alpha^T K^2 \alpha), \end{aligned}$$

where we have used $\Phi^T\Phi = K$. Clearly, we still have $\$\alpha^T \Phi^T C\Phi' = 0$. Using this

equation we get

$$(\Phi')^T\Phi'(=K') = K - u\alpha^T K - K\alpha u^T + u\alpha^T K\alpha u^T, \tag{3.32}$$

where

$$u = K^2\alpha/(\alpha^T K^2\alpha), \tag{3.33}$$

i.e., K' can be expressed as a function of K and α (we assume that $\alpha^T K^2\alpha \neq 0$ which is equivalent to $\alpha \neq 0$ since K (and hence K^2) is positive definite.)

The other approach is obtained by noting that the solution of (3.23) can be obtained by solving the Langrangian dual of the quadratic programming problem (3.23). For this, assume that the solutions up to step i are obtained in the form ΦA_i where we have collected the vectors $\alpha_1, \ldots, \alpha_i$ into the matrix A_i. Also, consider now the ϵ-loss of function of Vapnik [68]:

$$L(y, z) = \max(0, |y - z| - \epsilon). \tag{3.34}$$

It is relatively easy to derive that the problem reduces to the following quadratic programing problem:

$$\begin{aligned} L(\alpha, \alpha^*, \beta) = &-\frac{1}{2}(\alpha - \alpha^*)^T K(\alpha - \alpha^*) \\ &- (\alpha - \alpha^*)^T K^2 A_i\beta \\ &- \frac{1}{2}\beta^T A_i K^3 A_i\beta + (\alpha - \alpha^*)^T y \\ &- \epsilon(\alpha + \alpha^*)^T e \rightarrow \max \\ &\text{s.t. } 0 \leq \alpha, \alpha^* \leq Ce, \end{aligned} \tag{3.35}$$

where $e = (1, 1, \ldots, 1)^T$. Now the Langrangian to consider has the following form:

$$L(\alpha, \gamma) = \alpha^T(\lambda I + K)K\alpha - 2\alpha^T Ky + \alpha^T K^2 A\gamma$$

that should be minimized with respect to α and maximized with respect to γ. Differentiating L w.r.t. α we get that γ^* is such that

$$K^2 A\gamma^* = -2(\lambda I + K)K\alpha - 2Ky.$$

Therefore we can get the optimal α by minimizing

$$L(\alpha, \gamma^*) = -\alpha^T(n\lambda I + K)K\alpha. \tag{3.36}$$

3.3 Kernel Average Derivative Estimation

The other class of algorithms we consider assumes that the unknown regressor function f can be written in the form

$$f(x) = f_0(Bx) \tag{3.37}$$

for some matrix B that projects the d dimensional inputs onto an m dimensional subspace with $m \ll d$ (i.e. $BB^T = I_m$). Here f_0 is an unknown *link* function. Note that this representation is not unique. In the statistical literature models where the regressor function satisfies (3.37) are called *multi-index regression* models. Model (3.37) is a rather general expression of the hypothesis that all the information about f is concentrated in a low-dimensional projection Bx. Our goal here is to find the effective dimension m and to describe the effective dimension reducing space $\mathcal{S}$ [41] in the form of

$$\mathcal{S} = \Im B^T. \tag{3.38}$$

There are many approaches to estimate an effective dimension reducing space. One of the earliest approaches is principal component analysis (PCA). However, PCA does not take into account the dependent variables and hence cannot be expected to work in general. Under some restrictive assumptions the authors in [41] propose the so called sliced inverse regression approach. A modification of this method (principle Hessian directions) is explored in [42] and [14]. Here we propose to use the average derivative estimation based on [60] with kernel machines.

The basic idea of average derivative estimation is as follows: Consider the derivative of f (here and in what follows we assume that f is sufficiently smooth). Easily,

$$\frac{d}{dx} f(x) = B^T f_0'(Bx),$$

where $f_0' \in \mathbb{R}^m$ is the derivative of f_0. Therefore we get that for all $x \in \mathbb{R}^d$ and for

$$F(x) \stackrel{\text{def}}{=} \frac{d}{dx} f(x)$$

we have $F(x) \in \mathcal{S}$.

The basic idea now is to estimate f using a non-parametric estimator. Let $\hat{f}$ denote such an estimate obtained and let $x_1, \ldots, x_n$ be the data points used. Then define

$$\hat{F}(x) = \frac{d}{dx} \hat{f} \tag{3.39}$$

and compute the eigenvalue decomposition of

$$M = \sum_i \hat{F}(x_i)\hat{F}(x_i)^T, \tag{3.40}$$

i.e. let

$$M = U^T \Lambda U, \tag{3.41}$$

where U is an orthonormal matrix with its columns being the eigenvectors of M and Λ is a diagonal matrix where we assume that the eigenvalues of Λ are given in the diagonal in a decreasing order. If $\hat{F} = F$ then it is easy to see that only the first m eigenvalues of M differ from zero. Since $\hat{F}$ is only an approximation of F we may expect that M will have more than m non-zero eigenvalues. However, the hope is that the dimensionality of the effective dimension reducing subspace can be recovered by detecting a gap in the spectrum.

We should note here that if Z is an $n \times n$ orthogonal matrix and

$$\Psi = (F(x_1), \ldots, F(x_n)) \tag{3.42}$$

then, in general, the subspace spanned by the columns Ψ and ΨZ is the same. This idea can be used to get a more stable estimate of the subspace. However, in this work we have not used this idea.

Once estimates of m and B are obtained the procedure can be repeated but now for the projected data $(Bx_1, \ldots, Bx_n)$. Such an iteration is especially advantageous when $m \ll d$ and since most estimation techniques are sensitive to the dimensionality of the input data.

Here we propose to use kernel machines to obtain $\hat{f}$, an estimate of f. We shall call the resulting method KADE (Kernel based Average Derivative Estimation). The choice of using kernel machines is motivated by the widely accepted view that kernel machines are less sensitive to the dimensionality of the input space (as they are deemed to work in very high, or even infinite dimensional feature space) which is important in the first step of the algorithm.

3.4 Experiments

We illustrate the effectiveness of the proposed methods through a series of experiments. First we consider decorrelated learning regression (DLR).

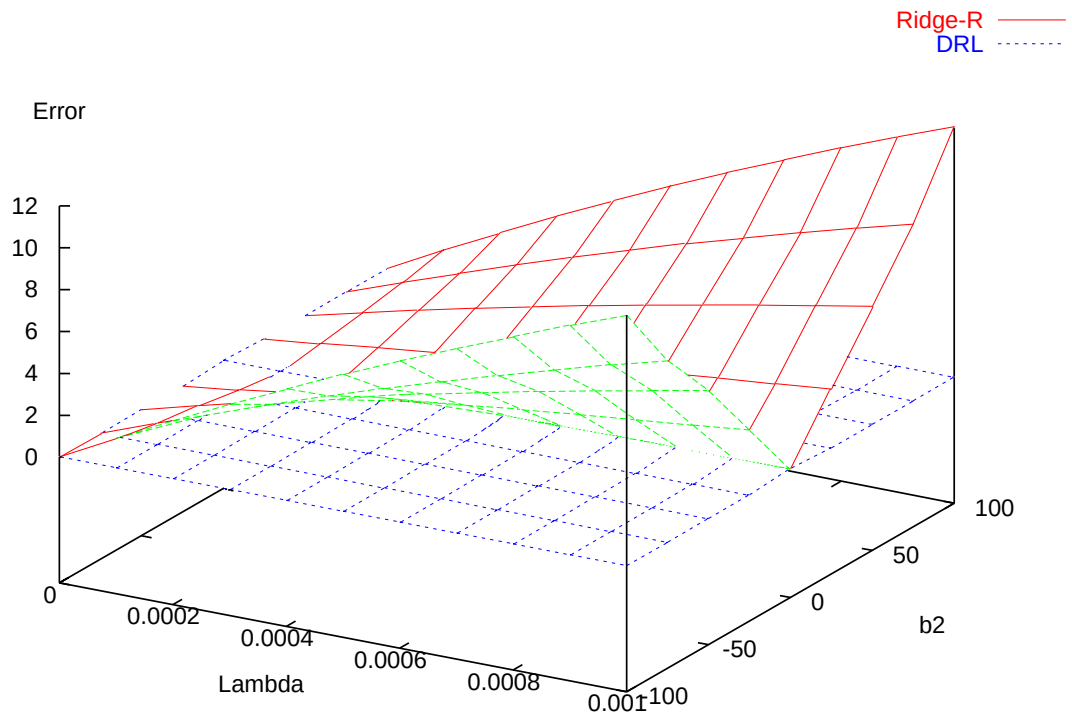

Figure 3.1: Estimation error for ridge regression (regularized least squared) and decorrelated learning regression as a function of the regularization parameter and a model parameter. In this case the model used highly correlated input variables. For more explanation see the text.

3.4.1 Experiments with DLR

In the first experiments we were interested in iwhether DLR has any advantage over regularized least squares regression in the simplest setting possible. Therefore we chose the linear kernel and linear regression problems. Specifically, we have chosen the following problem: Let x, z, ϵ be independent normally distributed random variables with variance $1, 0.1^2$ and 0.01^2, respectively. Then let

$$x_1 = x, \; x_2 = x + z, \; y = x_1 + b_2 x_2 + \epsilon. \tag{3.43}$$

This problem is representative of the situation commonly found in statistical practice when the regression variables are correlated. When the regression variables are highly correlated then ordinary least squares become unstable. One remedy is to use regularization that we consider here. In the experiments we have varies λ, the regularization parameter and b, the second regression coefficient of the model. We have also varied the number of points used for training in a wide range from 10 to 10^5. The results did not show a strong dependence on the number of points used. The square root of the average squared loss was measured on an independent test set of 2000 points. The plot of the dependency of this error measure on λ and b_2 is shown in Figure 3.1. It can

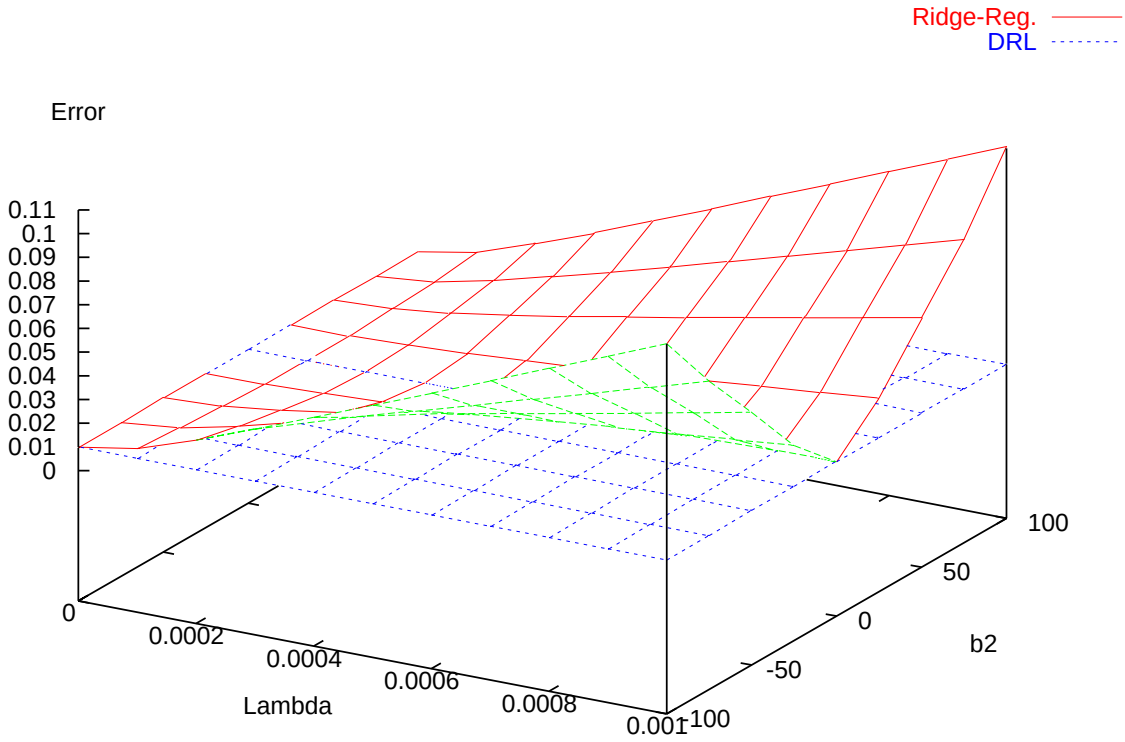

Figure 3.2: Estimation error for ridge regression (regularized least squared) and decorrelated learning regression as a function of the regularization parameter and a model parameter. In this case the model used uncorrelated (actually independent) input variables. For more explanation see the text.

be observed that in the case of ridge regression the error grows linearly both with b_2 and λ, whilst it stays close to the optimum for DLR, independently of b and λ (note that DLR used the same regularization parameter λ used by ridge regression).

One may object that in the case of highly correlated independent variables principle component regression (PCR) or partial least squares regression (PLS) should be used to find out the 'latent' variables of the data first. Hence, we have also experimented with decorrelated inputs as it is known that both PCR and PLS would decorrelate the inputs [26].[1] In this case we used

$$x_1, x_2 \sim N(0,1) \quad \text{and} \quad y = x_1 + b_2 x_2 + \epsilon. \tag{3.44}$$

The results of this experiment can be seen in Figure 3.2. As can be expected, the error of ridge regression is much smaller than in the previous case. However, it is still an order of magnitude higher than for the optimum. Note that DLR still behaves the same as in the previous example.

[1] It is not that widely known that PLS can also be formulated in terms of the principal component analysis of the data. However, in [26] it is shown that this is the case and the feedback from the dependent variables is present only in a weak form.

$(\lambda = 10^k)$ k	-6	-5	-4	-3	-2
LS-SVM	84.8 ± 11.0	84.7 ± 11.0	83.7 ± 10.9	76.9 ± 10.2	64.6 ± 8.2
DLR-I	74.0 ± 10.3	36.5 ± 6.0	15.9 ± 1.3	$\mathbf{12.9\pm2.7}$	13.1 ± 3.9
DLR	81.0 ± 10.6	56.9 ± 8.0	17.7 ± 3.2	13.2 ± 3.0	13.2 ± 3.1
$(\lambda = 10^k)$ k	-1	0	1	2	3
LS-SVM	58.9 ± 6.4	47.1 ± 5.5	33.3 ± 4.2	26.2 ± 3.3	22.5 ± 4.5
DLR-I	13.3 ± 4.1	13.4 ± 4.1	13.4 ± 4.1	13.4 ± 4.1	13.4 ± 4.1
DLR	13.2 ± 3.2	13.2 ± 3.2	13.2 ± 3.2	13.2 ± 3.2	13.2 ± 3.2
$(\lambda = 10^k)$ k	4	5	6		
LS-SVM	17.0 ± 2.8	15.2 ± 3.0	17.8 ± 6.5		
DLR-I	13.4 ± 4.1	13.4 ± 4.1	13.4 ± 4.1		
DLR	13.2 ± 3.2	13.2 ± 3.2	13.2 ± 3.2		

Table 3.1: Comparison of DLR and LS-SVM on the Boston Housing data. Columns correspond to different regularization parameters. LS-SVM stands for Least Squares Support Vector Machines, DLR-I is DLR with the data covariance matrix replaced by the identity matrix.

These experiments show that DLR is potentially more efficient than PLS or PCR and is rather insensitive to the actual value of the regularization parameter.

The results of these experiments promise that DLR is a competitive method and that it should be less sensitive to the actual value of the regularization parameter. Such a method is highly desirable as currently the only known sound method to determine good regularization parameters is by time-consuming cross-validation.

Next we ran experiments on the Boston Housing Data database (506 samples with 20 dimensional input variables). In this case we used least-squares ridge regression with 3rd order cosinus polynomial kernels. DLR used ridge regression for estimating the mixture parameters. Again we have varied λ and observed the error rate. Error was measured as the relative mean squared error, using 5-fold cross-validation. On the top of the extracted features we trained a linear ridge regression model. Results are shown in Table 3.1. Given the table one may conclude that the tolerance of DLR to the regularization parameter is indeed larger than that of LS-SVM alone. Also, notice that we have obtained consistently better results with DLR than with LS-SVM with identical settings.

We have also compared DLR when in the orthogonality criterion the data covariance matrix was replaced by the identity matrix. The results obtained did not show any significant difference to the case when the data covariance matrix was used. Further investigation is needed to find out the reason of this. Our current belief is that the covariance matrix of the data (in the appropriate RKHS) could be close to the identity matrix so that the two metrics induced are not much different.

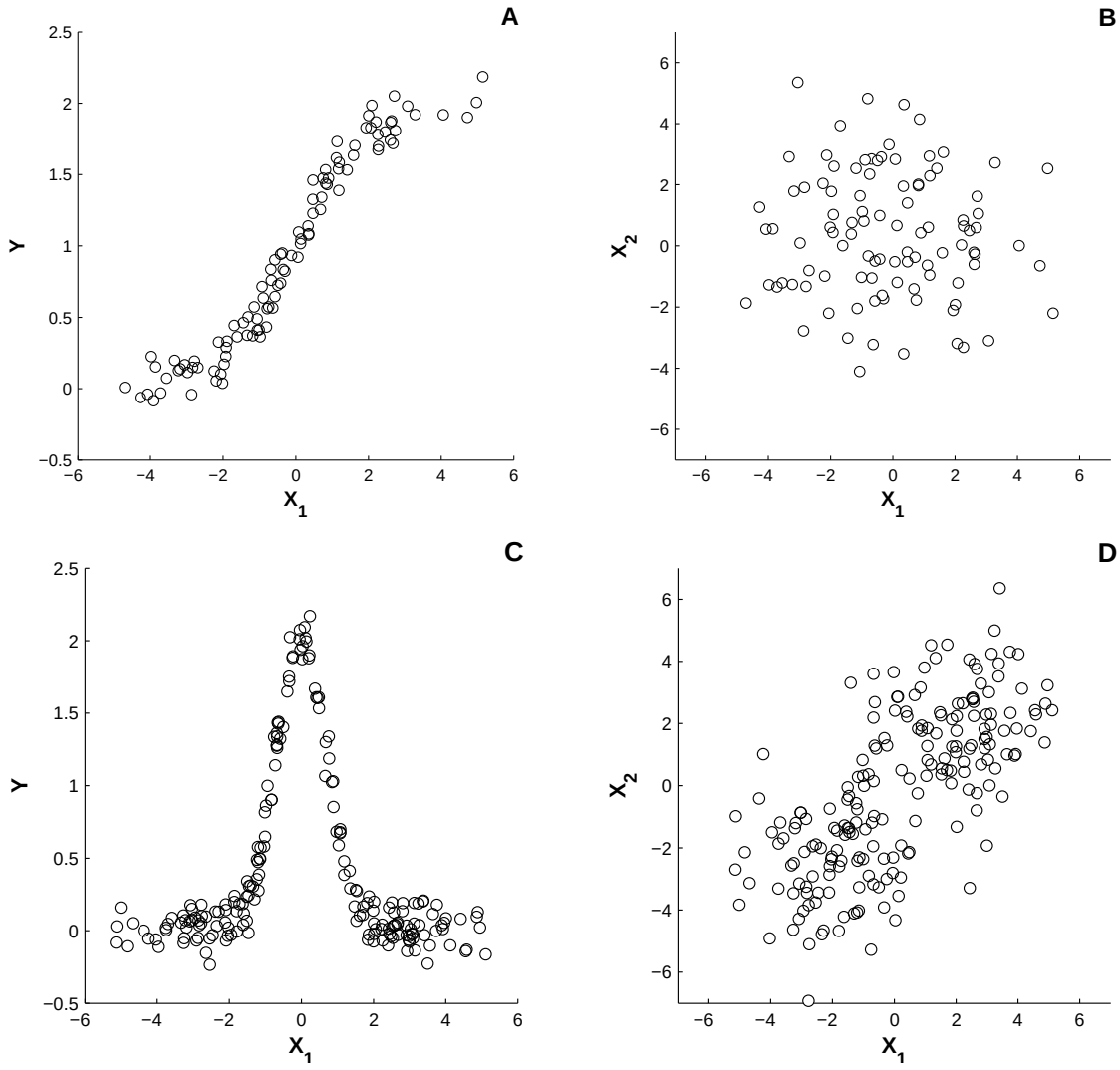

Figure 3.3: Datasets used for experimenting with KADE. For both datasets the dependent variable depends only on the first input variable.

3.4.2 Experiments with KADE

In these experiments we used the synthetic datasets of [24]. The first dataset has 100 samples and is two dimensional,

$$X_1, X_2 \sim N(0, 2.5^2) \tag{3.45}$$

and

$$Y = 1/(1 + \exp(-X_1)) + \epsilon, \tag{3.46}$$

with

$$\epsilon \sim N(0, 0.1^2). \tag{3.47}$$

We have used least-squares SVMs with 3rd order polynomial kernels. The eigenvalues we obtain are 0.0317,0.0002, i.e., $\lambda_2/\lambda_1 < 0.01$. The angle between $b = [1, 0]^T$ and the first eigenvector is -0.0100 (measured in radian). The second dataset is also two dimensional, it has 200 samples and X_1,X_2 are as before, but now

$$Y = 2 * \exp(-X_1^2) + \epsilon. \tag{3.48}$$

In this case we tried LS-SVM with the Gaussian RBF kernel with $\sigma = 0.1$. The eigenvalues are 0.1728 and 0.02209. The angle between b and the first eigenvector is 0.0091. Note that according to [24] SIR, PhD, CCA (Canonical Correlation Analysis), and PLS yield good acceptable results on the first dataset, whilst they do not perform very well on the second (the smallest absolute angle is obtained for CCA and it is 0.1818). For KDR the respective values are -0.0014 and 0.0052. The datasets are shown in Figure 3.3. The third dataset has 17 dimensions, the number of samples is 300. All variables are uniformly distributed on $[0, 1]$. The regression problem is given by

$$Y\ 0.9 * X_1 + 0.2 * 1/(1 + X_{17}) + \epsilon, \tag{3.49}$$

with

$$\epsilon \sim N(0, 0.01^2). \tag{3.50}$$

For the first and second eigenvector we get the multiple correlation coefficients

$$R(e_1) = 0.9874, \quad R(e_{17}) = 0.9842, \tag{3.51}$$

where e_i is the unit vector with its ith coordinate being 1.[2] Despite the fact that some of our results are slightly worse than those obtained with KDR, note that we ran only a *single* iteration of KADE. We expect that subsequent iterations would improve our results considerably.

[2]This result was obtained again with a Gaussian RBF kernel with $\sigma = 10$.

3.5 Conclusions

In this chapter we have proposed two methods for feature extraction for regression problems. The first method, decorrelated learning regression (DLR), can be viewed as a better motivated (and generalized) version of MMDA, a feature extraction method [31] that has shown some merit in some classification tasks. DLR estimates decorrelated features by solving a series of regression problems. We have shown experimentally that DLR is more robust to the choice of the regularization parameter than ridge-regression. Also, based on our experiments we have argued that it can be expected that the performance of DLR would be better than that of PLS [74] or PCR when PLS/PCR would be combined with regularization. One expects that this might be necessary for the purposes of capacity control when working in high or even infinite dimensional Hilbert spaces [55].

The second method we proposed is the adaptation of the "Average Derivative Estimation" algorithm to kernel machine based regression. Based on our experiments it can be expected that this method will be competitive with alternative methods, such as the KDR method of [24], especially since our method is straightforward to implement. Further, the corresponding optimization problem has a convex loss function and thus this method can be expected to be more stable than e.g. KDR. Naturally, there is room for further research with KDR. It would be especially important to try out the iterative version of KADE and to work out theoretically sound stopping conditions.

Part II

Kernel-Based Classification

Chapter 4

Kernel Hyperplane Classifiers

Numerous scientific areas depend on classification and regression methods which may be linear or non-linear. As we demonstrated earlier in the thesis by using the so-called kernel idea, linear methods can be readily generalized to nonlinear ones. The key idea was originally presented in Aizermann's paper [1] and it was successfully applied in the context of the ubiquitous Support Vector Machines (SVM) [69]. The roots of SV methods can be traced back to the need for the determination of the optimal parameters of a separating hyperplane, which can be formulated both in input space or in kernel induced feature spaces. However, optimality can vary from method to method and SVM is just one of several possible approaches.

In this chapter we introduce a new family of hyperplane classifiers. But, in contrast to Support Vector Machines - where a constrained quadratic optimization is used - some of the proposed methods lead to the unconstrained minimization of convex functions while others merely require solving a linear system of equations. In order that the efficiency of these methods could be checked, classification tests were conducted on standard databases. In our evaluation, classification results of SVM were of course used as a general point of reference, which we found were outperformed in many cases.

4.1 Introduction

Without loss of generality we shall assume that, as a realization of multivariate random variables, there are m-dimensional real attribute vectors in a compact set $\mathcal{X}$ over $\mathbb{R}^m$ describing objects in a certain domain, and that we have a finite $n \times m$ sample matrix $X = [\mathbf{x}_1, \dots \mathbf{x}_n]^T$ containing n random observations. Let us assume as well that we have an indicator function

$$\mathcal{L} : \mathbb{R}^m \to L \subseteq \mathbb{R}, \tag{4.1}$$

where

$$\mathcal{L}(\mathbf{x}_i) = y_i \tag{4.2}$$

gives the label of the sample $\mathbf{x_i}$, and let us denote the vector $[y_1, \ldots, y_n]^T$ by Y. Here, a finite set L means a classification task. Should L be an infinite set, the task will be a regression problem.

In this chapter we will restrict our investigations only to that of binary classification ($L = \{-1, +1\}$), as multiclass problems can be dealt with by applying binary classifiers [28]. But regression problems will not be entirely excluded here, since binary classifiers will be derived from regression formulae.

4.2 Hyperplane classifiers

Hyperplane classifiers seem to be weak approximators with respect to the separation of positive and negative classes. This is due to the fact that in low dimension spaces, sample sets where linear separation does not work can be easily defined. The kernel idea is designed to overcome this limitation, therefore the examination of hyperplane-based methods becomes interesting.

I introduce three methods in the following. In the first case, I propose to extend hyperplane classifiers with unique loss functions. The second method uses the regression formalism for classification in a way different from the generally documented methods. The third approach – also as the most significant result of the Chapter – is the so-called Minor Component Classifier, which performs classification by treating the input and the output space in a unified space.

4.2.1 Linear classifiers with various loss-functions

Linear classification attempts to separate the sample points with different labels via a hyperplane. A hyperplane is a set of point $\mathbf{z}$:

$$\begin{pmatrix} \mathbf{z}^T & 1 \end{pmatrix} \mathbf{a} = 0 \qquad \mathbf{z} \in \mathbb{R}^m, \quad \mathbf{a} \in \mathbb{R}^{m+1}. \tag{4.3}$$

For a sample $\mathbf{z}$ the left-hand side of Eq. (4.3) is a signed expression with absolute value proportional to the distance from the hyperplane. In addition, the sign of this expression corresponds to the sign of the half-space the point lies in.

A point $\mathbf{x}_i$ is well-separated by a hyperplane with parameter $\mathbf{a}$ if and only if:

$$y_i \begin{pmatrix} \mathbf{x}_i^T & 1 \end{pmatrix} \mathbf{a} > 0 \qquad i \in \{1, \ldots, n\}. \tag{4.4}$$

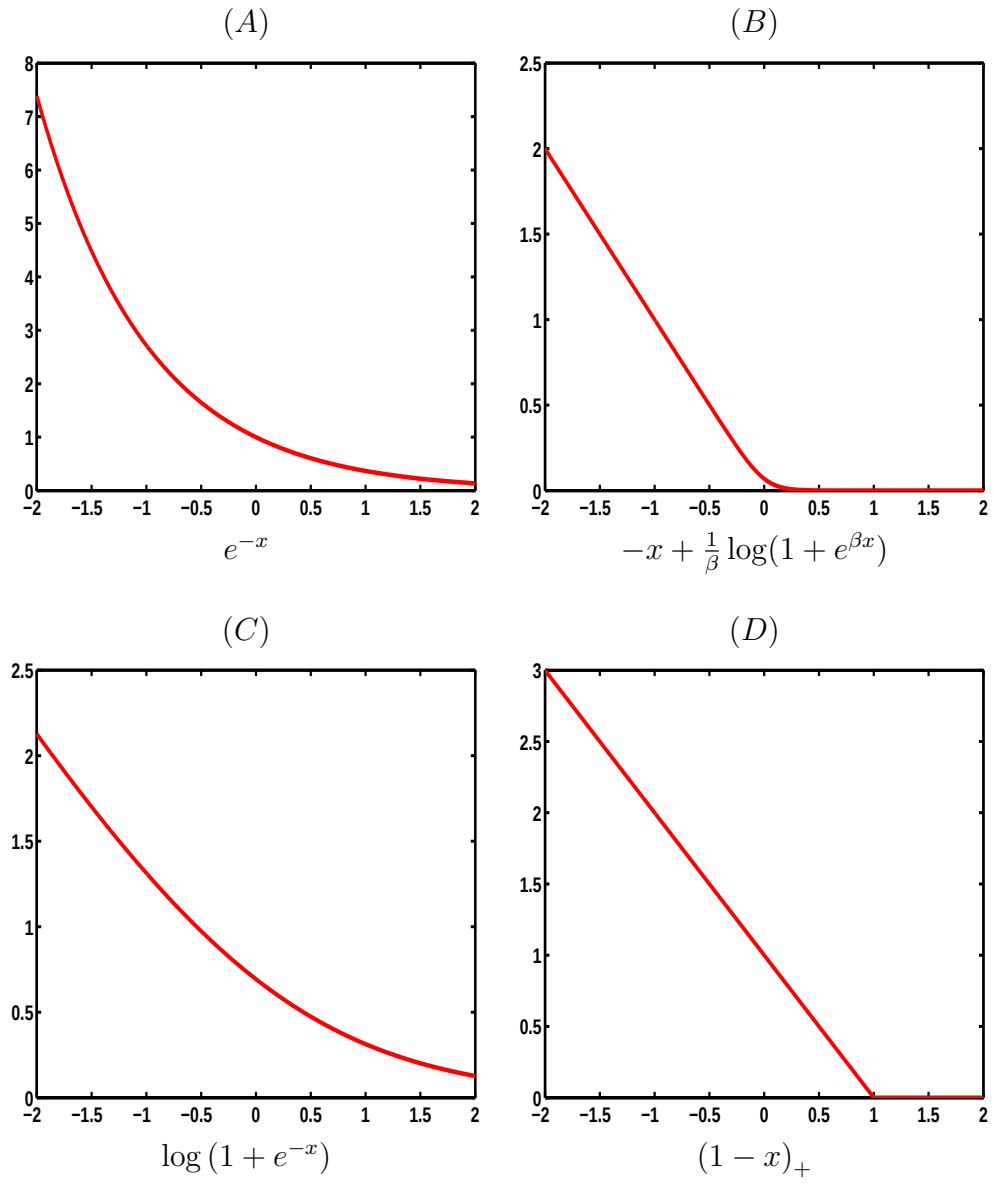

Figure 4.1: Examples for loss functions.

Based on these products a target function - whose lower value indicates a better separation - can be defined:

$$\tau(\mathbf{a}) = \sum_{i=1}^{n} g\left(y_i \begin{pmatrix} \mathbf{x}_i^T & 1 \end{pmatrix} \mathbf{a}\right), \tag{4.5}$$

where $g : \mathbb{R} \to \mathbb{R}$ is a strictly monotonic decreasing function, called a loss function. Of the many possibilities [18], four candidates are shown in Fig. 4.1. We should note here that using a signum-function approximating loss function, the measure estimates the number of poorly separated points when $\alpha \to \infty$.

Minimizing $\tau(\mathbf{a})$ we get an unconstrained minimization of a strictly convex function, which is in marked contrast to the quadratic optimization with constraints in SVM. With a suitably smooth loss function, the gradient vector of $\tau(\mathbf{a})$ will be smooth as

well, hence one can apply quasi-Newton methods or even the Newton iteration method.

After obtaining the optimal parameter of the separating hyperplane the binary classification of an arbitrary point $\mathbf{z}$ can be carried out by:

$$sign\left(\begin{pmatrix}\mathbf{z}^T & 1\end{pmatrix}\mathbf{a}\right). \tag{4.6}$$

4.2.2 Linear regression in classification

Linear regression attempts to optimally fit a hyperplane onto the indicator function $\mathcal{L}$. The indicator function has values $y_1, \ldots, y_n$ at the sample points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ while the regression hyperplane has function values $f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)$, where

$$f(\mathbf{z}) = \begin{pmatrix}\mathbf{z}^T & 1\end{pmatrix}\mathbf{a} \qquad \mathbf{z} \in \mathbb{R}^m, \quad \mathbf{a} \in \mathbb{R}^{m+1}. \tag{4.7}$$

Thus the error of the sample point $\mathbf{x}_i$ can be expressed by

$$\epsilon_i = y_i - f(\mathbf{x}_i) = y_i - \begin{pmatrix}\mathbf{x}_i^T & 1\end{pmatrix}\mathbf{a}. \tag{4.8}$$

The optimal parameter of the regression hyperplane can be obtained by minimizing the following sum:

$$\min_{\mathbf{a}} \sum_{i=1}^{n} \epsilon_i^2 = \min_{\mathbf{a}} \|Y - X_1\mathbf{a}\|_2^2, \tag{4.9}$$

where

$$X_1 = \begin{pmatrix}\mathbf{x}_1^T & 1 \\ \vdots & \vdots \\ \mathbf{x}_n^T & 1\end{pmatrix}. \tag{4.10}$$

The well-known solution of Eq. (4.9) is given by

$$\mathbf{a} = (X_1^T X_1)^+ X_1^T Y, \tag{4.11}$$

where $^+$ denotes the Moore-Penrose pseudo-inverse.

Though the regression makes use of the hyperplane in a different sense from that in the classification problem, the regression-based binary classification of an arbitrary point $\mathbf{z}$ can still be performed in the same way as that for a linear classifier:

$$sign\left(\begin{pmatrix}\mathbf{z}^T & 1\end{pmatrix}\mathbf{a}\right). \tag{4.12}$$

4.2.3 Minor Component Classifier

Let us take the sample points X with the corresponding labels Y, and represent

$$\begin{pmatrix} \mathbf{x}_1^T & y_1 \end{pmatrix}^T, \ldots, \begin{pmatrix} \mathbf{x}_n^T & y_n \end{pmatrix}^T \tag{4.13}$$

as vectors in $\mathbb{R}^{n+1}$. In this extended space a hyperplane with parameter $\bar{\mathbf{a}}$ contains points $\mathbf{z}$ where

$$\begin{pmatrix} \mathbf{z}^T & \mathcal{L}(\mathbf{z}) & 1 \end{pmatrix} \bar{\mathbf{a}} = 0, \qquad \mathbf{z} \in \mathbb{R}^m, \quad \bar{\mathbf{a}} \in \mathbb{R}^{m+2}. \tag{4.14}$$

The distance of $(\mathbf{x}_i \; y_i)$ from the hyperplane is

$$\delta(\mathbf{x}_i, y_i) = \frac{\begin{pmatrix} \mathbf{x}_i^T & y_i & 1 \end{pmatrix} \bar{\mathbf{a}}}{\|\mathbf{a}\|_2}, \tag{4.15}$$

so there exists an optimal hyperplane fitting on the extended sample points with least error:

$$\min_{\bar{\mathbf{a}}} \sum_{i=1}^{n} \delta(\mathbf{x}_i, y_i)^2 = \min_{\bar{\mathbf{a}}} \frac{\bar{\mathbf{a}}^T X_2^T X_2 \bar{\mathbf{a}}}{\bar{\mathbf{a}}^T \bar{\mathbf{a}}} \qquad X_2 = \begin{pmatrix} \mathbf{x}_1^T & y_1 & 1 \\ \vdots & \vdots & \vdots \\ \mathbf{x}_n^T & y_n & 1 \end{pmatrix}. \tag{4.16}$$

It can be proved that eigenvectors of $X_2^T X_2$ are the stationary points of the above function with the corresponding eigenvalues as function values. Thus the solution of the minimization problem can be readily obtained by finding the eigenvector of $X_2^T X_2$ which has the smallest eigenvalue [23].

We should note that the better the fit of a hyperplane onto the points, the lower the deviation of the sample points projections onto the normal vector of the hyperplane. Finding the best hyperplane means performing a Minor Component Analysis (MCA) [44] in the extended space, as MCA searches for directions with a small deviation of the sample points projections.

The binary classification of a point $\mathbf{u}$ in the original space can be performed by computing the absolute distances in the extended space for both labels $\{-1, 1\}$ and probabilities can be assigned to the labels via normalization:

$$P(\mathcal{L}(\mathbf{u}) = 1) = \frac{|\delta(\mathbf{u}, -1)|}{|\delta(\mathbf{u}, 1)| + |\delta(\mathbf{u}, -1)|} \tag{4.17}$$

$$P(\mathcal{L}(\mathbf{u}) = -1) = \frac{|\delta(\mathbf{u}, 1)|}{|\delta(\mathbf{u}, 1)| + |\delta(\mathbf{u}, -1)|} \tag{4.18}$$

4.3 Kernel-based non-linearization

The proposed methods, linear classifiers, linear regression and minor component classifier perform linear separation in the original sample space. Making the separation non-linear with kernels it must be shown that the methods optimal solutions are in the linear subspace of the appropriate extended points:

$$\mathbf{a} = X_1\boldsymbol{\alpha} \quad \boldsymbol{\alpha} \in \mathbb{R}^n, \tag{4.19}$$

and

$$\bar{\mathbf{a}} = X_2\boldsymbol{\beta} \quad \boldsymbol{\beta} \in \mathbb{R}^n. \tag{4.20}$$

Regarding a linear classifier the parameter vector $\mathbf{a}$ can be decomposed into two perpendicular components $\mathbf{a}_1$ and $\mathbf{a}_2$, where the first component lies in the subspace of the extended sample points X_1:

$$\mathbf{a} = \mathbf{a}_1 + \mathbf{a}_2 \qquad \mathbf{a}_1 = X_1\boldsymbol{\alpha}, \quad \boldsymbol{\alpha} \in \mathbb{R}^n, \quad \mathbf{a}_1 \perp \mathbf{a}_2. \tag{4.21}$$

The form of the measure τ then becomes

$$\begin{aligned} \tau(\mathbf{a}) &= \textstyle\sum_{i=1}^n g\left(y_i \begin{pmatrix} \mathbf{x}_i^T & 1 \end{pmatrix} (\mathbf{a}_1 + \mathbf{a}_2)\right) = \\ &= \textstyle\sum_{i=1}^n g\left(y_i \begin{pmatrix} \mathbf{x}_i^T & 1 \end{pmatrix} \mathbf{a}_1 + y_i \begin{pmatrix} \mathbf{x}_i^T & 1 \end{pmatrix} \mathbf{a}_2\right) = \\ &= \textstyle\sum_{i=1}^n g\left(y_i \begin{pmatrix} \mathbf{x}_i^T & 1 \end{pmatrix} \mathbf{a}_1\right), \end{aligned} \tag{4.22}$$

because $\mathbf{a}_2$ is orthogonal to all the extended sample points $\begin{pmatrix} \mathbf{x}_i^T & 1 \end{pmatrix}$.

Because the measure depends only on $\mathbf{a}_1$, the minimization in fact can be performed in the linear subspace of the extended sample points X_1. Actually, this result holds true for the other methods as well.

Utilizing the introduced formulas the solutions of the proposed methods can be found by optimizing $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ respectively:

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^n g\left(y_i \begin{pmatrix} \mathbf{x}_i^T & 1 \end{pmatrix} X_1^T\boldsymbol{\alpha}\right), \tag{4.23}$$

$$\boldsymbol{\alpha} = (X_1^T X_1 X_1 X_1^T)^+ X_1^T X_1 Y, \tag{4.24}$$

$$\min_{\boldsymbol{\beta}} \frac{\boldsymbol{\beta}^T X_2 X_2^T X_2 X_2^T \boldsymbol{\beta}}{\boldsymbol{\beta}^T X_2 X_2^T \boldsymbol{\beta}}. \tag{4.25}$$

Algorithms using only the dot product can be executed in the kernel feature space by

kernel function evaluations alone. The construction of an appropriate kernel function is a non-trivial problem, but there are many good suggestions about the sorts of kernel functions [16; 63; 69] which might be adopted along with some background theory. Among the functions available, the two most popular kernels are:

$$\text{Polynomial kernel:} \qquad k(\mathbf{x},\mathbf{y}) = \left(\mathbf{x}^T\mathbf{y}+1\right)^d, \qquad d \in \mathbb{N} \tag{4.26}$$

$$\text{Gaussian RBF kernel:} \qquad k(\mathbf{x},\mathbf{y}) = e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{r}}, \qquad r \in \mathbb{R}^+ \tag{4.27}$$

Employing the kernel-idea to make the proposed methods (4.23), (4.24) and (4.25) non-linear, we obtain the following three expressions:

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} g\left(y_i \sum_{j=1}^{n} \alpha_i k\left(\begin{pmatrix}\mathbf{x}_i\\1\end{pmatrix}, \begin{pmatrix}\mathbf{x}_j\\1\end{pmatrix}\right)\right), \tag{4.28}$$

$$\boldsymbol{\alpha} = (K^TK)^+K^TY, \tag{4.29}$$

$$\min_{\boldsymbol{\beta}} \frac{\boldsymbol{\beta}^T\bar{K}\bar{K}\boldsymbol{\beta}}{\boldsymbol{\beta}^T\bar{K}\boldsymbol{\beta}}, \tag{4.30}$$

where the matrices K and $\bar{K}$ contain the pairwise dot products of transformed points:

$$K_{ij} = k\left(\begin{pmatrix}\mathbf{x}_i\\1\end{pmatrix}, \begin{pmatrix}\mathbf{x}_j\\1\end{pmatrix}\right), \tag{4.31}$$

$$\bar{K}_{ij} = k\left(\begin{pmatrix}\mathbf{x}_i\\y_i\\1\end{pmatrix}, \begin{pmatrix}\mathbf{x}_j\\y_j\\1\end{pmatrix}\right). \tag{4.32}$$

The solution of (4.30) can be obtained by finding the eigenvector corresponding to the smallest nontrivial eigenvalue of the generalized eigenproblem

$$\bar{K}\bar{K}\boldsymbol{\beta} = \lambda\bar{K}\boldsymbol{\beta}. \tag{4.33}$$

Note here that if the transformed sample points lie entirely on a hyperplane in the space $\mathcal{F}$ then the normal vector of the hyperplane is not in the subspace of the transformed sample points. Thus perfect fitting of the hyperplane is never realized in regression methods nonlinearized with kernels.

		LINEAR CLASSIFIER	LINEAR REGRESSION	MCC	SVM
BUPA					
	train	72.29	71.70	**73.10**	72.40
	test	**65.98**	65.40	62.24	65.60
CHESS					
	train	**100.0**	97.42	95.98	**100.0**
	test	**98.08**	90.73	88.49	**98.08**
ECHO					
	train	**100.0**	92.35	91.57	**100.0**
	test	89.54	89.57	**90.32**	90.10
HHEART					
	train	86.64	85.96	85.27	**87.10**
	test	80.08	79.73	**80.40**	**80.40**
MONKS					
	train	**100.0**	93.35	93.35	**100.0**
	test	87.88	88.81	**89.60**	89.10
SPIRAL					
	train	**100.0**	**100.0**	**100.0**	**100.0**
	test	88.48	87.23	**90.80**	89.20

Table 4.1: The best training and testing results using tenfold cross validations. A set of kernel functions with different parameters were used during the tests, but only the best results are summarized here.

4.4 Experimental Results and Evaluation

When evaluating the efficiency of a new algorithm the usual method is to assess its performance by making use of standard databases. To this end we selected a set of databases again from the gold standard UCI Repository [9]. Namely, we carried out tests using the BUPA liver, chess, echo, Hungarian heart, monks and spiral databases. All sets were normalized so that each feature had a zero mean and unit deviation and we applied a tenfold cross-validation on all the sets.

Since a recent study [28] compared five different Support Vector algorithms using the UCI Repository and concluded that the methods have no significant difference in efficiency, we will employ [13] as the SVM classifier. The numerical results of tenfold cross-validations are shown in Table 4.1, where the best result is emphasized in bold. It confirms that regression based classification methods are indeed just as effective as the original separation algorithms. Moreover, making use of the labels in the regression task with the Minor Component Classifier the usual classification methods were surpassed in many cases so MCC can now be considered as a rival classification method.

4.5 Conclusions

In this chapter, the author introduced a family of hyperplane-based classification methods. The first method combined hyperplane classifiers with various loss functions, while the second one applied a regression approach for classification. In the third case, the author defined the hyperplane-based target function as one building on an eigenvalue-eigenvector solution. It is to be emphasized that elements of the introduced methods significantly rely on dimension-composition approaches. Extending the input vectors with a dimension containing an additional constant - namely 1 - allows the optimisation of the bias parameter of the hyperplane. If we further add the output dimension (i.e., those marked y) to the above, we arrive at a new learning algorithm - Minor Component Classifier - by mapping the hyperplane onto the enlarged sample set. Test results support the application of this novel family of hyperplane classifiers. The most promising member of the described method family is the Minor Component Classifier.

Chapter 5

Convex Machines

Convex formulations for classification problems are widely used in the machine learning community owing to the simplicity of the optimization task. Here we explore this property. First, we define a sufficiently broad convex optimization form for the classification task, which – as special cases – implies numerous popular classifiers such as Logistic Regression, Support Vector Machines and its Smooth or Least Square counterparts. The non-linear Gauss-Seidel method with box constraints (BOX-GS) is able to solve the above optimization problem, i.e. it converges to a single optimum. Although the BOX-GS method is a gradient-based method and converges relatively fast in practice, if we have a large number of parameters to be optimized, applying various heuristic refinements is a reasonable strategy. Hence we will also present a number of modifications on the BOX-GS method which guarantee that the number of nonzero components of the heuristic solution is less than a preset parameter. Numerical results and comparisons demonstrate the effectiveness of the proposed methods on publicly available datasets.

5.1 Introduction

In this chapter we concentrate on two key topics in machine learning: classification and heuristic optimization. Intrinsically the main goal here is to give a new family of heuristic convex optimization methods for large-scale classification problems.

Nowadays the machine learning community is focusing more and more on the convex formulation of classification tasks. This is due to two important consequences: i) the optimization process is not plagued by local minima. ii) even despite the large number of variables to be optimized numerous approximate optimization strategy are feasible in practice. By now, it is well-known from the literature that Support Vector Machines (SVMs), Boosting and its variants such as Logistic Regression Boosting or Adaboost lead to a similar convex optimization problem over a convex domain. The following authors have sought to unify different classification methods using the convex formalism:

R. Schapire et al. first discovered the relationship between boosting and SVMs [20]. T. Evgeniou et al. summarized the theoretical framework for the problem of learning from examples in the context of Regularization Networks and Support Vector Machines [18]. In [58] the authors examined the equivalence of Boosting and SVMs from the linear programming point of view. They developed a general mechanism for converting SVM-like algorithms into boosting-like algorithms and vice versa, which provides a deeper insight into the relationship between the two. Many authors investigated the properties of the convex risk minimization approach to machine learning and its consequences for SVMs [12; 61; 69]. The unification approach we follow in this chapter is similar to the one described in [75] and is motivated by function approximation using a sparse combination of basis functions.

5.2 Convex Machines

Now - as in earlier chapters - consider the problem of classifying *n* points in a compact set $\mathcal{X}$ over $\mathbb{R}^m$, represented by vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$, according to the membership of each point $\mathbf{x}_i$ in the classes $\{1, \ldots, c\}$, as specified by $y_1, \ldots, y_n$. A multiclass problem can be transformed into a set of binary classification tasks – where we usually have y_i in $\{-1, +1\}$ – using various algorithms like the one-against-all method [72] or the output coding scheme [33], say. Hence our investigation here can be restricted to the problem of binary classification without loss of generality.

Now let V be a vector space, which will be viewed as a function space here. Let $S \subset V$ denote a finite set of basis functions

$$S = \{f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})\}, \qquad f_i : \mathcal{X} \to \mathbb{R} \tag{5.1}$$

and let $Span(S)$ stand for the linear subspace spanned by S, that is

$$Span(S) = \left\{ f_{\boldsymbol{\alpha}} : \mathcal{X} \to \mathbb{R} \mid f_{\boldsymbol{\alpha}}(\mathbf{x}) = \sum_{i=1}^{k} \alpha_i f_i(\mathbf{x}),\ \mathbf{x} \in \mathcal{X},\ \boldsymbol{\alpha} \in \mathbb{R}^k \right\}. \tag{5.2}$$

The classification problem may be defined by the following optimization problem

$$\min_{f_{\boldsymbol{\alpha}}(x) \in Span(S)} E_{\mathbf{x},y} L(f_{\boldsymbol{\alpha}}(x), y), \tag{5.3}$$

where L is a loss function measuring the quality of the predictor $f_{\boldsymbol{\alpha}}(x)$ and E denotes the expectation over (x, y). A possible convex restriction of Eq. (5.3) is

$$\min_{f_{\boldsymbol{\alpha}}(x) \in Box(S,\mathcal{B})} E_{\mathbf{x},y} H(y_i f_{\boldsymbol{\alpha}}(x_i)). \tag{5.4}$$

Here the loss function $L(f_{\boldsymbol{\alpha}}(x), y)$ is assumed to be of the form $H(y_i f_{\boldsymbol{\alpha}}(x_i))$, where $H : \mathbb{R} \to \mathbb{R}$ is a twice continuously differentiable, non-negative, decreasing, convex function. Of the many possibilities for the function H four candidates have been plotted in Fig. 4.1 (cf. Chapter 4.). $Box(S, \mathcal{B})$, furthermore, is a box constrained subset of $Span(S)$, i.e.

$$\mathcal{B} = \mathcal{B}_1 \times \cdots \times \mathcal{B}_n \subseteq \mathbb{R}^n \tag{5.5}$$

is a cartesian product space of non-empty intervals and

$$Box(S, \mathcal{B}) = \left\{ f_{\boldsymbol{\alpha}} : \mathcal{X} \to \mathbb{R} \mid f_{\boldsymbol{\alpha}}(\mathbf{x}) = \sum_{i=1}^{k} \alpha_i f_i(\mathbf{x}),\ \mathbf{x} \in \mathcal{X},\ \boldsymbol{\alpha} \in \mathcal{B} \right\}. \tag{5.6}$$

Due to the fact that the problem of approximating a function from sparse data is ill-posed [70] we have already assumed two different types of restrictions on the shape of the predictor function: i) $f_{\boldsymbol{\alpha}}(x)$ is a linear combination of a *finite* set of basis functions and ii), this linear combination is constrained components by components using *intervals*. In addition, inspired by regularization theory [18; 66] we add a regularization term to the cost function to be optimized:

$$\min_{f_{\boldsymbol{\alpha}}(x) \in Box(S,\mathcal{B})} E_{\mathbf{x},y} H(y_i f_{\boldsymbol{\alpha}}(x_i)) + \lambda \|\boldsymbol{\alpha}\|_A^2, \tag{5.7}$$

where $\lambda > 0$ and $A \in \mathbb{R}^{k \times k}$ is an arbitrary symmetric positive-definite matrix.

The above convex optimization task implies numerous classification methods such as Support Vector Machines [68; 69], its Smooth [39] or Least Squares [64] counterparts, and Logistic Regression [21], which will be discussed later. We should note here that in certain cases the hinge loss will be substituted by a smooth, twice continuously differentiable approximant. If we have a large number of data points in the training set and/or numerous variables in Eq. (5.7), the optimization procedure requires using advanced optimization tricks. We give a brief list of such solutions through the example of SVM's without aiming at completeness. SVM was originally formulated as a quadratic programming (QP) problem. Unfortunately, even for a thousand point dataset, one is faced with a fully dense quadratic problem with a constraint matrix having a million entries. Unfortunately too, the dual QP problem also contains the kernel matrix, which has a quadratic size data structure. To reduce the time and space complexities different techniques were developed by the machine learning community: i) a low rank approximation of the kernel matrix can be used [73]; ii) the optimization procedure may be performed by various sophisticated optimization methods like SMO [50], Chunking [11], decompositions [19] and coordinate ascent methods [22]; iii) the QP problem might be transformed into another quasi-similar forms (e.g. LS-SVM, RSVM, SSVM) for making the optimization task easier; and iv) the application of various approximate algorithms

are suitable for speeding up the computations [3; 67].

Here we restrict our investigations to the box-constraint non-linear Gauss-Seidel method [8], because it is quite suitable for the optimization problem defined in Eq. (5.7.) First, this method converges to a single optimum. Moreover we make some extensions so that during the iterations the number of non-zero parameters stays below a pre-set parameter. To this end we both apply straightforward heuristics and feature selection algorithms like Sequential Forward Selection, Sequential Forward Floating Selection and the Plus-l Take Away-r method [53]. We will demonstrate that the methods proposed in the Chapter are effective and fast using standard datasets of different sizes [9]. The structure of the remaining part of the Chapter will follow the same pattern as these described hereunder.

5.3 Methods implied

In this section for demonstration purposes we will show that some ubiquitous kernel-based classification methods can be reformulated so that they have the same form as the problem defined in Eq. (7).

5.3.1 Support Vector Machines

Let $\mathcal{X}$ be a compact set in the m-dimensional Euclidean space. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a Mercer kernel, if and only if it is a) continuous, b) symmetric, and c) positive definite. We mean by '$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is positive definite' that for every finite set $\{\mathbf{x}_1, \ldots, \mathbf{x}_r\} \subset \mathcal{X}$ the $r \times r$ kernel matrix $K = [k(\mathbf{x}_i, \mathbf{x}_j))]_{i,j=1}^{r}$ is positive semi-definite. Now let the set of the basis functions be defined by

$$S = \{y_1 k(\mathbf{x}, \mathbf{x_1}), \ldots, y_n k(\mathbf{x}, \mathbf{x}_n)\}, \tag{5.8}$$

where k is a Mercer kernel, $\mathbf{x} \in \mathcal{X}$ and $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ form the training database.

The standard linear SVM problem [16; 69] without the bias term is given by the following for some $C > 0$,

$$\begin{array}{rll} \min_{\mathbf{w}} & C\mathbf{e}^T\boldsymbol{\xi} + \frac{1}{2}\mathbf{w}^T\mathbf{w} & \\ \text{s.t.} & YX\mathbf{w} + \boldsymbol{\xi} \geq \mathbf{e}\ , & \\ & \boldsymbol{\xi} \geq \mathbf{0} & \end{array} \tag{5.9}$$

where Y is a diagonal matrix with $y_1, \ldots, y_n$ lying along the diagonal, $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^T$ and $\mathbf{e}$ is a column vector of ones in the appropriate dimension. We usually solve this

by using the primal Lagrangian defined as

$$\begin{array}{ll} \max_{\mathbf{w},\boldsymbol{\alpha}} & C\mathbf{e}^T\boldsymbol{\xi} + \frac{1}{2}\mathbf{w}^T\mathbf{w} - \boldsymbol{\alpha}^T(YX\mathbf{w} + \xi - \mathbf{e}) \\ \text{s.t.} & \boldsymbol{\alpha}, \boldsymbol{\xi} \geq \mathbf{0} \end{array}, \tag{5.10}$$

from which taking the derivatives we have the following conditions for the single optimum

$$\mathbf{w} = X^T Y \boldsymbol{\alpha} \qquad \mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{e}. \tag{5.11}$$

With these conditions from Eq. (5.10) the following Langrangian dual problem can be turned out:

$$\begin{array}{lrcl} \min_{\boldsymbol{\alpha}} & C\mathbf{e}^T\boldsymbol{\xi} + \frac{1}{2}\boldsymbol{\alpha}^T YXX^TY\boldsymbol{\alpha} & & \\ \text{s.t.} & YXX^TY\boldsymbol{\alpha} + \boldsymbol{\xi} & \geq & \mathbf{e} \\ & \boldsymbol{\alpha} & \geq & \mathbf{0} \\ & -\boldsymbol{\alpha} & \geq & C\mathbf{e} \\ & \boldsymbol{\xi} & \geq & \mathbf{0} \end{array}, \tag{5.12}$$

where XX^T may be substituted by a $K = [k(\mathbf{x}_i, \mathbf{x}_j))]_{i,j=1}^n$ kernel matrix. We should note here that strictly positive definite kernels do not require bias term [52]. At a solution of problem (5.12) $\boldsymbol{\xi}$ is given by $(\mathbf{e} - YKY\boldsymbol{\alpha})_+$, where

$$(\mathbf{z}_+)_i = \max\{0, z_i\} \qquad i = 1, \ldots, n. \tag{5.13}$$

Exploiting this connection in Eq. (5.12) we get

$$\begin{array}{ll} \min_{\boldsymbol{\alpha}} & \sum_{i=1}^n \left(1 - y_i \sum_{j=1}^n \alpha_j y_j k(\mathbf{x_i}, \mathbf{x_j})\right)_+ + \frac{1}{2C}\boldsymbol{\alpha}^T YKY\boldsymbol{\alpha} \\ \text{s.t.} & \mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{e} \end{array}, \tag{5.14}$$

which is a already problem in the form of Eq. (5.7) with the following definitions

$$\begin{array}{rcl} H(x) & = & (1-x)_+, \\ \lambda & = & \frac{1}{2C}, \\ A & = & YKY, \\ \mathcal{B} & = & [0, C]^n. \end{array} \tag{5.15}$$

Since the hinge loss H(x) (Fig. 4.1.D) is not two times continuously differentiable we may replace it using a very accurate smooth approximation of $(x)_+$ defined by

$$p(x) = x + \frac{1}{\beta}\log(1 + e^{-\beta x}),\ \beta \gg 0. \tag{5.16}$$

Actually, it can be shown that as the smoothing parameter β tends to infinity the unique solution of the smoothed problem approaches the unique solution of the original one.

5.3.2 Smooth Support Vector Machines

Y. Lee and O. Mangasarian defined the Smooth Support Vector Machine (SSVM) both in a linear and a kernel-based non-linear form [39]. We show that the latter can be reformulated as an optimization task in form of Eq. (5.7). The SSVM is given by the following unconstraint minimization problem for some $C > 0$,

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^n,\, b\in\mathbb{R}} \quad \frac{C}{2}\|p\left(\mathbf{e} - Y\left(KY\boldsymbol{\alpha} - b\mathbf{e}\right)\right)\|_2^2 + \frac{1}{2}\left(\boldsymbol{\alpha}^T\boldsymbol{\alpha} + b^2\right), \tag{5.17}$$

where b denotes the bias term, $K = [k(\mathbf{x}_i, \mathbf{x}_j))]_{i,j=1}^r$ is the kernel matrix and the domain of function p of Eq. (5.16) is extended to vectors component-wise. Now after some algebraic re-arrangement and denoting the identity matrix by I and b with α_{n+1} we get the following minimization problem:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^{n+1}} \quad \sum_{i=1}^{n} \frac{C}{2} p\left(1 - y_i\left(\sum_{j=1}^{n} y_j\alpha_j k\left(\mathbf{x}_i, \mathbf{x}_j\right) - \alpha_{n+1}\right)\right)^2 + \frac{1}{2}\boldsymbol{\alpha}^T I\boldsymbol{\alpha}. \tag{5.18}$$

It is obvious that Eq. (5.18) is of form Eq. (5.7), but in contrast to the SVM case the set of basis function S has an additional element:

$$S = \{y_1 k(\mathbf{x}, \mathbf{x_1}), \ldots, y_n k(\mathbf{x}, \mathbf{x}_n), -1\}, \tag{5.19}$$

and the appropriate choice for the definitions are

$$\begin{aligned} H(x) &= \tfrac{C}{2} p\left(1 - x\right)^2, \\ \lambda &= \tfrac{1}{2}, \\ A &= I, \\ \mathcal{B} &= \left(-\infty, \infty\right)^{n+1}. \end{aligned} \tag{5.20}$$

5.3.3 Least Squares Support Vector Machines

J. A. K. Suykens and J. Vandewalle defined a variant of SVM methods, the Least Squares Support Vector Machines (LS-SVM), which simplifies the original SVM optimization problem to a set of linear equations [64]. Employing our notations the form of the linear system of equations of LS-SVM can be specified by

$$\begin{bmatrix} YKY + \frac{1}{C}I & \mathbf{y} \\ \mathbf{y}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{e} \\ 0 \end{bmatrix}, \tag{5.21}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^n$, $b \in \mathbb{R}$ and $\mathbf{y} = (y_1, \ldots, y_n)^T$. Now let us denote the right hand side matrix by Z and b with α_{n+1} again. Using this notations Eq. (5.21) can be solved by

an equivalent least squares optimization problem:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^{n+1}} \|Z\boldsymbol{\alpha}-\mathbf{e}\|_2^2 = \mathbf{e}^T\mathbf{e} + 2\mathbf{e}^T Z\boldsymbol{\alpha} + \boldsymbol{\alpha}^T Z^T Z\boldsymbol{\alpha}. \tag{5.22}$$

Since $\mathbf{e}^T\mathbf{e}$ is a constant and $\boldsymbol{\alpha}^T Z^T Z\boldsymbol{\alpha}$ is an appropriate norm-square, we only need to bring $2\mathbf{e}^T Z\boldsymbol{\alpha}$ into the form of $E_{\mathbf{x},y}H(y_i f_{\boldsymbol{\alpha}}(x_i))$. To this end first we define H(x) by slightly modifying the hinge loss:

$$H(x) = (\gamma - x)_+, \tag{5.23}$$

where γ is a sufficiently large constant, which we define later. Naturally $p(x)+\gamma-1$ is an appropriate smooth approximation for $H(x)$. Based on the above preparations we may give the LS-SVM method in the form of Eq. (5.7),

$$\begin{array}{ll} \min\limits_{\boldsymbol{\alpha}\in\mathbb{R}^{n+1}} & H\left(y_i\left(\sum_{j=1}^{n+1}\alpha_j f_j\left(\mathbf{x}_i\right)\right)\right) + \boldsymbol{\alpha}^T Z^T Z\boldsymbol{\alpha} \\ \text{s.t.} & -\delta\mathbf{e} \le \boldsymbol{\alpha} \le \delta\mathbf{e}. \end{array}, \tag{5.24}$$

where δ is a parameter chosen so that the optimal solution $Z^{-1}\mathbf{e}$ surely falls into $[-\delta\mathbf{e}, \delta\mathbf{e}]$ and the base functions are arbitrary continuous functions satisfying the following conditions for the training data

$$\begin{array}{rcl} f_j(\mathbf{x}_i) & = & \begin{cases} y_j k(\mathbf{x}_i,\mathbf{x}_j) & \text{if } i \ne j \\ y_j k(\mathbf{x}_i,\mathbf{x}_j) + \left(1+\frac{1}{Cy_i}\right) & \text{otherwise} \end{cases} \qquad j=1,\ldots,n. \\ f_{n+1}(\mathbf{x}_i) & = & 1 \end{array} \tag{5.25}$$

The definitions in Eq. (5.7) in the case of LS-SVM will be

$$\begin{array}{rcl} \lambda & = & 1, \\ A & = & Z^T Z, \\ \mathcal{B} & = & [-\delta\mathbf{e}, \delta\mathbf{e}]. \end{array} \tag{5.26}$$

We have yet another duty, namely we need to define γ, which is an arbitrary positive constant larger than

$$\max_{i,j,\boldsymbol{\alpha}} \left(y_i \left(\sum_{j=1}^{n+1} \alpha_j f_j\left(\mathbf{x}_i\right) \right) \right) \quad i,j \in \{1,\ldots,n\}, \quad \boldsymbol{\alpha} \in [-\delta\mathbf{e}, \delta\mathbf{e}]. \tag{5.27}$$

The above selection for γ ensures that the modified hinge loss H(x) will be evaluated only for values less than γ.

5.3.4 Kernel Logistic Regression

Now we will show that Logistic Regression and its non-linear kernel-based version, Kernel Logistic Regression – see [21; 56] – fall into the group covered by Eq. (5.7). Like that defined by [21] the above methods lead to the following unconstrained optimization task:

$$\min_{\boldsymbol{\alpha},\, b} \quad C\sum_{i=1}^{n} H\left(y_i b - \sum_{j=1}^{n} \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)\right) + \boldsymbol{\alpha}^T Y K Y \boldsymbol{\alpha}, \tag{5.28}$$

where $H(x) = \log(1 + e^x)$. Using the notation $\alpha_{n+1} = b$ and dividing the objective function by C we have that

$$\min_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{n} g\left(-y_i\left(\sum_{j=1}^{n+1} \alpha_j f_j(\mathbf{x}_i)\right)\right) + \frac{1}{C}\boldsymbol{\alpha}^T Y K Y \boldsymbol{\alpha}, \tag{5.29}$$

which is undoubtedly in the form of Eq. (5.7) taking into account that

$$\begin{aligned} \lambda &= 1, \\ A &= YKY, \\ \mathcal{B} &= (-\infty, \infty)^{n+1}. \end{aligned} \tag{5.30}$$

5.4 The Non-linear Gauss-Seidel Method with box constraint

In the previous section we demonstrated that problem (5.7) implies a set of well-known classification methods. As we mentioned earlier, numerous optimization techniques can be used for solving Eq. (5.7) or its variants. Numerous authors in different contexts suggested using different coordinate ascent based iterative learning like the Non-linear Gauss-Seidel or the Gauss-Southwell method for similar optimization tasks. We also propose to apply the Non-linear Gauss-Seidel method here with an additional box constraint [8], which is suitable for solving Eq. (5.7). First we discuss the iteration itself, then in the next section we include some suggested heuristic improvements of the method.

Definition 5.1 (projection mapping)

$$[\ ]^p : \mathbb{R}^k \to \mathcal{A} \qquad [\mathbf{x}]^p = \mathbf{z} \Leftrightarrow \|\mathbf{x} - \mathbf{z}\|^2 = \min_{\mathbf{y}\in\mathcal{A}} \|\mathbf{x} - \mathbf{y}\|^2 \tag{5.31}$$

Definition 5.2 (box constrained Gauss-Seidel iteration)

$$\mathbf{x}_i^{t+1} = \left[\mathbf{x}_i^t - \gamma \nabla_i f(\mathbf{z}_i^t)\right]_i^p \tag{5.32}$$

where

$$\gamma > 0, \qquad \mathbf{z}_i^t = (x_1^{t+1}, \ldots, x_{i-1}^{t+1}, x_i^t, \ldots, x_k^t), \qquad \mathbf{x}^{t+1} = \mathbf{z}_{k+1}^t. \tag{5.33}$$

During the iteration process each component of the actual solution $\mathbf{x}^t$ is successively upgraded by the gradient rule. If the solution falls outside the domain it will be replaced by the nearest point of the set with the aid of the projection mapping. We know that the box constrained GS iteration method is convergent for every function $f : \mathcal{A} \to \mathbb{R}$ over a non-empty, convex and closed set $\mathcal{A}$, where f is twice continuously differentiable and lower bounded. Moreover, the gradient should be a Lipschitz function and there must exist a $\delta > 0$ such that

$$0 < \delta \leq \nabla_{ii}^2 f(\mathbf{x}). \tag{5.34}$$

The limit point of the iteration is the extremum of the function over $\mathcal{A}$ [8]. We know also from Bertsekas & Tsitsiklis's book [8] that the Lipschitz condition concerning the gradient can be ignored if every level set of the function is bounded, which is the case for problem (5.7).

5.5 Sparse Convex Machines

Sparse solutions (i.e. sparse input-output models) for classification problems are beneficial for two reasons. First, we may avoid the problem of overfitting and second, both the optimization procedure and the evaluation of the classifier is faster. Therefore forcing as many $\boldsymbol{\alpha}_i$ parameters to be zero as possible in the predictor function

$$f_{\boldsymbol{\alpha}} = \sum_{i=1}^{k} \alpha_i f_i(\mathbf{x}) \tag{5.35}$$

is a reasonable strategy. For the sake of controlling the sparsity the number of basis functions with zero-coefficients will be restricted by making the following assumption

$$\sum_{i=1}^{k} |sign(\alpha_i)| \leq q, \tag{5.36}$$

where q is a preset parameter. Unluckily, such a condition makes the optimization problem of Eq. (5.7) combinatorial, so the suggested non-linear Gauss-Seidel technique in its original form cannot be applied. Our aim is to select from the available basis functions a subset of order q at most with minimal function value. This task is NP hard [15] so the only effective way here is to employ heuristics.

Algorithm 2 MGRAMM(q)

input: $q,\ f_1, \ldots, f_k$ // no. of subjects

method:

$Y = \{1, \ldots, k\};\ I = \emptyset;$
for $i = 1 \ldots q$

$$t = argmax_{j \in Y - I} \quad \| f_j - \sum_{l=1}^{i-1} \frac{\langle f_j, f_l^* \rangle}{\langle f_l^*, f_l^* \rangle} f_l^* \ \|_2$$

$$I = I \cup \{t\};$$

$$f_i^* = f_t - \sum_{l=1}^{i-1} \frac{\langle f_t, f_l^* \rangle}{\langle f_l^*, f_l^* \rangle} f_l^*;$$

return I;

5.5.1 Basic Heuristics

In this section we deal with algorithms that do not use the Convex Machine (CM)'s objective function defined in Eq. (5.7) during the optimal basis function subset selection of order q.

RANDOM The simplest strategy is the random selection approach when we randomly select q basis functions from among the k basis functions. This approach does not have an objective function that can be minimized so we will choose instead the subset with the best performance after several executions.

MGRAMM Convex Machines approximates the optimal separator surface using a linear combination of the basis functions. Hence the approximation can be performed on an orthogonal basis of the function space, as in the case of the result of the Gramm-Schmidt orthogonalization algorithm. Despite this, the dimension of the basis is the rank of the function set which can exceed the desired number q. Moreover, the algorithm generates an orthogonal function system with linear combinations of basis functions instead of selecting the individual functions.

To solve the above we will define a greedy iterative selection strategy (cf. Algorithm 2) based on a modified version of the Gramm-Schmidt orthogonalization algorithm. Among the available basis functions we choose the one with a maximal residual norm after the Gramm-Schmidt process at each step. The result of this greedy method is not the orthogonal function system itself but the basis functions used in the linear combinations.

Assume that the basis functions are elements of L_2 so the dot product is the integral of the product function. When analytical computations of the integrals

Algorithm 3 CORR(q)

input: $q,\ f_1, \ldots, f_k$ // no. of subjects

method:

$Y = \{1, \ldots, k\}$; $I = \emptyset$;

for i = 1...q

$t = argmin_{j \in Y - I} \sum_{l=1}^{i-1} \frac{\langle f_j, f_l^* \rangle^2}{\langle f_j, f_j \rangle}$

$I = I \cup \{t\}$;

$f_i^* = \frac{f_t}{\langle f_t, f_t \rangle^{0.5}}$;

return I;

are not possible we utilize the following approximation in the algorithm using the sample points

$$\langle f, g \rangle = \sum_{i=1}^{n} f(\mathbf{x}_i) g(\mathbf{x}_i) \qquad f, g : \mathcal{X} \to \mathbb{R}. \tag{5.37}$$

CORR The MGRAMM method tries to choose an orthogonal basis of the functions with the help of the Gramm-Schmidt process. The choice might be good when the dot product of functions is available. Employing the approximation in Eq. (5.37) the result of the algorithm will be also just an approximation of the desired basis.

Such an estimation can be carried out in different ways. The orthogonality of the elements in the basis can be also employed, since the mutual correlation coefficients must be zero. Our aim is to select functions such that the squared sum of the element in the correlation matrix should be minimal. Similar to MGRAMM this method (cf. Algorithm 3) will be a greedy iterative process and also will exploit the fact that the mutual correlation coefficient for normalized functions takes the form of Eq. (5.37).

5.5.2 Complex Heuristics

Measure-based subset selection is an active area of other fields in artificial intelligence like Feature Selection [53], say. In this context one should select r features from the available m to maximize the classification performance of a machine learning algorithm. The elaborated techniques can be employed for our subset selection problem if the required measure is replaced by the objective function value of the CM task. In the following let $CM(I)$ denote the optimal value of the objective function in CM when

Algorithm 4 SFS(q)

input: $q,\ f_1, \ldots, f_k$ // no. of subjects

method:

$Y = \{1, \ldots, k\}$; $I = \emptyset$;
for i = 1...q
  $t = argmin_{j \in Y-I}\ CM\,(I \cup \{j\})$
  $I = I \cup \{t\}$;
return I;

the basis function set is restricted in the following way

$$S = \{f_{i_1}(\mathbf{x}), \ldots, f_{i_l}(\mathbf{x})\} \qquad i_1, \ldots, i_l \in I. \tag{5.38}$$

SFS The Sequential Forward Selection method is a greedy approach for the measure-based subset selection problem. Starting with the empty index set it extends the indices with the locally optimal element without backtracking (cf. Algorithm 4).

PTA The SFS method is a sequential algorithm, hence previous steps cannot be modified when detecting their latter impact on the result. A solution to the problem is the Plus *l* Take Away *r* approach. It periodically extends the actual index set by *l* elements and afterwards removes *r* so that the measure is locally optimal after each step. By doing this the effects of previous selections can be eliminated during the execution of the subroutine (cf. Algorithm 5).

SFFS When running a PTA routine r removing steps always follow l extending ones. Hence it is possible to execute a removing step when the evolving set has a worse measure value than the previous one of the same order. Conversely, an extending step can be performed when we get a better solution at the that particular level by removing a function. These problems are absent in the Sequential Forward Floating Selection algorithm. It sequentially removes elements after one extending step while the measure we obtain is better than the previous ones of the same order (cf. Algorithm 6).

5.6 Results

We now demonstrate the effectiveness of the Convex Machine approach by comparing its results with other methods.

Algorithm 5 PTA(q, l, r)

input: $q,\ f_1, \ldots, f_k$ // no. of subjects

method:

$Y = \{1, \ldots, k\}$;
if (l > r) then i = 0; $I = \emptyset$; goto Step1;
else i = k; $I = Y$; goto Step2;
Step1:
repeat l times
$t = argmin_{j \in Y - I}\ CM(I \cup \{j\})$;
$I = I \cup \{t\}$; i = i + 1;
if (i == q) goto Step3;
Step2:
repeat r times
$t = argmin_{j \in I}\ CM(I - \{j\})$
$I = I - \{t\}$; i = i - 1;
if (i == q) goto Step3;
goto Step1;
Step3:
return I;

Algorithm 6 SFFS(q)

input: $q,\ f_1, \ldots, f_k$ // no. of subjects

method:

$Y = \{1, \ldots, k\}$;
$Y_0 = \emptyset$; i = 0;
Step1:
$t = argmin_{j \in Y - Y_i}\ CM(Y_i \cup \{j\})$;
$Y_{i+1} = Y_i \cup \{t\}$; i = i + 1;
if (i == q) goto Step3;
Step2:
$t = argmin_{j \in Y_i}\ CM(Y_i - \{j\})$
if $CM(Y_i - \{t\}) < CM(Y_{i-1})$ then
$Y_{i-1} = Y_i - \{t\}$; i = i - 1;
goto Step2;
else
goto Step1;
Step3:
return Y_q;

	ANN	SVM	CM
balance	89.03	93.55	99.79
	86.35	90.63	**95.41**
bupa	72.01	81.73	80.69
	68.07	**74.39**	71.92
glass	84.24	99.79	100.0
	69.87	84.70	**86.23**
iono	93.35	99.40	99.94
	86.17	91.09	**92.41**
monks	90.64	97.50	99.05
	87.28	95.82	**96.51**
pima	78.68	82.49	80.55
	76.09	75.58	74.82
wdbc	98.71	99.47	100.0
	97.61	**97.62**	96.93
wpbc	85.71	98.47	99.04
	76.41	77.36	**79.63**

Table 5.1: Ten-fold cross-validation training and testing results on some UCI datasets using three different methods. ANN is a feed-forward neural network with one hidden layer where the number of hidden units was set at three times the class number. SVM used the cosine polynomial kernel defined in Eq. (5.39) with $q = 3$ and $\sigma = 1$ for nonlinearity. With the help of Eq. (5.14) the CM method applied the same basis functions.

5.6.1 Classification Tests with CM

In order to evaluate how well each algorithm classifies an unknown dataset, we performed a tenfold cross-validation on publicly available datasets from the UCI repository [9]. The performance of the CM method was compared with Artificial Neural Networks (ANN) and Support Vector Machines (SVM).

We applied a feed-forward neural network (MLP) with one hidden layer, where the number of hidden neurons was set at three times the class number. The backpropagation learning rule was applied for training. MLP was executed five times on each dataset and then we chose the parameter values which gave the best performance on training sample.

For an impartial comparison we employed our 1-norm SVM implementation where the bias term was absent [52]. Multiclass cases were handled by the one-against-all approach. Additionally, the cosine polynomial kernel we applied made the SVM method non-linear

$$\kappa\left(\mathbf{x},\mathbf{y}\right)=\left(\frac{\mathbf{x}^{T}\mathbf{y}}{\left\|\mathbf{x}\right\|\left\|\mathbf{y}\right\|}+\sigma\right)^{q},\qquad q\in\mathbb{N},\ \sigma\in\mathbb{R}_{+} \tag{5.39}$$

with parameters $q = 3$ and $\sigma = 1$.

The basis functions for the CM problem were defined by the above kernel function

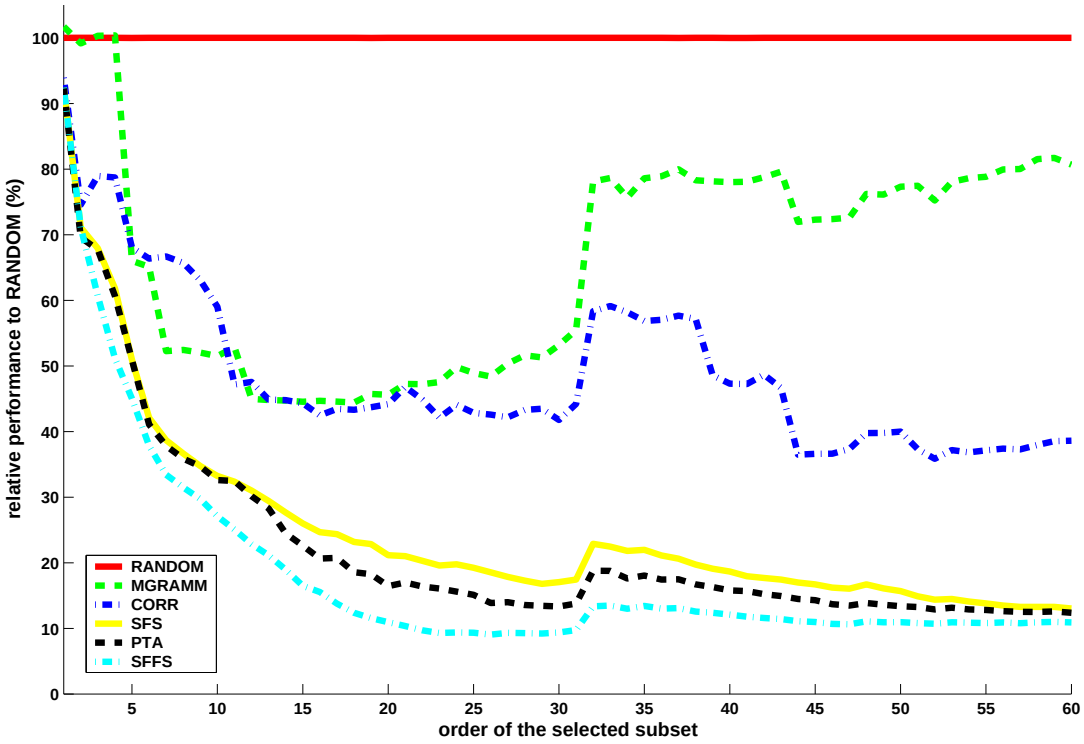

Figure 5.1: Performances of the proposed heuristics controlling CM sparsity on the Iono database expressed in percentages of the RANDOM method result. The CM measures were used as performance indicators regardless of how the methods works.

using the sample points of a training set, as shown in Eq. (5.14). Thus

$$f_j(\mathbf{z}) = y_j \kappa(\mathbf{z}, \mathbf{x}_j). \qquad j = 1, \ldots, n \tag{5.40}$$

The coefficients of the basis functions were not restricted in our tests, i.e. we used the domain $\mathcal{A} = (-\infty, \infty)^n$. In the regularization term of Eq. (5.7) we set the identity matrix equal to A with $\lambda = 1$.

It turned out that, on most of the datasets tested, the tenfold testing correctness of the CM problem was the highest for these methods. We summarize all these results in Table 5.1. It confirms that the CM classification method is indeed just as effective as the ubiquitous machine learning algorithms. Moreover, their performances were surpassed in many cases. It can be readily seen that the problem of overfitting the data was present more often in the methods with global optima. It might be explained with the locally optimal solution of the ANN method, which can be regarded as a kind of regularization. Similar results are expected when using sparse heuristics to solve a CM problem.

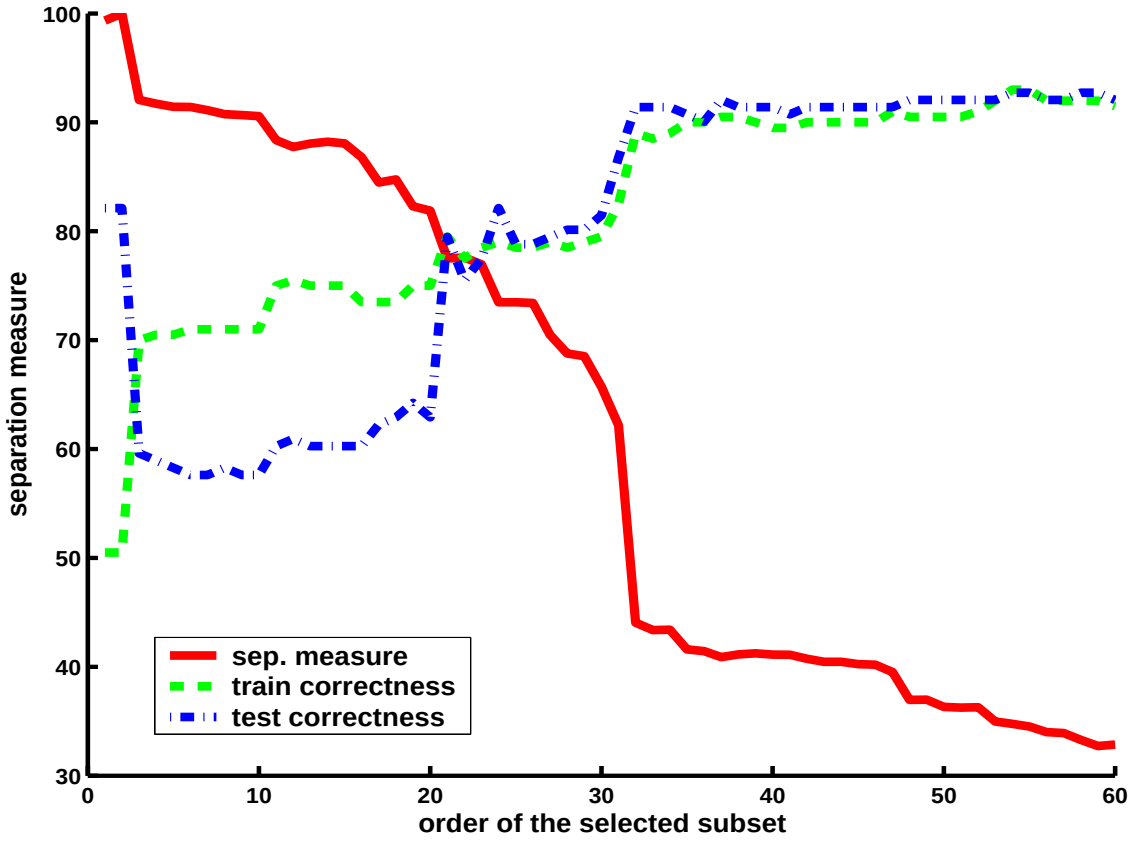

Figure 5.2: The consistency of the measure in the CM method and its abstraction ability with the aid of the MGRAMM method on the Iono database. The decreasing CM measure means a better testing correctness.

5.6.2 Examining the Effect of the Sparse Representation

We also examined the performance of the proposed heuristics controlling CM sparsity. We compared the methods on the Iono database by examining the value of the CM objective function, regardless of how the methods worked. The results of the six heuristics are shown in Fig. 5.1. We used the performance of the RANDOM method as a reference so the results of other algorithms are expressed in percentages. The RANDOM method chose its best from 5 executions and the PTA was performed with settings $l = 3$ and $r = 1$.

As the reader will notice, the Feature Selection (FS)-based techniques performed better than the general purpose selection algorithms. The SFFS method achieved the best performance, surpassing both the SFS and PTA techniques. In the other group the MGRAMM and CORR approaches achieved similar results, and both of them outperformed the RANDOM method here. These latter algorithms tend to the FS-based methods and have faster computational speeds.

During the subset selection we optimize some measures while the abstraction ability is the most important in the machine learning sense. The consistency of the measure

RANDOM MGRAMM CORR	10%	20%	30%	100%
balance	95.10 95.25 **95.41**	95.25 95.40 95.10	95.40 95.25 95.25	95.41
bupa	70.49 69.14 69.12	71.35 70.53 69.12	69.14 **71.61** 69.42	71.92
glass	84.75 85.16 85.18	**89.66** 86.62 85.16	87.00 85.91 86.66	86.23
iono	89.16 91.23 91.58	**93.19** 92.04 91.32	92.68 90.54 91.85	92.41
monks	93.18 92.94 92.65	93.70 91.66 94.40	94.76 90.89 **95.11**	96.51
pima	**78.51** 77.89 77.62	77.24 76.43 76.22	75.97 77.87 76.60	74.82
wdbc	**97.44** 97.25 97.10	97.27 96.93 96.93	97.44 96.09 96.93	96.93
wpbc	78.27 76.37 74.05	78.29 75.14 75.93	77.29 73.20 **79.70**	79.63

Table 5.2: Ten-fold cross-validation testing results of the Convex Machines method using the heuristics RANDOM, MGRAMM and CORR. The sparsity was controlled by maximizing the number of available basis functions to 10%, 20% and 30% of the complete sets, respectively.

in the CM method and its abstraction ability can be seen in Fig. 5.2 with the aid of the MGRAMM method on the previous database. As can readily be seen, the decreasing CM measure value means a better abstraction ability, i.e. testing correctness. Thus the measure of the CM approach might indeed be employed as an objective function of machine learning algorithms.

5.6.3 Classification experiments using Heuristics

The performance of heuristics with faster computational speed were examined with the help of ten-fold cross-validation. We summarize our results in Tables 5.2 and 5.3. The sparsity of solutions were maximized using 10%, 20% and 30% of the available functions. The RANDOM and PTA methods both had the same parameters as those above.

SFS PTA SFFS	10%	20%	30%	100%
	94.66	94.77	94.91	
balance	94.75	95.19	94.92	**95.41**
	95.41	94.97	95.08	
	69.46	69.22	68.96	
bupa	69.20	70.66	71.41	**71.92**
	70.37	71.20	69.88	
	84.36	**88.62**	86.63	
glass	85.10	86.56	85.74	86.23
	85.18	85.47	86.91	
	89.24	91.81	90.71	
iono	91.51	92.25	92.71	92.41
	92.13	**93.24**	92.48	
	93.07	93.55	94.64	
monks	92.80	91.85	90.79	**96.51**
	92.65	94.90	95.49	
	77.86	76.43	75.97	
pima	**78.42**	77.14	76.55	74.82
	77.74	76.49	78.03	
	97.44	97.22	97.38	
wdbc	97.35	96.96	96.16	96.93
	97.18	97.29	97.43	
	76.26	75.92	74.75	
wpbc	78.06	78.19	77.03	79.63
	75.82	76.39	**79.98**	

Table 5.3: Ten-fold cross-validation testing results of the Convex Machines method using the heuristics SFS, PTA and SFFS. The sparsity was controlled by maximizing the number of available basis functions to 10%, 20% and 30% of the complete sets, respectively.

As observed, all of the algorithms selected subsets with adequate testing correctness. This kind of capacity reduction in the CM learning method brings about a sort of regularization which is reflected in the results: results with a reduced basis outperform the original ones in many cases. The various algorithms here have their best performance on different tasks. In general, different requirements in the learning phase will lead the user to select one of the available heuristics.

5.7 Conclusions

This chapter gave an overview of the CM methodology, which is a reformulation of certain machine learning algorithms including several well-known nonlinear classification

methods. The CM problem can be solved by the convergent nonlinear Gauss-Seidel iteration process, which is sufficiently fast for this task. The numerical results on its abstraction ability show that the CM method can be considered as a rival classification method to both ANN and SVM. Moreover, the sparsity of the CM problem can be effectively controlled by the proposed heuristics. The FS-based ones performed better than the general purpose selection techniques but they required longer computational times. The faster algorithms in the other group just approximate the performance of FS techniques but they could be employed in larger problems. Future work includes a new heuristic based on a CM objective function which can be utilized in very large classification problems. We also plan to use chunking algorithms like those described in [11] for problems which do not fit in the memory.

Chapter 6

Conclusions

In this brief chapter, I summarize the consequences of the presented thesis. The primary inductor of scientific improvements is the appearance of technical innovations. While in the sixties, artificial intelligence-based methods were mainly limited to linear and quadratic models, current computational capacities allow a much more intensive improvement in model building. A further aspect, which contributed to the advances of the field is that the number and size of learning databases increased considerably.

The thesis is built around the kernel idea, which is suitable for the transformation of linear models into non-linear ones, while only minimally increasing the complexity of the model. The kernel idea is able to transform algorithms that only use the dot product of a set of sample vectors as their input. In this case, we can acquire alternative models with the non-linear re-definition of the dot product. Thus, the main objective is to re-define the basic tasks of machine learning, such as feature extraction, classification, regression in the form of dot products. The results included in my dissertation contribute to the described idea with a number of novel algorithms.

In the first part of the thesis, I defined feature extraction techniques that effectively increase the precision of classification and regression methods. We succeeded in justifying the grounds for these techniques by making them work on standard data bases of machine learning and on the task of face recognition. In summary, we can state that the four elaborated methods pave the way to the effective reduction of high dimensional feature spaces.

In the second part, we focused on the topic of classification. We first introduced a family of hyperplane-based classification methods. These outline a group of methods that gives a deeper insight into the nature of how variably the hyperplane can be used for classification. The leading motif of the newly defined method group is the underlying geometrical approach. In the same part, we define a general classification scheme leading to the optimisation of convex functions, which comprises a number of techniques developed over the past few years. The unified approach that determined

our way of thinking in this case supposes a traditional numeric mathematical thinking. The methods introduced in part two performed well on both real and artificial data. With this, we managed to emphasize usability beyond the constructional results.

Appendix A

Summary

A.1 Summary in English

Introduction

Model building is probably the most significant component of scientific thinking baring abstract similarities in all branches of science. We build models and examine how they function, we apply modifications to them as long as we reach our desired goal. Artificial intelligence is a special branch of science, which provides an endless horizon to the lovers of computational model building.

Machine learning forms a branch of artificial intelligence [59]. It comprises a set of algorithms that enable computers to learn. The concept of learning usually builds on two different approaches: inductive and deductive learning. My thesis follows the inductive learning approach, which retrieves rules or descriptive patterns from massive datasets. Presently, the main focus of machine learning is the complex task of automatic information extraction. Further important application possibilities lie in natural language processing, syntactic pattern recognition, the improvement of search engines, medical diagnosis, bio-informatics, speech recognition, object recognition, and enhancing computer games with humanlike intelligence - only to mention a few fields. Certain machine learning methods attempt to eliminate human resource from the process of data analysis, while others try to make human-machine interaction more human.

Considering the current level of technological advances, machine learning has become the most researched and most intensively developing field of artificial intelligence. The present thesis focuses on the examination and elaboration of the most novel approach in machine learning, namely that of kernel methods.

Kernel methods (KM) are a family of pattern recognition algorithms [61], whose most significant member is the Support Vector Machine (SVM) [69]. The general task of pattern recognition is to identify and examine representative correlations (e.g., clusters, classification decisions, etc.) on general data (e.g., vectors, documents, se-

quences, pictures, etc.). The KM approach was named after kernel functions, which work in a derived feature space where the real coordinates of patterns never have to be calculated. These methods only rely on the dot product of paired sample points, which are calculated implicitly by applying the kernel functions. Apart from SVM, there are a number of other algorithms belonging to the family of KM: e.g., various regression methods, Fisher's linear discriminant analysis (LDA) [25], principal components analysis (PCA) [29], canonical correlation analysis (CCA) [2], ridge regression [49], spectral clustering [48], and many others. Generally speaking, the majority of kernel methods lead to effectively soluble problems, convex optimisation or eigenproblems.

Results

The theses of the present dissertation can be differentiated in two ways and can be separated into two different groups. In one reading, the results acquired by the author fall into the topic of kernel-based feature extraction and classification methods within the field of machine learning. In an other reading, we can learn about algorithmic constructions and practical applications. Hereunder, following the structure of the dissertation, the results are introduced according to the first approach. It is important to note that the list of results below enumerate only those parts of the novelties to which the author has contributed in major part.

The author's kernel-based feature extraction methods form the first group of results. These results are described in detail in Chapter 2 and 3 of the dissertation.

I/1. The author has defined the direct version of the MMDA algorithm [31; 37]. This method employs a feature extraction technique that increases the efficiency of classification methods. The author managed to prove the feasibility of the defined method by applying it on several examples of the UCI machine learning database [9].

I/2. The MMDA algorithm is apt by nature to reduce high dimensional feature spaces in order to increase classification efficiency. The author has developed a modified version of this algorithm for face recognition. With the help of the FERET gold standard face recognition database [17], he managed to prove the usability of the introduced method. He also managed to surpass the state-of-the-art results in the field [37].

I/3. Beyond classification, regression problems may also form the focus of feature extraction. The author has also developed a version of the MMDA algorithm for solving regression tasks, with a focus on the retrieval of correlation-free features. The name of the method is Kernel Decorrelated Learning Regression (KDLR)

[65]. Based on tests performed over standard regression problems we can state that the approach leads to more efficient regression in practice.

I/4. The author proposed the combination of the statistics-based average derivative estimation method on the one hand, and kernel functions on the other hand. The aim of this novel method called Kernel Average Derivative Estimation (KADE) is the identification of sub-spaces that are relevant from a regression's point of view [65]. By testing on artificial data and comparing relating algorithms the author proved that the identified sub-spaces enable more effective regression in a good number of cases.

Novel kernel-based classification algorithms form the second group of results. These results are detailed in Chapter 4 and 5.

II/1. The author defined a family of hyperplane-based classification methods [34]. He proposed three modifications, each following traditional geometrical concepts: i) he used various loss functions in hyperplane-based classification; ii) he applied linear regression in a unique way to improve classification; iii) he embedded the output space into the input space and he developed the Minor Component Classifier (MCC) method, which defines a classification hyperplane with the help of an eigenvector pertaining to the smallest eigenvalue of a sample point matrix. The author formed the testing environment of the methods and performed demonstrational tests. Results prove that the methods develop by the author are comparable to results performed by SVM [61].

II/2. The author constructed a classification scheme called Convex Machine Technique, which applies a rare combination basis functions. The method contains a number of machine learning techniques, such as: Support Vector Machine (SVM) [61], Smooth Support Vector Machine (SSVM) [39], Least Square Support Vector Machine (LSVM) [64], Kernel Logistic Regression (KLR) [21], just to mention a few. Inspired by basic numeric mathematical methods, the author also developed three base function selection techniques (RANDOM, MGRAMM, CORR) [35], which he tested on certain elements of the UCI database [9].

II/3. He further developed three complex base function selection techniques (SFS, SFFS, PTA) in order to improve the efficiency of classification on the one hand, and to decrease the size complexity of the classification model on the other hand. The defined methods build on the analogy of state-of-the-art feature space selection techniques. Base on test results, it can be declared that these methods support effective classification [65].

Finally, Table A.1 summarizes which publication covers which method of the thesis.

Thesis Topics	[31]	[34]	[35]	[36]	[37]	[65]	*Chapter*	*Type*
MMDA	•						2	Feature Extraction
MMDA FACE version	•				•		2	Feature Extraction
KDLR	•					•	3	Feature Extraction
KADE						•	3	Feature Extraction
Hyperplane Classifiers		•					4	Classification
Convex Networks			•	•			5	Classification
Basic Basis Selection Methods			•				5	Basis Selection
Complex Basis Selection Methods				•			5	Basis Selection

Table A.1: The relation between the thesis topics and the corresponding publications.

A.2 Summary in Hungarian

Bevezetés

A modellalkotás a tudomány talán legfontosabb komponense, amely absztrakt módon minden tudományágban azonos. Modelleket alkotunk és megvizsgáljuk azok működését, módosításokat végzünk rajta mindaddig, amíg el nem érjük kitűzött célunkat. A mesterséges intelligencia egy különös tudományág, amely rendkívül jó táptalajt ad a számítógépes modellalkotás szerelmeseinek.

A gépi tanulás a mesterséges intelligencia tudományterület részét képezi [59]. Olyan algoritmusok konstrukcióját foglalja magában, amelyek a szamítógépet "tanulási" képességekkel ruházzák fel. A tanulásnak rendszerint két különböző megközelítése van: induktív és deduktív. A tézis az induktív irányzatot követi, amely rendszerint szabályokat vagy deszkriptív mintákat nyer ki masszív adathalmazokból (a tézisben az általánosítást a statisztikai megközelítés módszertanával végezzük). A gépi tanulás fókuszában jelen pillanatban leginkább az automatikus információ kinyerés összetett feladata áll. Ezen kívül fontos alkalmazásokat jelent - a teljesség igénye nélkül - a természetes nyelvfeldolgozás, a szintaktikus mintafelismerés, kereső motorok javítása, orvosi diagnosztika, bioinformatika, beszédfelismerés, tárgyak azonosítása és például számítógépes játékok intelligenciával történő felruházása. Bizonyos gépi tanulási módszerek az embert próbálják kiiktatni az adatanalízis folyamatából, míg más eljárások éppen az ember gép interakciót próbálják emberibbé tenni.

A gépi tanulás, a jelen kor technológiai szintje mellett a mesterséges intelligencia leginkább kutatott és legrelevánsabban fejlődő területévé vált. A dolgozat a gépi tanulás legújszerűbb módszertanához a kernel módszerek világához kapcsolódó eljárások konstrukciójával és vizsgálatával foglalkozik.

A kernel módszerek (KM) a mintafelismerés algoritmusainak egy olyan családja [61], amelynek legjelentősebb tagja a Support Vector Machine (SVM) [69]. A mintafelismerés általános feladata nem más, mint reprezentatív összefüggések keresése és tanulmányozása (például klaszterek, korrelációs összefüggések, klasszifikációs döntések vizsgálata) általános adatokon (vektorok, dokumentumok, szekvenciák, képek, stb.)

A KM megközelítés a nevét a kernel függvényekről kapta, amelyek egy olyan származtatott tulajdonság térben dolgoznak, ahol a minták tényleges koordinátáit soha nem kell kiszámolni. A módszerek csak a mintapontok páronként vett skalárszorzatára támaszkodnak, amelyeket implicit módon a kernel függvények alkalmazásával számítanak ki.

A kernel modszerek csaladjába beletartozik az SVM-en kívül számos más algoritmus: különböző regressziós eljárások, a Fisher-féle lineáris diszkrimináns analízis (LDA) [25], a főkomponens analízis (PCA) [29], a kanonikus korreláció analízis (CCA) [2], a 'ridge' regresszió [49], a spektrális klaszterezés [48], és még sok más eljárás. Általánosságban elmondható, hogy a kernel módszerek többsége hatékonyan megoldható feladatokra, konvex optimalizációra vagy sajátérték-sajátvektor problémára vezetnek.

Eredmények

A disszertáció tézisei lényegében két különböző módon két-két csoportra oszthatók. Az egyik osztályozás szerint a szerző eredményei a gépi tanulás témakörének kernel-alapú tulajdonságkinyerő és klasszifikációs módszereinek tárgykörébe esik. A másik felosztás szerint pedig algoritmikus konstrukciókról és gyakorlati alkalmazásokról beszélhetünk. A következőkben követve a disszertáció felépítését az első felosztás szerint vesszük számba az elért eredményeket. Fontosnak tartjuk megjegyezni, hogy az alább részletezett eredménylista a kapcsolódó cikkek nóvumainak csak azon részét katalogizálja, ahol a disszertáció szerzőjének hozzajárulása volt domináns.

Az eredmények első csoportját a szerző Kernel alapú tulajdonságkinyerő módszerei képezik. Ezen eredményeket a 2-es és 3-as fejezetek írják le a tézisben.

I/1. A szerző kidolgozta az MMDA algoritmus direkt változatát [31; 37]. A módszer egy olyan tulajdonságkinyerő eljárás, amely megnövelheti a klasszifikációs módszerek hatékonyságát. Az UCI gépi tanulási adatbázis számos példáján sikerült bizonyítani az eljárás használatának létjogosultságát [9].

I/2. Az MMDA algoritmus a motivációjából következően alkalmas nagydimenziós tulajdonságterek redukálására a nagyobb klasszifikációs hatékonyság növelése érdekében. A szerző megkonstruálta a módszer arcfelismerésre kidolgozott módosított változatát. A FERET 'gold standardot' képező arcfelismerési adatbázis [17] segítségével bizonyította a bevezetett eljárás eredményességét. Az irodalomban fellelhető eredményeket sikerült több ponton jelentősen meghaladni [37].

I/3. A tulajdonságkinyerés fókuszában a klasszifikáción túl a regresszió is állhat. A szerző kidolgozta az MMDA algoritmus regressziós problémák megoldására adaptált változatát, korrelációmentes tulajdonságok kinyerésére. A módszer neve Kernel Decorrelated Learning Regression (KDLR) [65]. Standard regressziós feladatokon történő teszt alapján kimondható, hogy az eljárás a gyakorlatban hatékony regresszióhoz vezethez.

I/4. A szerző javaslatot tett a statisztikából ismert átlagos deriváltbecslő eljárás és a kernel függvények alkalmazásának kombinációjára. A Kernel Average Derivative Estimation-nek (KADE) elnevezett eljárás célja a regresszió szempontjából releváns alterek megtalálása [65]. Mesterséges adatokon történő teszteléssel és a kapcsolódó algoritmusokkal való összehasonlítással a szerző kimutatta, hogy a megtalált alterek számos esetben hatékonyabb regressziót tesznek lehetővé.

A tézis eredményeinek második csoportjába újszerű Kernel alapú klasszifikációs algoritmusok tartoznak. Az eredményeket a 4-es és 5-ös fejezetek részletezik.

II/1. A szerző definiálta hipersík alapú klasszifikációs módszerek egy családját [34]. Három geometriai megfontolásokat követő módosítást/konstrukciót javasolt. i) különféle veszteségfüggvényeket használt a hipersik alapú klasszifikációban. ii) a lineáris regressziót sajátos módon alkalmazta klasszifikációhoz. iii) az output teret beágyazta az input térbe és kidolgozta a Minor Component Classifier (MCC) modszert, amely egy a mintapontokbol számított mátrix legkisebb sajátértékéhez tartozó sajátvektora segítségével definiál klasszifikációs hipersíkot. A szerző kialakította a módszerek tesztelési környezetét, majd végrehajtotta a működést demonstráló teszteket. Az eredmények azt igazolják, hogy az SVM-el [61] kompetitív eljárások kialakítására került sor.

II/2. A szerző megkonstruált egy klasszifikációs sémát az ún. 'Convex Machine' technikát, amely bázisfügvények ritka kombinációját alkalmazza. A kidolgozott módszertan magában foglal számos gépi tanulási technikát. A teljesség igénye nélkül ezek közé tartozik a Support Vector Machine (SVM) [61], a Smooth Support Vector Machine [39], a Least Square Support Vector Machine (LSVM) [64] es a Kernel Logistic Regression (KLR) [21]. Kialakított továbbá három alapvető numerikus matematikai módszerek által inspirált bázisfüggvény-kiválasztási módszert (RANDOM, MGRAMM, CORR) [35], amelyeket tesztelt az UCI adatbázis [9] egyes elemein.

II/3. A szerző kidolgozott három komplexebb bázisfüggvényszelekciós módszert (SFS, SFFS és PTA) a klasszifikáció hatékonyságának javítására és a klasszifikációs modell méret-komplexitásának csökkentésére. A definiált eljárások az irodalomból ismert hatékony tulajdonságtér-szelekciós módszerek analógiájára épülnek. A kialakitott tesztek eredménye alapján elmondható, hogy ezek az eljárások segítik a hatékony klasszifikációt [65].

Az összefoglaló végén az A.2-es táblázat mutatja a tézispontok és azok publikáltságának összefüggését.

Tézis eredmények	[31]	[34]	[35]	[36]	[37]	[65]	Fejezet	Eredmény kategóriája
MMDA	•						2	Tulajdonságkinyerés
MMDA arcfelismerő	•				•		2	Tulajdonságkinyerés
KDLR	•					•	3	Tulajdonságkinyerés
KADE						•	3	Tulajdonságkinyerés
Hipersík alapú klasszifikáció		•					4	Klasszifikáció
Konvex gépek			•	•			5	Klasszifikáció
Alapvető bázisszelekciós eljárások			•				5	Bázisfüggvény-szelekció
Komplex bázisszelekciós eljárások				•			5	Bázisfüggvény-szelekció

A.2. táblázat. A tézisek és a kapcsolódó publikációk összefüggése.

Bibliography

[1] M. Aizerman, E. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[2] S. Akaho. A kernel method for canonical correlation analysis. In *International Meeting of Psychometric Society (IMPS), Osaka*, 2001.

[3] S. Asharaf, M. Narasimha Murty, and S. K. Shevade. Multiclass core vector machine. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 41–48, New York, NY, USA, 2007. ACM.

[4] K. Baek, B.A.Draper, J.R. Beveridge, and K. She. PCA vs. ICA: A comparison on the FERET data set. In *Proc. of the Fourth International Conference on Computer Vision, Pattern Recognition and Image Processing*, pages 824–827, Durham, NC, USA, 2002.

[5] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12(10):2385–2404, 2000.

[6] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:711–720, 1997.

[7] K. P. Bennett and C. Campbell. Support vector machines: Hype or hallelujah? *SIGKDD Explorations*, 2(2):1–13, 2000.

[8] D.P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice Hall, 1989; republished by Athena Scientific, 1997.

[9] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, http://www.ics.uci.edu/∼mlearn/mlrepository.html, 1998.

[10] B.E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learing Theory*, pages 144–152, 1992.

[11] P. S. Bradley and O. L. Mangasarian. Massive data discrimination via linear support vector machines. *Optimization Methods and Softwares*, 13:1–10, 2000.

[12] A. Christmann and I. Steinwart. On robustness properties of convex risk minimization methods for pattern recognition. *J. Mach. Learn. Res.*, 5:1007–1034, 2004.

[13] R. Collobert and S. Bengio. Svmtorch: Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1:143–160, 2001.

[14] D. Cook. Principle Hessian directions revisited. *J. Amer. Statist. Ass.*, 93(441):84–93, 1998.

[15] T. M. Cover and J. V. Campenhout. On the possible orderings in the measurement selection problem. *IEEE Trans. Systems, Man, and Cybernetics*, 7:657–661, 1977.

[16] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[17] M. Teixeira D. Bolme, R. Beveridge and B. Draper. The csu face identification evaluation system: Its purpose, features and structure. In *International Conference on Vision Systems*, pages 304–311. Springer-Verlag, 2003.

[18] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1):1–50, 2000.

[19] S. Fine and K. Scheinberg. Efficient svm training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

[20] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences,*, 55(1):119–139, 1997.

[21] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–407, 2000.

[22] T. Friess, N. Cristianini, and C. Campbell. The kernel adatron algorithm: a fast and simple learning procedure for support vector machine. In *Proc. 15th International Conference on Machine Learning*. Morgan Kaufman Publishers, 1998.

[23] D. R. Fuhrmann and B. Liu. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1):1–50, 2000.

[24] K. Fukumizu, F.R. Bach, and M.I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

[25] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, San Diego, California, 1990.

[26] I.S. Helland. Partial least squares regression and statistical models. *Scand. J. Statist.*, 17:97–114, 1990.

[27] R. Herbrich and T. Graepel. A PAC-bayesian margin bound for linear classifiers: Why SVMs work. In *Advances in Neural Information Processing Systems 13*, pages 224–230, Cambridge, MA, 2000. MIT Press.

[28] C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425, 2002.

[29] I. T Jolliffe. *Principal component analysis*. New York : Springer, 2002.

[30] A. Kocsor. Kernel springy discriminant analysis. In *NATO ASI on Learning Theory and Practice (LTP 2002)*, K.U. Leuven, Belgium, 2002.

[31] A. Kocsor, K. Kovács, and Cs. Szepesvári. Margin maximizing discriminant analysis. In Fosca Giannotti et al. Jean-François Boulicaut, Floriana Esposito, editor, *Machine Learning: ECML 2004: 15th European Conference on Machine Learning, vol. 3201*, pages 227–238. Springer-Verlag GmbH, September 2004.

[32] A. Kocsor and K. Kovács. Kernel springy discriminant analysis and its application to a phonological awareness teaching system. In K. Pala P. Sojka, I. Kopeek, editor, *Text, Speech and Dialogue : 5th International Conference, TSD 2002, LNAI vol. 2448*, pages 325–328, Brno, Czech Republic, September 2002. Springer-Verlag GmbH.

[33] E. B. Kong and T. G. Dietterich. Error-correcting output coding corrects bias and variance. In *International Conference on Machine Learning (ICML)*, pages 313–321, 1995.

[34] K. Kovács and A. Kocsor. Various hyperplane classifiers using kernel feature spaces. *Acta Cybernetica*, 16(2):271–278, 2003.

[35] K. Kovács and A. Kocsor. Classification using sparse combination of basis functions. *Acta Cybernetica*, 17(2):311–323, 2004.

[36] K. Kovács and A. Kocsor. Improving a basis function based classification method using feature selection algorithms, accepted for ieee international workshop on soft computing applications. In *IEEE International Workshop on Soft Computing Applications, (IEEE-SOFA)*, pages 208–211, 2005.

[37] K. Kovács, A. Kocsor, and Cs. Szepesvári. Maximum margin discriminant analysis based face recognition. In M. Vincze D. Chetverikov, L. Czuni, editor, *Proceedings of the Joint Hungarian-Austrian Conference on Image Processing and Pattern Recognition, HACIPPR*, pages 71–78. Oesterreichische Computer Gesellschaft, 2005.

[38] A. Krogh and J. Vedelsby. Neural network ensembles, cross validation, and active learning. In *Advances in Neural Information Processing Systems NIPS 11*, pages 231–238, 1995.

[39] Y. Lee and O. L. Mangasarian. SSVM: A smooth support vector machine. *Computational Optimization and Applications*, 20:5–22, 2001.

[40] H. Li, T. Jiang, and K. Zhang. Efficient and robust feature extraction by maximum margin criterion. In *Advances in Neural Information Processing Systems 16*, Vancouver, Canada, 2003.

[41] K.-C. Li. Sliced inverse regression for dimension reduction. (With discussion). *J. Amer. Statist. Ass.*, 86(414):316–342, 1991.

[42] K.-C. Li. On principal Hessian directions for data visualization and dimension reduction: Another application of Stein's lemma. *J. Amer. Statist. Ass.*, 87(420):1025–1039, 1992.

[43] X. Liu, A. Srivastava, and D. Wang. Intrinsic generalization analysis of low dimensional representations. *Neural Networks*, 16(5-6):537–545, 2003.

[44] F. Luo, R. Unbehauen, and A. Cichocki. A minor component analysis algorithm. *Neural Networks*, 10(2):291–297, 1997.

[45] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Roy. Soc. London, A*, 209:415–446, 1909.

[46] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, editors, *Neural Networks for Signal Processing IX*, pages 41–48. IEEE, 1999.

[47] B. Moghaddam, T. Jebara, and A. Pentland. Bayesian face recognition. *Pattern Recognition*, 33(11):1771–1782, 2000.

[48] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14 (NIPS)*, pages 849–856. MIT Press, Cambridge, 2002.

[49] I. Nouretdinov, T. Melluish, and V. Vovk. Ridge regressioon confidence machine. In *Proc. 18th International Conf. on Machine Learning*, pages 385–392. Morgan Kaufmann, San Francisco, CA, 2001.

[50] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods: support vector learning*, pages 185–208, Cambridge, MA, USA, 1999. MIT Press.

[51] T. Poggio. On optimal nonlinear associative recall. *Biological Cybernetics*, 19:201–209, 1975.

[52] T. Poggio, S. Mukherjee, R. Rifkin, A. Rakhlin, and A. Verri. B. In *Proceedings of the Conference on Uncertainty in Geometric Computations*, 2001.

[53] P. Pudil, J. Novovicova, and J. Kittler. Feature selection based on the approximation of class densities by finite mixtures of the special type. *Pattern Recognition*, 28(9):1389–1397, 1995.

[54] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[55] R. Rosipal and L. J. Trejo. Kernel partial least squares regression in reproducing kernel hilbert space. *Journal of Machine Learning Research*, 2:97–123, 2001.

[56] V. Roth. The generalized lasso. *IEEE Transactions on Neural Networks*, 15(1):16–28, 2004.

[57] V. Roth and V. Steinhage. Nonlinear discriminant analysis using kernel functions. In *Advances in Neural Information Processing Systems NIPS 12*, pages 568–574, 1999.

[58] G. Rätsch, S. Mika, B. Schölkopf, and K.-R. Müller. Constructing boosting algorithms from svms: an application to one-class classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1184–1199, 09 2002.

[59] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.

[60] A. Samarov. Exploring regression structure using nonparametric functional estimation. *J. Amer. Statist. Ass.*, 88(423):836–847, 1993.

[61] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, 2002.

[62] B. Schölkopf, A. Smola, and K. Müller. Kernel principal component analysis. In *Advances in Kernel Methods - SV Learning*, pages 327–352. MIT Press, Cambridge, MA, 1999.

[63] A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans. Introduction to large margin classifiers. In *Advances in Large Margin Classifiers*, pages 1–29. MIT Press, Cambridge, MA, 2000.

[64] J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

[65] Cs. Szepesvári, A. Kocsor, and K. Kovács. Kernel machine based feature extraction algorithm for regression problems. In Lorenza Saitta Ramon López de Mántaras, editor, *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI 2004*, pages 1091–1091, Valencia, Spain, August 2004. IOS Press.

[66] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems*. W. H. Winston, Washington, D.C., 1977.

[67] I. W. Tsang, A. Kocsor, and J. T. Kwok. Simpler core vector machines with enclosing balls. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 911–918, New York, NY, USA, 2007. ACM.

[68] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, NY, USA, 1995.

[69] V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons Inc., 1998.

[70] G. Wahba. *Splines models for Observational Data*. Vol. 59, SIAM, Philadelphia, 1990.

[71] G. Wahba. Support vector machines, reproducing kernel Hilbert spaces, and randomized GACV. In *Advances in kernel methods: support vector learning*, pages 69–88. MIT Press, 1999.

[72] J. Weston and C. Watkins. Support vector machines for multiclass pattern recognition. In *Proceedings of the Seventh European Symposium On Artificial Neural Networks (ESANN)*, pages 219–224, 1999.

[73] C. Williams and M. Seeger. Using the nystrom method to speed up kernel machines. In editors T. K. Leen, T. G. Diettrich V. Tresp, editor, *Advances in Neural Information Processing Systems 13*, pages 682–688. MIT Press, 2001.

[74] H. Wold. Partial least squares. In *Encyclopedia of Statistical Sciences*, volume 6, pages 581–591. Wiley, New York, 1985.

[75] J. Zhang and Y. Liu. SVM decision boundary based discriminative subspace induction. Technical Report CMU-RI-TR-02-15, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, June 2002.

[76] M. Zhu and A.M. Martinez. Optimal subclass discovery for discriminant analysis. In *Proc. of IEEE Workshop on Learning in Computer Vision and Pattern Recognition (LCVPR)*, 2004.