Fys-3921
Master's Thesis in Electrical Engineering

Information Theoretic Learning for Pattern Classification

by

Ola Kvisle Storås

December, 2007

FACULTY OF SCIENCE
Department of Physics and Technology
University of Tromsø

Abstract

This thesis is a study of pattern classification based on information theoretic criteria. Information theoretic criteria are important measures based on entropy and divergence between data distributions. First, the basic concepts of pattern classification with the well known Bayes classification rule as a starting point is discussed. We discuss how the Parzen window estimator may be used to find good density estimates. The Parzen window density estimator can be used to estimate cost functions based on information theoretic criteria. Furthermore, we explain a model of an information theoretic learning machine. With cost functions based on information theoretic criteria, we argue that a learning machine potentially captures much more information about a data set than the traditional mean squared error cost (MSE) function. We find that there is a geometric link between information theoretic cost functions estimated using Parzen windowing, and mean vectors in a Mercer kernel feature space. This link is used to propose and implement different classifiers based on the integrated squared error (ISE) divergence measure, operating implicitly in a Mercer kernel feature space. We also apply spectral methods to implement the same ISE classifiers working in approximations of Mercer kernel feature spaces. We investigate the performance of the classifiers when we weight each data point with the the inverse of the probability density function at that point. We find that the ISE classifiers working implicitly in the Mercer kernel feature space performs similar to a Parzen window based Bayes classifier. Using a weighted inner-product definition gives slightly better results for some data sets, while for other data sets the classification rates are slightly worse. When comparing the results between the implicit ISE classifier using unweighted data points and the Parzen window Bayes classifier, some of the results indicate that the ISE classifier favor the classes with highest entropy.

Acknowledgment

I would like to thank my supervisor in this thesis, Robert Jenssen, for introducing and explaining many new fascinating ideas in pattern analysis during the work with this thesis.

Contents

Chapter 1

Introduction

Patterns are used to describe any relations, regularities or structure inherent in a data set generated by a source. Similar patterns are often grouped into a class. By detecting significant patterns in the available data, a learning machine can make predictions about new data coming from the same source [26]. In pattern classification we use some labeled training data set, where each label represent a class, and use a learning machine to predict the correct class label for a new unlabeled sample. A learning machine may be viewed as a device that adjust a set of parameters through a learning process. For each new set of training data given to the machine, the parameters are updated using a criteria that captures the wanted information to describe the data in a new form. Pattern classification is one of the fundamental problems in machine learning and signal processing [29], [26],[4]. It is an important part in computer vision, medical imaging, optical character recognition, geostatistics, handwriting recognition, biometric identification, natural language processing, document classification, email spam detection and credit scoring, to list a few examples [29], [4], [23], [1], [15], [8].

Information theoretic learning (ITL) methods emerged in [18] and [5]. Information theoretic learning here refers to the use of a general learning machine as we describe in Section 4.2, where a criteria related to Renyi's quadratic entropy is used to update the learning process. The criteria often use Renyi's quadratic entropy to find divergence measures between data in different distributions. A divergence measure can be thought of as a generalization of algebraic distance measures (such as the Euclidean norm) to probability distribution spaces [5]. In [18] two important information theoretic divergence measures were presented, one is based on the Cauchy-Schwarz (CS) inequality and the other on the integrated squared error (ISE) between two probability distributions.

The Renyi's quadratic entropy is estimated using a Parzen window density estimation method, and may be thought of as a generalization of variance to processes with non-Gaussian distributions [5]. In [18] and [5] information theoretic concepts are explained and used in time series prediction, independent component analysis (ICA), feature extraction and blind source deconvolution.

Independent of ITL, a number of kernel methods have emerged in the recent years. Kernel methods generally solve machine learning problems in two parts: A module, also known as a kernel function, performs a mapping of data to a new feature space. In this new feature space a learning algorithm is used to discover linear patterns [26]. The use of a kernel function allows us to operate implicitly in a possible high dimensional space through evaluations of inner-products. This is also known as the "kernel trick". The advantage of operating in high dimensional spaces is that the probability that the data is linearly separable increases with the number of dimensions we operate in [2]. Kernel methods have been used successfully in various fields of machine learning, and include algorithms such as support vector machines (SVMs), kernel Fisher discriminant analysis (KFD) and kernel principal component analysis (KPCA) [17].

The affinity matrix of a data set is calculated with the pairwise inner-products of data samples with a kernel function, Element (i, j) of this matrix contains the inner-product between data sample i and j, computed with a kernel function. The inner-products of the affinity matrix may be weighted such that each data point is multiplied with the inverse overall probability density function at that point. This affinity matrix with weighted inner-products is referred to as the Laplacian matrix. Spectral methods based on the spectral properties, i.e. the eigenvectors and eigenvalues of the affinity matrix or the Laplacian matrix, have been popular in recent clustering applications [5],[9].

It has been shown [13] and [9], that when using the non-parametric Parzen window density estimator with a Mercer kernel function to estimate the Renyi's quadratic entropy for a data set, the result may be interpreted in terms of a mean vector in a Mercer kernel feature space. By measuring distances between class mean vectors in a Mercer kernel feature space we can create different information theoretic divergence measures between class distributions. The CS divergence measure is shown in [13] to be related to measuring angles between class mean vectors in a Mercer kernel feature space and have interesting connections to graph theory. The ISE divergence measure is in a similar manner shown to be related to the Euclidean distance between mean vectors in a Mercer kernel space [9].

The link between ITL and Mercer kernel methods has been used to develop recent classifier [12], [9], and clustering [9], [11], [10] algorithms based on the CS divergence measure. The classifier proposed in [12] works implicitly in a Mercer kernel feature space, while the CS based spectral clustering algorithms work in approximate Mercer kernel spaces spanned by the principal eigenvectors of the Laplacian matrix. These methods have all used weighted inner-product kernel functions when calculating the CS divergence measure.

This thesis is inspired by the use of the CS divergence measure in both classification and clustering applications. We provide necessary background on information theoretic learning concepts to understand newly discovered, important relations between Mercer kernel theory, information theoretic measures and density estimation. In particular we focus on the properties of the ISE information theoretic divergence measure and use Mercer kernel properties and geometric properties of this measure to argue that it may be used as a cost function in an information theoretic classifier. Previous information theoretic classifiers have to our knowledge only been implemented using the weighted CS divergence measure. Thus we aim to use the ISE divergence measure in a similar manner. We investigate performance and properties of classifiers based on the ISE divergence using both weighted and unweighted inner-products, operating implicitly in Mercer kernel spaces through evaluations with Mercer kernels.

We also investigate spectral versions of the ISE classifiers, where we eigendecompose the Laplacian data matrix or the affinity matrix. None of the ISE divergence based classifiers presented in this thesis have been presented before, so we could not know how they would perform. We choose to compare the results obtained with our implementation of the Bayes classifier, both because it is a well known classifier and because we find out that the ISE based classifier rule may be expressed in a very similar way to the Bayes classifier rule.

We present two different versions of the ISE classifier operating implicitly in different Mercer spaces. The standard ISE and the Laplacian ISE classifier, operating on unweighted and weighted training data, respectively. We also develop spectral versions of these classifiers, the spectral ISE classifier and the spectral Laplacian ISE classifier, working in the approximated Mercer spaces spanned by the principal eigenvectors of the affinity and Laplacian matrices, respectively. The ISE based classifiers working implicitly in a Mercer kernel space in general seems to give best results. For some data sets the Laplacian induced weights improve the classification rates, but in others it reduces or does not change the classification rates significantly.

1.0.1 Quick summary of content in this thesis.

- We provide necessary background information about the relatively new concepts used in information theoretic learning. We also review background information necessary to understand basic density estimation and pattern classification.

- We introduce and investigate new classifiers based on the information theoretic ISE divergence measure and kernel methods, using both weighted and unweighted data.

- We investigate relations between an ISE divergence based classifier operating implicitly in a Mercer kernel space and the well known Parzen window based Bayes classifier. We find that using unweighted data the ISE classifier is comparable to the Bayes classifier with slightly different properties.

- We use the spectral properties of the affinity and Laplacian matrices of the data, to propose and investigate ISE based classifiers working directly in approximated Mercer kernel spaces. We note that in most cases the spectral versions of the ISE classifier perform slightly worse than the versions working implicitly in Mercer spaces.

1.1 Definitions and notation used

$\mathbf{x}$,y	A training pattern and class label
i,N	Counter and number of patterns
i.i.d	Independent identically distributed
pdf	Probability density function
ISE	Integrated Squared error
MISE	Mean Integrated Squared Error
AMISE	Asymptotic Mean Integrated Squared Error
IP	Information Potential
ITL	Information Theoretic Learning
$W_{\sigma^2}(\cdot,\cdot)$	A Gaussian kernel, with bandwidth σ^2
$K_h(\cdot,\cdot)$	A general Mercer kernel, with bandwidth h
$\mathcal{F}$	A Mercer kernel feature space
d	Dimensionality of data
C	Number of classes

Table 1.1: Definitions and abbreviations

Density functions are usually referenced with small letters e.g. $p(x)$. Probabilities are referenced with large letters e.g. $P(x)$. Integrals with no limits are assumed to be with lower limit $-\infty$ and upper limit ∞. The sample $\mathbf{x}$ may take any value in the d dimensional space, unless otherwise specified. Expectations with regard to a variable or function f is denoted by $E_f\{\cdot\}$ if it may be unclear what we calculate the expected value with respect to.

1.2 Structure and literature

Structure of this thesis This thesis is divided in three parts. In Part I we present the theory necessary to understand and implement our classifiers. Part II contains analysis and experiments done to check how the theory works on some popular data sets. In Part III we conclude the thesis and suggests further work that may be done.

- Chapter 2 introduce the basic concepts of pattern classification. The Bayes classifier is used as an example and shown to give a minimum probability of classification error.
- Chapter 3 discusses various methods for density estimation with an emphasis on the Parzen window density estimator and its properties.
- Chapter 4 explains the concept of an information theoretic learning machine. A detailed discussion of various information theoretic criteria is given. Some sample based estimators for information theoretic criteria are discussed.

- Chapter 5 explain the kernel trick and how it can be used to express information theoretic measures in a Mercer kernel space from a simple geometric viewpoint. The standard ISE classifier is developed using this geometric view of the ISE divergence between two distributions. This classifier is shown in a special case to be equal to the familiar Bayes classifier using density estimates. Some relations between the ISE divergence measure and graph theory are discussed.
- Chapter 6 presents the Laplacian ISE classifier, a modified version of the standard ISE classifier using weighted data samples.
- Chapter 7 presents spectral versions of the standard ISE and Laplacian classifier, working in approximated Mercer kernel spaces.
- Chapter 8 provides a short analysis of the effect of different kernel sizes and kernel types used in the previously presented classifiers.
- Chapter 9 presents and discusses results found using the different classifiers on some popular data sets. We try to illustrate the effects of weighting the data samples and how data distributes in the approximate Mercer kernel spaces.
- Chapter 10 concludes this thesis and suggests some work that may be done in the future.
- The appendix contains some additional classification results not listed in Chapter 9.

Literature Information theory in general is covered by [3] and the article which first used entropy as a measure in communication, [25]. The principles behind information theoretic learning was mainly introduced in [18] with an nice overview in [5] and [9]. Good introduction books to pattern classification methods are [29],[4] and [8]. The CS divergence based classifier which inspired much of the work with our ISE divergence based classifiers is presented in [12]. Important relations between Mercer kernel functions, Parzen window density estimators, CS divergence and graph theory is reviewed in [13]. A survey of kernel methods used in pattern analysis is given in [26] and [17]. Spectral methods used in this thesis are influenced by material in [26] [9], [24], [6] and [31].

Part I

Theory

Chapter 2

Pattern classification basics

This chapter gives a definition of pattern classification. It also reviews the well known Bayes classification rule, which is shown to give a minimum probability of classification error. The Bayes classifier may thus be used as a benchmark when testing other classifiers with regard to error rates. In Part II, Chapter 9, we use an implementation of this classifier to compare with the new classifiers which will be presented in later chapters of this thesis.

Definition The task of classification is to find a rule, which based on observations of training patterns assigns an unclassified pattern to one of several possible classes. A classification rule with two different classes is to estimate a function

$$g : \mathbb{R}^d \longrightarrow \{-1, 1\},$$

using input-output training data pairs generated i.i.d according to an unknown probability distribution

$$p(\mathbf{x}, y), \qquad (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N) \in \mathbb{R}^{\mathrm{d}} \times Y, \qquad Y\{-1, 1\}$$

such that g will correctly classify a new sample $\mathbf{x}$ [17]. A sample $\mathbf{x}$ is assigned to the class labeled $+1$ if $g(\mathbf{x}) \geq 0$. The sample $\mathbf{x}$ is assumed to be generated from the same pdf $p(\mathbf{x}, y)$ as the training data.

2.1 The Bayes classifier

To explain more in detail how we can define a classifier we begin with the Bayes classifier. The reason for starting with this is that it can be shown to give an optimal result with regard to minimum probability of classification error, under certain conditions. This will be proved in the next section. It is also well known and has some theoretical connections to the ISE classifier, which will be explained in Section 5.3.1.

We want to classify an unknown feature vector $\mathbf{x}$ to one of C possible classes $\omega_1, \ldots, \omega_C$ in a way that assigns $\mathbf{x}$ to the class where it's "most likely" to belong. We define what is "most likely" with the probabilities $P(\omega_i|\mathbf{x})$, $i = 1, \ldots, C$, also known as the *a posteriori probabilities*. A possible classification rule is to assign $\mathbf{x}$ to the class ω^* satisfying [29]

$$\omega^* = \max_{\omega_i} P(\omega_i|\mathbf{x}), \qquad i = 1, \ldots, C. \tag{2.1}$$

Assume that the *a priori probabilities* $P(\omega_i)$ $i = 1, \ldots, C$, are known. If they are unknown they can be estimated from our training data as $P(\omega_i) = \frac{n_i}{N}$ where N is the total number of training samples, and n_i is the number of samples belonging to class ω_i. The class-conditional probability density functions $p(\mathbf{x}|\omega_i)$, also known as *likelihood functions of* ω_i, *with respect to* $\mathbf{x}$, are also assumed to be known [29]. If these are unknown we will see later how they can be estimated.[1] To calculate the a posteriori probabilities we can use Bayes rule [29]

$$P(\omega_i|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_i)P(\omega_i)}{p(\mathbf{x})}, \qquad i = 1, \ldots, C, \tag{2.2}$$

where

$$p(\mathbf{x}) = \sum_i p(\mathbf{x} \mid \omega_i)P(\omega_i).$$

Since $p(\mathbf{x})$ is a common factor for all classes it may be ignored and we can use

$$\omega^* = \max_{\omega_i} p(\mathbf{x}|\omega_i)P(\omega_i), \qquad i = 1, \ldots, C, \tag{2.3}$$

as our classifier. If the a priori probabilities $P(\omega_i)$, are equal Eq. (2.3) reduces to

$$\omega^* = \max_{\omega_i} p(\mathbf{x}|\omega_i), \qquad i = 1, \ldots, C,$$

and the search for the most probable class for feature $\mathbf{x}$ reduces to evaluating the conditional pdfs at $\mathbf{x}$. Since the classifier in Eq. (2.3) is obtained using Bayes rule, it is often referred to as the Bayes classifier. In the two class case we use Eq. (2.3) to assign $\mathbf{x}$ to class ω_1 if

$$p(\mathbf{x}|\omega_1)P(\omega_1) > p(\mathbf{x}|\omega_2)P(\omega_2).$$

Our classification function can now be defined as to classify $\mathbf{x}$ to class ω_1 if

$$g(\mathbf{x}) = p(\mathbf{x}|\omega_1)P(\omega_1) - p(\mathbf{x}|\omega_2)P(\omega_2) \geq 0$$

2.2 Minimum probability of classification error for the Bayes classifier

In Fig. 2.1, the Bayesian classifier for two equiprobable classes for a one-dimensional feature vector x is illustrated. The region to the left of the dotted threshold line clearly contains most of $p(x, \omega_1) = p(x|\omega_1)P(\omega_1)$ and we define this as R_1, and the region to the right

[1]If $\mathbf{x}$ is discrete the likelihood functions become probabilities and are denoted with $P(\mathbf{x}|\omega_i)$

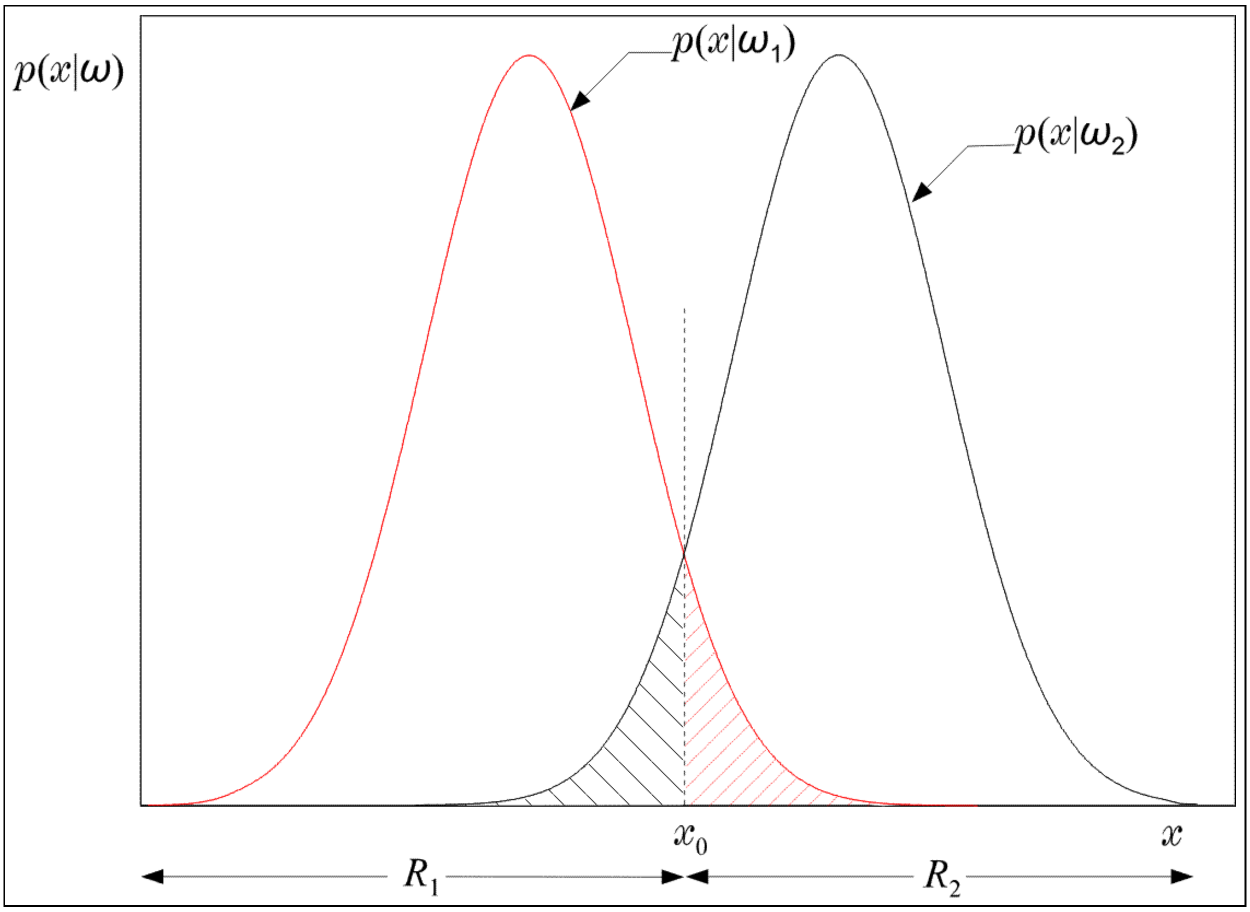

Figure 2.1: Illustration of two decision regions. Figure borrowed from [29]

of the threshold as R_2. Let R_1 and R_2 be the regions where we classify x to ω_1 and ω_2, respectively. The total area covered by $p(x|\omega_1)$ in region R_2 and $p(x|\omega_2)$ in R_1 will be the probability of causing classification errors. If the threshold is moved to the left or right, this area and probability will increase. This means that if we want to minimize the probability of an error, the decision regions R_1 and R_2 must be selected by moving the threshold so this area is as small as possible.

In a multiclass situation with a multidimensional feature vector $\mathbf{x}$ we have C different classes with decision regions $(R_1, \dots R_C)$ our feature vector $\mathbf{x}$ can be placed in. We now generalize the situation in Fig. 2.1. Writing the probability of a correct decision by the joint probability

$$P(\mathbf{x} \in R_i, \omega_i),$$

then the probability of erroneously assigning $\mathbf{x}$ to ω_j by not selecting the correct class ω_i is

$$P(\mathbf{x} \in R_j, \omega_i), \qquad \forall j \neq i.$$

The total probability for committing an error in classification is thus

$$\begin{aligned} P_e &= \bigcup_{\forall j \neq i} P(\mathbf{x} \in R_j, \omega_i), \\ &= \sum_{\forall j \neq i} P(\mathbf{x} \in R_j | \omega_i) P(\omega_i), \\ &= \sum_{\forall j \neq i} P(\omega_i) \int_{R_j} p(\mathbf{x}|\omega_i) \mathrm{d}\mathbf{x}, \\ &= \sum_{\forall j \neq i} \int_{R_j} P(\omega_i|\mathbf{x}) p(\mathbf{x}) \mathrm{d}\mathbf{x}. \end{aligned}$$

The total probability for correct classification is

$$\sum_i \int_{R_i} P(\omega_i|\mathbf{x}) p(\mathbf{x}) \mathrm{d}\mathbf{x} = 1 - P_e.$$

Hence

$$P_e = 1 - \sum_{\forall i} \int_{R_i} P(\omega_i|\mathbf{x}) p(\mathbf{x}) \mathrm{d}\mathbf{x}.$$

The error is clearly minimized when the regions R_i are selected in a way where

$$R_i : \qquad P(\omega_i|\mathbf{x}) > P(\omega_j|\mathbf{x}), \qquad \forall j \neq i,$$

which is the same as Eq. (2.1).

Chapter 3

Density estimation

The cost functions used by the classifiers discussed in this thesis are all dependent on finding some sort of density estimate for the class distributions of the data. The case where we know the distributions of the feature vectors in each class ω_i, given by the likelihood functions $p(\mathbf{x}|\omega_i)$, is unfortunately not the reality for most data sets. We have to find estimates of these distributions. There are basically two categories of methods for estimation of pdfs, parametric and non-parametric methods. In this section, for completeness, a short description of parametric methods for density estimation is given. For more details [29] is recommended. The non-parametric methods are far more important in information theoretic classification, since we often cannot assume that the data set has a parametric distribution shape. In particular the Parzen window method for density estimation will be investigated. Throughout this chapter we assume a data set of N samples, $x_i,\ i = 1, \ldots, N$, generated i.i.d from unknown distributions, unless otherwise is specified.

3.1 Parametric methods

Assume that the pdf to be estimated is described in parametric form by some unknown parameter vector θ, so it can be written as $f(\mathbf{x}; \theta)$. We have a limited number N of i.i.d training data, $x_1, \ldots, x_N$ available from our distribution. Using these samples we can use different methods to find an estimate of the parameters in θ such that the estimate $\hat{f}(\mathbf{x}; \theta)$ is as close as possible to the true pdf. This means that we assume that our data is generated from a distribution with a shape that is close to a parametric form, e.g we assume a Gaussian, Rayleigh or some other well known distribution and try to adjust its parameters to fit our data as good as possible. The parameters in θ are usually found by maximum likelihood estimation. In [29] some methods for parametric estimation are described, e.g. maximum likelihood estimation, mixture models, maximum entropy etc.

3.2 Non-parametric methods

To avoid the need to make assumptions about a parametric shape of the desired distribution, we must often use non-parametric methods. In this section, we review different methods of density estimation with a one-dimensional random variable x taken from a continuous, univariate density function $f(x)$. We start with the simple histogram method and expand it until we end up with the Parzen window estimator. Some of the most important properties of the Parzen window estimator are then discussed. In the last section, the Parzen window method is expanded to the case where we have multivariate distributions, where the variable $\mathbf{x}$ is a multidimensional vector.

3.2.1 The histogram density estimator

The oldest way to find a non-parametric estimate of a function is given by the *histogram* [30], [9]. Given an origin x_0, and a bin width h, the bins of the histogram are defined as $[x_0 + mh, x_0 + (m+1)h)$ for positive and negative integers m. The histogram estimate of the function $f(x)$ is then

$$\hat{f}(x) = \frac{1}{Nh}(\text{no of } x_i \text{ in same bin as } x). \tag{3.1}$$

This estimator is obviously discontinuous and not usable when we need to find derivatives.

3.2.2 The naive density estimator

This is also a variant of the histogram method. Define the pdf evaluated at x as

$$f(x) = \lim_{h \to 0} \frac{1}{2h} P(x - h < X < x + h) \tag{3.2}$$

The probability $P(x - h < X < x + h)$ can be estimated by counting the number of data samples falling into a bin of size $2h$ centered at x. This can be defined more precisely with a weight function

$$W(x) = \begin{cases} \frac{1}{2} & \text{if } |x| \leq 1 \\ 0 & \text{otherwise} \end{cases},$$

such that the *naive estimator* can be expressed as [9]

$$\hat{f}(x) = \frac{1}{Nh} \sum_{i=1}^{N} W\left(\frac{x - x_i}{h}\right). \tag{3.3}$$

Introducing a rescaling notation $W_h(u) = h^{-1} W(u/h)$ we rewrite Eq. (3.3) as

$$\hat{f}(x) = \frac{1}{N} \sum_{i=1}^{N} W_h(x - x_i). \tag{3.4}$$

From Eq. (3.3) we see that an estimate for the pdf at x is given by placing a "box" around each sample x_i with width $2h$ and height $(2Nh)^{-1}$ and sum up. This estimator is not continuous, since it is a sum of discontinuous functions.

3.2.3 The Parzen window density estimator

Parzen generalized the weight function $W_h(\cdot)$ in Eq. (3.4) to a *kernel function* or *Parzen window* which is a function satisfying [29]

$$K_h(x) \geq 0 \qquad \text{and} \qquad \int_x K_h(x)\mathrm{d}x = 1.$$

The subscript h refers to the *bandwidth* or *window width* of the kernel [30]. Usually $K(\cdot)$ is chosen to be a unimodal probability density function that is symmetric around zero. This makes sure that the estimator

$$\hat{f}(x) = \frac{1}{N}\sum_{i=1}^{N} K_h(x - x_i) \tag{3.5}$$

produces an estimate which is also a density.

To investigate some of its properties, we find expressions for the mean value and variance of Eq. (3.5). Let $\hat{f}(x)$ be the estimate of the true density $f(x)$ at x, with x' a random variable with density $f(x)$. Then

$$\begin{aligned} E\{\hat{f}(x)\} =& E\{K_h(x - x')\} \\ =& \int K_h(x - x')f(x')\mathrm{d}x' \\ =& K_h(x) \star f(x). \end{aligned} \tag{3.6}$$

The density estimate is therefore a smoothed version of the true density. The *bias* of the estimator is given by [30]

$$E\{\hat{f}(x)\} - f(x) = [K_h(x) \star f(x)] - f(x), \tag{3.7}$$

and the *variance* is [30]

$$\begin{aligned} Var\{\hat{f}(x)\} =& E\{[\hat{f}(x) - E\{\hat{f}(x)\}]^2\} \\ =& \frac{1}{N}\{(K_h^2(x) \star f(x)) - [(K_h(x) \star f(x)]^2\}. \end{aligned} \tag{3.8}$$

It is common to measure the closeness of the estimator $\hat{f}(x)$ to the target density $f(x)$ in the point x by the size of the mean squared error (MSE)

$$MSE\{\hat{f}(x)\} = E\left\{[\hat{f}(x) - f(x)]^2\right\},$$

which can be written as

$$\begin{aligned} MSE\{\hat{f}(x)\} =& \frac{1}{N}\{(K_h(x)^2 \star f(x)) - [(K_h(x) \star f(x))]^2\} + \{(K_h(x) \star f(x)) - f(x)\}^2 \\ =& Var\{\hat{f}(x)\} + [Bias\{\hat{f}(x)\}]^2. \end{aligned}$$

Instead of just estimating the function $f(x)$ at a single point we want to estimate it over the whole x-space. The mean integrated error (MISE) is a more appropriate measure for analyzing $\hat{f}(x)$ where

$$MISE\{\hat{f}(x)\} = \int MSE\{\hat{f}(x)\}\mathrm{d}x$$
$$= \int E\{[\hat{f}(x) - f(x)]^2\}\mathrm{d}x + \int Var\{\hat{f}(x)\}\mathrm{d}x. \tag{3.9}$$

The bias and variance term in Eq. (3.9) depend on the kernel width h in different ways. It has been shown that Eq. (3.9) for large sample sizes N, the asymptotic mean integrated squared error (AMISE) is given by [14], [30]

$$AMISE\{\hat{f}(x)\} = (Nh)^{-1}R(K) + \frac{1}{4}h^4\mu_2(K)^2R\left(f''\right) \tag{3.10}$$

where $\mu_2(K) = \int z^2K(z)\mathrm{d}z$, $R(f'') = \int\{f''(x)\}^2\mathrm{d}x$, $f''(x) = \frac{\mathrm{d}^2}{\mathrm{d}^2x}f(x)$ and $R(K) = \int K(z)^2\mathrm{d}z$. We see that minimizing the left term (the variance) with a large kernel window h results in a huge increase in the bias part which is proportional to h^4. This is what is known as the *variance-bias trade-off* in kernel size selection. There exists many ways to find the kernel size h. We mention two popular methods here. The first differentiates Eq. (3.10) and equates it to zero, obtaining

$$h_{AMISE} = \left[\frac{R(K)}{\mu_2^2(K)R(f'')N}\right]^{\frac{1}{5}}$$

The other method estimates $R(f'')$ by assuming that the true underlying density is a normal density. Then the kernel size is given by [9]

$$h_{AMISE} = 1.06N^{-\frac{1}{5}}$$

Several other methods exist, see [14] and [30].

3.2.4 The multivariate Parzen window

The extension of the Parzen window to feature data in a d-dimensional space is a little more difficult. The sparseness of data in higher dimensional spaces makes the estimation more difficult, unless we have very many samples. This phenomenon is usually referred to as *the curse of dimensionality.* Remembering that the kernel function in the one-dimensional case specify the window width, this window will in the multidimensional case be replaced with hypercubes and each dimension of the cube requires a parameter to be estimated for the kernel. A direct extension of the univariate kernel estimate in Eq. (3.5), is obtained by replacing the point x with a vector-point $\mathbf{x} \in \mathbb{R}^d$ and the variable x_i with a d-variate sample $\mathbf{x}_i$ with density $f(\mathbf{x})$. The Parzen estimator becomes [30]

$$\hat{f}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i) \tag{3.11}$$

where $\mathbf{H}$ is a symmetric positive definite $d \times d$ matrix called the *bandwidth matrix*

$$K_{\mathbf{H}}(\mathbf{x}) = |\mathbf{H}|^{-1/2} K(\mathbf{H}^{-1/2}\mathbf{x}).$$

With further restrictions on $\mathbf{H}$, see [30] for details, we get the single bandwidth kernel estimator

$$\hat{f}(\mathbf{x}) = \frac{1}{Nh^{-d}} \sum_{i=1}^{N} K\{(\mathbf{x} - \mathbf{x}_i)/h\}. \tag{3.12}$$

There exists several methods to give an estimate of the optimal kernel size for a multivariate data set. The optimal kernel size is usually selected to minimize the MISE between $\hat{f}(\mathbf{x})$ and the target density $f(\mathbf{x})$. The normal reference rule for the MISE kernel size is given by Silverman's rule [28], [9]

$$\hat{h} = \sigma_x \left[\frac{4}{(2d+1)N} \right]^{\frac{1}{d+4}}, \tag{3.13}$$

where $\sigma_x^2 = d^{-1} \sum_i \Sigma_{ii}$ and Σ_{ii} are the diagonal elements of the sample covariance matrix. Due to the curse of dimensionality this method is not regarded as reliable for higher dimensional data. In this thesis most of the data sets are of higher dimensions, so we have chosen a cross validation technique to find the best kernel sizes in the density estimates.

Some well-known Parzen windows with $u = \frac{x-x_i}{h}$ are listed below, where for $u \geq 1$ all windows evaluate to zero, except the Gaussian kernel. For simplicity we only present the one-dimensional versions, but they can easily be extended to multivariate versions.

- Uniform
$$K_h(u) = 1/2.$$
- Epanechnikov
$$K_h(u) = \frac{3}{4}(1 - u^2).$$
- Gaussian
$$W_{\sigma^2}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{(x - x_i)^2}{2\sigma^2} \right\}.$$
- Quartic
$$K_h(u) = \frac{15}{16}(1 - u^2)^2.$$
- Triweight
$$K_h(u) = \frac{35}{32}(1 - u^2)^3.$$
- Cosinus
$$K_h(u) = \frac{\pi}{4} \cos\left(\frac{\pi}{4} u\right).$$

Chapter 4

Information theoretic learning principles

This chapter starts with a brief introduction of information theory. An overview of the concepts in an information theoretic learning machine is then given. Information theoretic criteria, which gives us the tools to measure the shape of, and distance between, probability distributions, is then explained in detail. In the last section we discuss some cost function estimators used in ITL.

4.1 Information theory

Information theory is in this thesis related to C.E Shannon's report from 1948, *A mathematical theory of communication* [25]. Shannon defined a *measure of information* or *uncertainty* associated with a stochastic experiment and named it *entropy.* This measure was used to answer important questions in communication. Shannon used entropy to find a limit to how much information can be transferred over a noisy channel, and to find ways to design optimal codes for data compression.

Entropy can be thought of as the uncertainty associated with the value of a realization of a single random variable. It is a measure on how much information that is gained about the content of a stochastic random variable after a stochastic experiment.

Shannon also defined a measure called *mutual information*, which is the amount of information that one random variable carries about another, i.e. the reduction in the uncertainty of one random variable due to the knowledge of the other. Mutual information is a special case of a more general quantity, called *relative entropy.* The *relative entropy* or *divergence* can be used as a measure of "distance" between two distributions. It is a measure of the inefficiency of assuming that a distribution is given by a density function $q(\cdot)$, when it in fact has a distribution given by a density function $p(\cdot)$. This is also refereed to as a *divergence measure* between distributions. In this thesis we use different estimates of entropy and divergence as information theoretic measures.

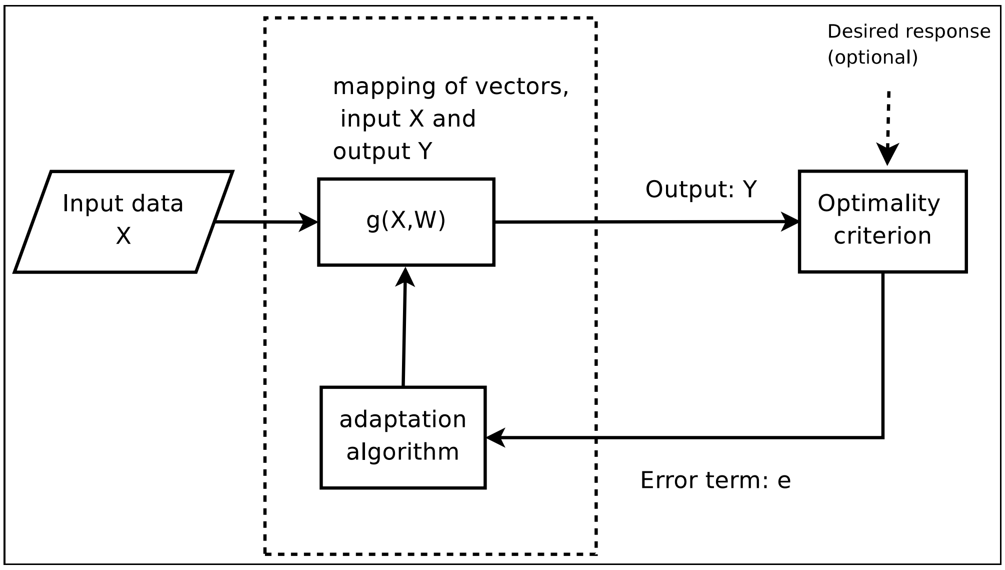

Figure 4.1: A general learning machine.

4.2 An information theoretic learning machine

A very general learning machine may be described using a model like the one in Fig. 4.1. Machine learning is in general divided into *supervised, semi-supervised* and *unsupervised* tasks. The model in Fig. 4.1 can be used to describe all of the above machine learning tasks. We have some input data X, containing information or measurements from a real-world event. We want the learning machine to perform some specific task on X. This is done by giving the input data X to a possibly non-linear parametric mapping function

$$g : \mathbb{R}^d \rightarrow \mathbb{R}^M, \tag{4.1}$$

which transforms the input vector $X \in \mathbb{R}^d$ to $Y \in \mathbb{R}^M$

$$Y = g(X, W), \tag{4.2}$$

where W are the parameters of the mapping function. If the optimality criterion is based on an information theoretic measure, either entropy or divergence, we call this an *information theoretic learning machine*. The mapper function in Eq. (4.2) transform the input data to a new form depending on the task of the learning machine. The output of the mapper, Y is compared with an optimality criterion and optionally a desired response z for the input X. For each presentation of training data the optimality criterion is evaluated and the error term e is fed to an adaptation algorithm which update the parameters W.

Supervised learning concerns a learning machine with a desired response for each input X. This thesis focus on supervised learning, specifically on classification of data. If the learning task is classification, the desired response z for the training data contains a class label. The task is then to classify a random input sample to one of C classes. The mapping function may in this case be described as

$$g : \mathbb{R}^d \to \{y_1, \dots, y_c\},$$

where $y_1, \dots, y_c$ are the possible class labels, and $M = 1$ in Eq. (4.1). *Regression* is another example of a typical supervised learning task, where the mapping is

$$g : \mathbb{R}^d \to \mathbb{R}.$$

Semi-supervised learning tasks concerns learning where the labels of the input data are only partially known. One example is *ranking* where we only have available the relative ordering of the the examples in the training set, while our aim is to enable a similar ordering of novel data.

Unsupervised tasks are learning tasks where the wanted information from the data has to be extracted without any desired response in the optimality criterion. *Clustering* is one typical example of unsupervised learning, where the aim is to find a natural division of data into homogeneous groups [26]. *Anomaly and novelty detection* are other examples, where the task is to detect samples that deviate from the normal. Other important unsupervised tasks are finding low-dimensional representations of the input data, important examples of this is *principal component analysis* (PCA) and *independent component analysis* (ICA). In PCA, the mapping in Eq. (4.1) aims to project the input X to a lower M-dimensional space, where M denotes the number of uncorrelated features in X. In ICA the goal is to project X to a lower M-dimensional space where each of the features of X are mutually independent.

Traditionally the criteria for optimality in Fig. 4.1 has been to minimize the MSE cost function between the output Y of the mapper and the desired output z

$$J(Y) = E\left\{(z - Y)^2\right\}. \tag{4.3}$$

In our general machine learning model we want to transfer as much information as possible about our data into the mapper function $g(X, W)$, such that this mapper is able to describe our data as accurately as possible. The optimality criterion is thus critical in obtaining the parameters W. If we use MSE, the information transferred from the measurements X and the desired responses z to the parameters W is purely based on second order statistics constraints. This is only optimal if the input data is drawn from Gaussian distributions, which is a rather strict restriction.

To transfer as much information as possible to the parameters W the error term in Fig. 4.1 must be computed with a criterion transferring as much information as possible about the input data X, and the desired response z, to the parameters W of the mapper $Y = g(X, W)$. If we base our calculations on information theory and optimize with information theoretic criteria in Fig. 4.1 and Eq. (4.2) we have what Principe et al describes as *information theoretic learning* [18]. The main advantage with information theoretic criteria are that they are functions of probability densities and capture all data statistics, not just the second-order statistics. This gives us learning machines where the parameters W describes our data in a much better way than the traditional MSE can.

4.3 Information theoretic quantities

In this section we discuss some important information theoretic quantities that may be used as information theoretic criteria in an ITL machine.

Definitions A discrete stochastic variable $\mathbf{X}$ is associated with a triple $(\mathbf{x}, \mathcal{A}_{\mathbf{X}}, \mathcal{P}_{\mathbf{X}})$, where the outcome $\mathbf{x}$ is the value of the stochastic variable which takes on a set of possible values $\mathcal{A}_{\mathbf{X}} = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_N\}$. These have probabilities (distribution) $\mathcal{P}_{\mathbf{X}} = \{p_1, p_2, \ldots, p_N\}$, where $P(\mathbf{x} = \mathbf{a}_i) = p_i$, $p_i \geq 0$ and $\sum_{\mathbf{a}_i \in \mathcal{A}_{\mathbf{X}}} P(\mathbf{x} = \mathbf{a}_i) = 1$.

A continuous stochastic variable $\mathbf{X}$ is associated with a probability density function $f_{\mathbf{X}}(\mathbf{x})$, where the outcome $\mathbf{x}$ is the value of the stochastic variable. The pdf is defined as the derivative of the cumulative distribution function (cdf), defined as $P(\mathbf{X} \leq \mathbf{x}_0) = F_{\mathbf{X}}(\mathbf{x}_0)$, where $0 \leq F_{\mathbf{X}}(\mathbf{x}) \leq 1$. Hence, $f_{\mathbf{X}}(\mathbf{x}_0) = \frac{\partial}{\partial \mathbf{x}} F_{\mathbf{X}}(\mathbf{x}) \mid_{\mathbf{x}=\mathbf{x}_0}$, and $\int_{-\infty}^{\infty} f_{\mathbf{X}}(\mathbf{x}) d\mathbf{x} = 1$.

We have statistical independence between random variables $\mathbf{X}_1, \ldots, \mathbf{X}_N$ if and only if $f(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \prod_{i=1}^{N} f(\mathbf{x}_i)$.

A metric on a set $\mathcal{X}$ is a function $u : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$. For all x,y,z in $\mathcal{X}$, this function is required to satisfy the following conditions

1. $u(x, y) \geq 0$.
2. $u(x, y) = 0$ if and only if x=y.
3. $u(x, y) = u(y, x)$.
4. $u(x, z) = u(x, y) + u(y, z)$.

4.3.1 Entropy

All information theoretic criteria are related to the concept of entropy. We now explain Shannon's measure of entropy and some of it's properties. A more general version of entropy, the Renyi entropy is then reviewed.

Shannon's entropy

Assume there is some uncertainty in the outcome of an random experiment and that the possible outcomes of the experiment is given by a probability distribution. This "uncertainty" was first quantified by Shannon as $H = H_N(p_1, p_2, \ldots, p_N)$ satisfying the following criteria [25] [9]

1. $H_N(p_1, p_2, \ldots, p_N)$ is a symmetric function of its variables.
   As an example, $H_N(p_1, p_2, \ldots, p_N) = H_N(p_2, p_1, \ldots, p_N)$.
2. $H_N(p_1, p_2, \ldots, p_N)$ is a continuous function of $p_1, p_2, ..., p_N$.
3. $H_N(\frac{1}{N}, \ldots, \frac{1}{N})$ attains the maximum value.

4. $H_{N+1}(tp_1, (1-t)p_1, p_2, \ldots, p_N) = H_N(p_1, p_2, \ldots, p_N) + p_1 H_2(t, 1-t)$ for any distribution p_X and $0 \leq t \leq 1$.

The fourth property of Shannon entropy may be explained as follows [25]. If a choice is broken down into two successive choices, the original entropy (H) is the weighted sum of the individual values of H [25]. This is illustrated in Fig. 4.2.

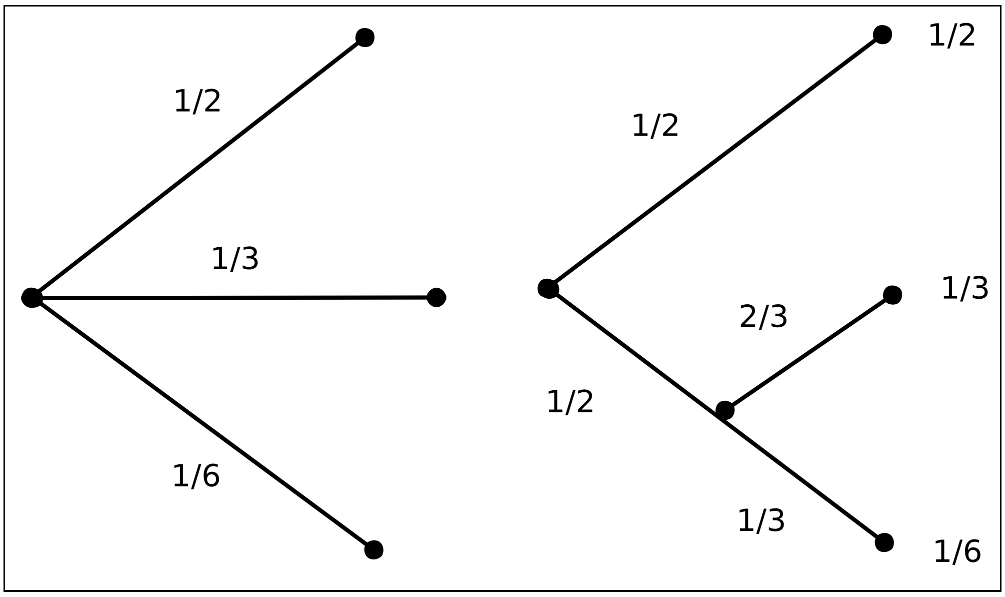

Figure 4.2: At the left we have three possibilities, each chosen according to the probabilities $p_1 = \frac{1}{2}$, $p_2 = \frac{1}{3}$, $p_3 = \frac{1}{6}$. On the right, we first choose between two possibilities each with probability $\frac{1}{2}$. If the second possibility is chosen, we make another choice with probabilities $\frac{2}{3}$ and $\frac{1}{3}$. The final results have the same probabilities as before. We require, in this special case, that $H(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}) = H(\frac{1}{2}, \frac{1}{2}) + \frac{1}{2}H(\frac{2}{3}, \frac{1}{3})$. The $\frac{1}{2}$ coefficient is because this second choice only occurs half the time.

Shannon showed that the only H satisfying the above assumptions is [25]

$$H_N(p_1, p_2, \ldots, p_N) = H_N(\mathcal{P}_{\mathbf{X}}) = -K \sum_{p_i \in \mathcal{P}_{\mathbf{X}}} p_i \log_b p_i, \tag{4.4}$$

with the convention that $0 \log_b 0 = 0$. This measure he called entropy, because it is the same expression used to define entropy in statistical mechanics. K is some constant, depending of the units of the sample data. If the base $b = 2$, the entropy unit is *bits* and if $b = e$ the unit is *nats*. In this thesis we leave the base b of the logarithm unspecified, since it is just a measurement scale. Entropy is usually denoted by $H(\mathbf{X})$ where $\mathbf{X}$ is a label for a random variable, and not the argument of a function. Shannon's entropy depend on the quantity

$$I(p_i) = -\log p_i, \tag{4.5}$$

which was proposed by Hartley as a measure of the information received by learning that a single event of probability p_i took place [7]. Hence the Shannon entropy is a weighted average of informations $I(p_i)$

$$H(\mathbf{X}) = E\{I(p_i)\}. \tag{4.6}$$

Properties of Shannon entropy

Several properties of the Shannon entropy can be derived based on the four basic properties [9] [25] [3]

1. Adding or removing an event with probability zero does not contribute to the entropy, $H_N(p_1, p_2, \ldots, p_N, 0) = H_N(p_1, p_2, \ldots, p_N)$.
2. It vanishes when one outcome is certain, $H_N(p_1, p_2, \ldots, p_N) = 0$, $p_i = 1$, $p_j = 0, j \neq i, i = 1, \ldots, N$.
3. The maximum of H_N increases as N increases.
4. $H_N \geq 0$.

Example To illustrate some properties of Shannon entropy let the stochastic variable X be given by

$$X = \begin{cases} 1, & \text{with probability } p \\ 0, & \text{with probability } 1-p. \end{cases}$$

The entropy in this case is

$$H(X) = -p\log(p) - (1-p)\log(1-p),$$

as shown in Fig. 4.3 as a function of p. Note in Fig. 4.3 that the entropy is zero when $p = 0$ or $p = 1$, meaning there is no uncertainty about the outcome of the stochastic experiment. If $p = \frac{1}{2}$ the uncertainty is maximized, and we need on average 1 bit to transmit the outcome of the experiment.

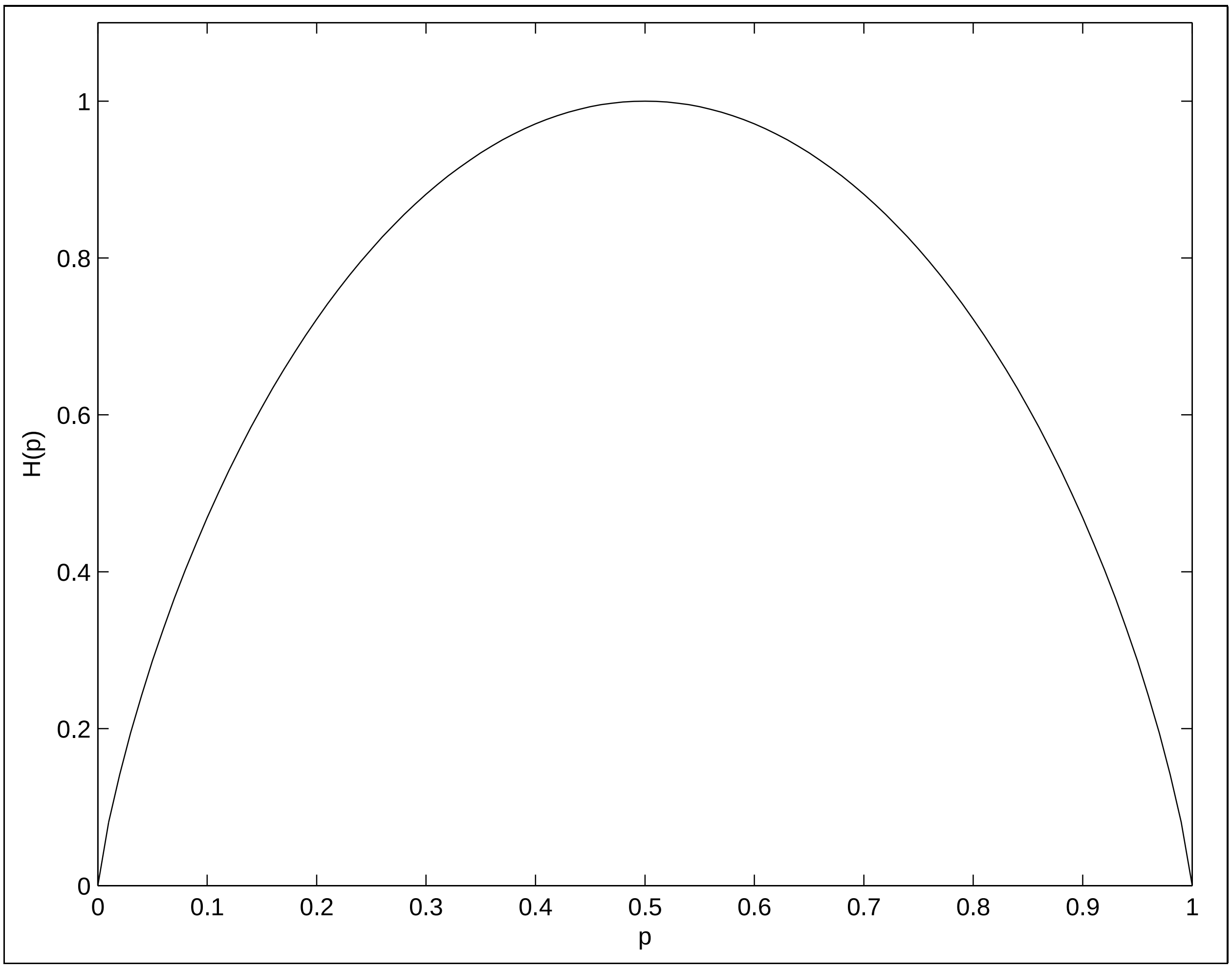

Figure 4.3: $H(X) = H(p)$ in bits with Shannon's entropy, notice that $H(X) = 1$, when $p = \frac{1}{2}$.

Shannon differential entropy

For a continuous stochastic variable $\mathbf{X}$ with density $f(\mathbf{x})$[1] the differential entropy $h(\mathbf{X})$ is defined as [25] [3],

$$h(\mathbf{X}) = -\int f(\mathbf{x}) \log f(\mathbf{x}) \mathrm{d}\mathbf{x}. \tag{4.7}$$

This can also be written as an expected value

$$h(\mathbf{X}) = E_f\{-\log f(\mathbf{x})\}. \tag{4.8}$$

Properties of Shannon's differential entropy

1. If $\mathbf{X}$ is limited to a certain volume v in space, then $h(\mathbf{X})$ is maximum and equal to $\log v$ when $f(\mathbf{x})$ is constant (uniform density function) in the volume.

2. Differential entropy may be negative. If we consider the uniform density above, for $v < 1$, $\log v < 0$.

3. The normal distribution maximizes the entropy over all distributions with the same covariance. This property can be exploited to measure the non-Gaussianity of a stochastic variable.

4. The differential entropy is a measure that is relative to the coordinate system. Consider for example changing coordinates by a linear transformation, $\mathbf{Y} = \mathbf{MX}$. In that case,
$$h(\mathbf{Y}) = h(\mathbf{X}) + \log|\det(\mathbf{M})|.$$

Renyi's entropy

As explained above, Shannon's entropy is a measure of the *average* amount of information contained in a single observation of a random variable. Renyi used a more general theory of mean values, where the mean of the real numbers, $x_1, \ldots, x_N$, with positive weighting $p_1, \ldots, p_N$, has the form [18] [22]

$$\overline{x} = \varphi^{-1} \sum_{i=1}^{N} p_i \varphi(x_i), \tag{4.9}$$

where $\varphi(x)$ is a Kolmogorov-Nagumo function, which is an arbitrary continuous and strictly monotonic function defined on the real numbers. He found that a general entropy measure H obeys the relation [18]

$$H = \varphi^{-1}\left(\sum_{i=1}^{N} p_i \varphi(I(p_i))\right), \tag{4.10}$$

[1] We assume that the stochastic variable has a density function where the integral does exist.

where $I(p_i)$ is Hartley's information measure. In order to be an information measure $\varphi(\cdot)$ cannot be arbitrary, since information is additive. We have two choices, $\varphi(x) = x$ or $\varphi(x) = 2^{(1-\alpha)x}$. The first case gives Shannon's entropy and the second gives *Renyi's entropy of order* α [18]

$$H_{R_\alpha}(\mathbf{X}) = \frac{1}{1-\alpha}\log\left(\sum_{k=1}^{N} p_k^\alpha\right) \qquad \alpha > 0, \alpha \neq 1. \tag{4.11}$$

There is a well known relation between Shannon's and Renyi's entropy. Let H_S denote Shannon's entropy, then [18]

$$H_{R_\alpha} \geq H_s \geq H_{R_\beta} \qquad \text{if} \qquad 0 < \alpha < 1 \text{ and } \beta > 1,$$

$$\lim_{\alpha \to 1} H_{R_\alpha} = H_S.$$

Renyi's and Shannon's entropies can also be related to each other in another way. If we consider the probability mass function $P = (p_1, p_2, \ldots, p_N)$ as a *point in the N-dimensional space*, this point will always be in the first quadrant of a N-dimensional hyperplane with each axis intersecting the coordinate one. The distance of the point P to the origin is the α root of

$$d_\alpha = \sum_{k=1}^{N} p_k^\alpha = ||P||^\alpha$$

and the α root of d_α is called the α-norm of the probability mass function [18]. Renyi's entropy satisfies all of Shannon's criteria in 4.3.1 on page 23. Except of the fourth property. The Renyi's entropy of order $\alpha = 2$, is denoted by *Renyi's quadratic entropy* and corresponds to the 2-norm of the probability mass function.

If we repeat the example on page 24 with Renyi's quadratic entropy measure, we get a similar shape in Fig. 4.4, as in Shannon's entropy in Fig. 4.3 on page 25.

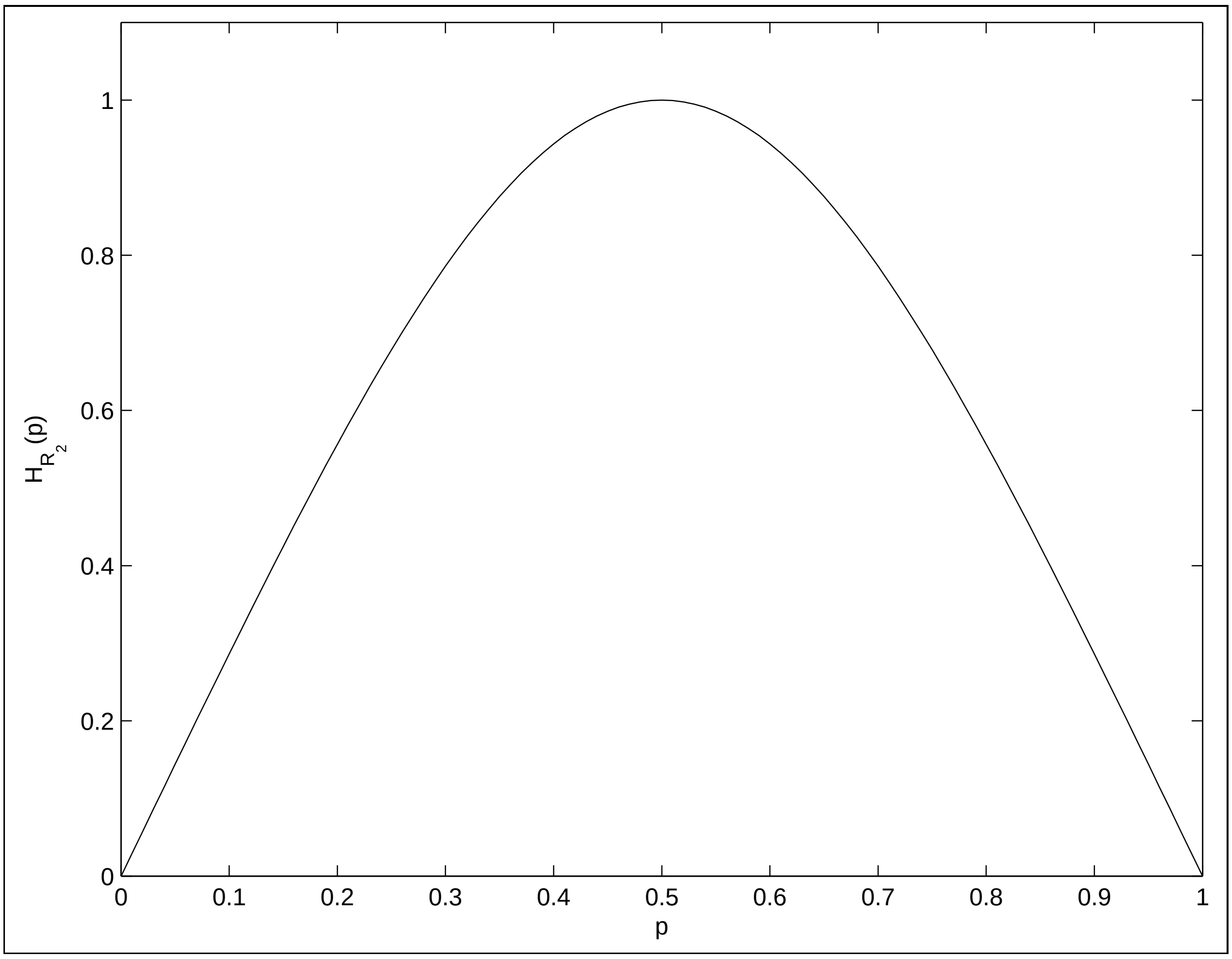

Figure 4.4: $H(X) = H(p)$ in bits with Renyi's quadratic entropy measure, notice that $H(X) = 1$ when $p = \frac{1}{2}$.

Renyi's differential entropy

For the continuous random variable $\mathbf{X}$ with pdf $f(\mathbf{x})$ we obtain the differential version of Renyi's entropy [18]

$$\begin{aligned} h_{R_\alpha}(\mathbf{X}) &= \frac{1}{1-\alpha}\log\int f^\alpha(\mathbf{x})\mathrm{d}\mathbf{x}, && (4.12)\\ &= \frac{1}{1-\alpha}\log E_f\{f^{1-\alpha}(\mathbf{x})\}. && (4.13)\end{aligned}$$

If we set $\alpha = 2$, we get the differential Renyi quadratic entropy

$$\begin{aligned} h_{R_2}(\mathbf{X}) &= -\log\int f^2(\mathbf{x})\mathrm{d}\mathbf{x}, && (4.14)\\ &= -\log E_f\{f(\mathbf{x})\}. && (4.15)\end{aligned}$$

Some properties of the Renyi entropy of order α are the following [9]

1. Just as for Shannon entropy, the Renyi entropy is maximized for a uniform distribution for random variables with finite support.
2. The Renyi entropy is not in general maximized by the Gaussian distribution in the fixed variance case.
3. The Renyi entropy is invariant to rotations and translations.

4.3.2 Divergence

This section reviews some of the most common measures of divergence or relative entropy used in information theoretic learning. Divergence is used as a measure of statistical similarity, and one can think of it as a generalization of algebraic distance measures to probability spaces [5].

Kullback-Leibler divergence

This measure discriminates two probability density distributions $p(\mathbf{x})$ and $q(\mathbf{x})$, and it is also referred to as relative entropy

$$\begin{aligned} D_{KL}\{p,q\} &= \int p(\mathbf{x})\log\frac{p(\mathbf{x})}{q(\mathbf{x})}\mathrm{d}\mathbf{x},\\ &= E_p\left\{\log\frac{p(\mathbf{x})}{q(\mathbf{x})}\right\}. && (4.16)\end{aligned}$$

Some properties of this measure are [9]

1. $D_{KL}\{p,q\} \geq 0, \forall p,q$.
2. $D_{KL}\{p,q\} = 0$ only if $p(\mathbf{x}) = q(\mathbf{x})$.
3. $D_{KL}\{p,q\}$ is additive for independent random events.

This measure is not a metric, since it is not symmetric i.e $D_{KL}\{p,q\} \neq D_{KL}\{q,p\}$, and it does not satisfy the triangle inequality. The divergence measure is invariant under the following changes in $\mathbf{x}$ [9]

1. Permutation in the order of which the components are arranged.
2. Amplitude scaling.
3. Monotonic nonlinear transformation.

The Kullback-Leibler divergence is implicitly based on Shannon's entropy, since

$$D_{KL}\{p,q\} = -\int p(\mathbf{x}) \log q(\mathbf{x}) \mathrm{d}\mathbf{x} - \left(-\int p(\mathbf{x}) \log p(\mathbf{x}) \mathrm{d}\mathbf{x} \right), \tag{4.17}$$

where the last part is Shannon's differential entropy and the first part can be interpreted as "cross entropy" between $p(\mathbf{x})$ and $q(\mathbf{x})$ [9].

Mutual Information

The mutual information $MI(\mathbf{X};\mathbf{Y})$ between two random variables $\mathbf{X}$ and $\mathbf{Y}$ with joint density $f(\mathbf{x},\mathbf{y})$ is defined as [3]

$$MI(\mathbf{X};\mathbf{Y}) = \int f(\mathbf{x},\mathbf{y}) \log \frac{f(\mathbf{x},\mathbf{y})}{f(\mathbf{x})f(\mathbf{y})} \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{y}. \tag{4.18}$$

From this definition [3]

$$MI(\mathbf{X};\mathbf{Y}) = h(\mathbf{X}) - h(\mathbf{X}|\mathbf{Y}) = h(\mathbf{Y}) - h(\mathbf{Y}|\mathbf{X}). \tag{4.19}$$

Mutual information is a special case of Kullback-Leibler divergence, measuring the distance between the joint probability distribution and the product of the marginal distributions.

Renyi's divergence

Renyi analyzed the Kullback-Leibler divergence and expressed it with a general mean value, in a similar way as entropy. Renyi proposed the following distance measure between pdfs $p(\mathbf{x})$ and $q(\mathbf{x})$ [22]

$$\begin{aligned} D_{R_\alpha}\{p,q\} &= \frac{1}{1-\alpha} \log \int \frac{p^\alpha(\mathbf{x})}{q^{\alpha-1}(\mathbf{x})} \mathrm{d}\mathbf{x}, \\ &= \frac{1}{1-\alpha} \log E_p \left\{ \frac{p^\alpha(\mathbf{x})}{q^{\alpha-1}(\mathbf{x})} \right\}. \end{aligned} \tag{4.20}$$

The Renyi divergence possesses the following properties [22]

1. $D_{R_\alpha}\{p,q\} \geq 0,\ \forall p,q,\ \alpha > 0.$
2. $D_{R_\alpha}\{p,q\} = 0,$ if and only if $p(\mathbf{x}) = q(\mathbf{x}), \forall \mathbf{x} \in \mathbb{R}^d$.
3. $\lim_{\alpha\to 1} D_{R_\alpha}\{p,q\} = D_{KL}\{p,q\}$.
4. $D_{R_\alpha}\{p,q\}$ is additive for independent events.

We see that Renyi divergence is not symmetric and hence not a metric. We can use Renyi divergence to measure the mutual information between random variables, measuring the distance between the joint pdf and the product of marginal densities.

Cauchy-Schwarz divergence

Principe et al.[18], defined a pdf divergence measure based on the Cauchy-Schwarz (CS) inequality. Let $p(\mathbf{x})$ and $q(\mathbf{x})$ be pdf functions, i.e non-negative and integrating to unity. Define the inner product between two square integrable functions $p(\mathbf{x})$ and $q(\mathbf{x})$ as $\langle p,q\rangle = \int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x}$. Then by the Cauchy-Schwarz inequality, and the fact that $p(\mathbf{x})$ and $q(\mathbf{x})$ are always non-negative

$$\langle p,q\rangle^2 \leq \langle p,p\rangle \cdot \langle q,q\rangle,$$

with equality if and only if the two functions are linearly dependent. The Cauchy-Schwarz pdf divergence is defined [13],[12] as

$$\begin{aligned} D_{CS}\{p,q\} &= -\log\left\{\frac{\langle p,q\rangle}{\sqrt{\langle p,q\rangle\langle q,q\rangle}}\right\} \\ &= -\log\left\{\frac{E_p\{g(\mathbf{x})\}}{\sqrt{E_p\{p(\mathbf{x})\}E_q\{q(\mathbf{x})\}}}\right\}. \end{aligned} \tag{4.21}$$

Some properties [9] of the Cauchy-Schwarz divergence are

1. $D_{CS}\{p,q\} \geq 0,\ \forall p,q.$
2. $D_{CS}\{p,q\} = 0,$ if and only if $p(\mathbf{x}) = q(\mathbf{x}), \forall\, \mathbf{x} \in \mathbb{R}^d$.
3. $D_{CS}\{p,q\} = D_{CS}\{q,p\}$.
4. $D_{CS}\{p,q\}$ is additive for independent events.

The CS divergence does not satisfy the triangle inequality, and for this reason it is not a distance metric [9]. CS divergence can be used as a measure of statistical independence between random variables. CS divergence is implicitly based on Renyi's quadratic entropy, since

$$D_{CS}\{p,q\} = \\ -\log\int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x} - \frac{1}{2}\left(-\log\int p^2(\mathbf{x})\mathrm{d}\mathbf{x}\right) - \frac{1}{2}\left(-\log\int q^2(\mathbf{x})\mathrm{d}\mathbf{x}\right), \tag{4.22}$$

where $-\log \int p^2(\mathbf{x})\mathrm{d}\mathbf{x}$ is the Renyi quadratic entropy with respect to $p(\mathbf{x})$ and $-\log \int q^2(\mathbf{x})\mathrm{d}\mathbf{x}$ is the entropy with respect to $q(\mathbf{x})$. The first term may be regarded as a "cross-entropy" between $p(\mathbf{x})$ and $q(\mathbf{x})$ [9].

Integrated squared error

Principe et al. in [18] also proposed an integrated squared error (ISE) distance measure between the two pdfs, $p(\mathbf{x})$ and $q(\mathbf{x})$. As before, the inner product between two square integrable functions $p(\mathbf{x})$ and $q(\mathbf{x})$ is $\langle p, q\rangle = \int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x}$. The ISE divergence measure is defined as

$$\begin{aligned} D_{ISE}\{p,q\} &= \int [p(\mathbf{x}) - q(\mathbf{x})]^2 \mathrm{d}\mathbf{x}, \\ &= \int p^2(\mathbf{x})\mathrm{d}\mathbf{x} - 2\int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x} + \int q^2(\mathbf{x})\mathrm{d}\mathbf{x}, \\ &= E_p\{p(\mathbf{x})\} - 2E_p\{q(\mathbf{x})\} + E_q\{q(\mathbf{x})\} \\ &= \langle p,p\rangle - 2\langle p,q\rangle + \langle q,q\rangle \end{aligned} \tag{4.23}$$

This measure is obviously zero if the two pdfs are equal, it is always non-negative and symmetric. It does not satisfy the additive property, so we must be careful when calling it an information theoretic measure [9].

Other divergence measures

There exists many other important distance measures between pdfs. A well known example is the Csiszar divergence. For an arbitrary convex function $h(\cdot)$ such that $h(1) = 0$ we define [5]

$$D_h\{p,q\} = \int p(\mathbf{x}) h\left(\frac{q(\mathbf{x})}{p(\mathbf{x})}\right) \mathrm{d}\mathbf{x}. \tag{4.24}$$

Some other measures are the Jeffrey's distance, which is a symmetric version of the Kullback-Leibler distance, and Chernoff distances [9]. Common to most of the measures are that they are not metrics, but they can give us useful information about divergence between pdfs.

4.4 Estimation of information theoretic cost functions

In Section 4.2 we explained how a learning machine depends on an optimality criterion to calculate error terms to be used in an adaptation algorithm. In information theoretic learning this criterion is based on entropy or divergence of probability density functions. The output of the cost function or optimality criterion in Fig. 4.1 is typically used to update the parameters W of the mapper $g(X, W)$ during a training phase. We have already seen in Chapter 3 that a Parzen window density estimator utilizes a kernel function to give an estimate of a pdf. In this section we review some techniques to find estimates of information measures using the Parzen window technique for density estimation. Since the Parzen window density estimate can be continuous, we also want our cost functions to be continuous. Thus we focus on estimating differential information theoretic measures.

4.4.1 Renyi quadratic entropy estimate

The differential Renyi quadratic entropy associated with the pdf $f(\mathbf{x})$ is given by [22]

$$h_{R_2}(\mathbf{X}) = -\log \int f^2(\mathbf{x})\mathrm{d}\mathbf{x} \tag{4.25}$$

Since the Renyi quadratic entropy contains a product of densities, we take advantage of the convolution property of Gaussians, and use the Gaussian kernel $W_{\sigma^2}(\cdot,\cdot)$ in the *plug in* density estimate. Since the logarithm is a monotonic function, we focus on the quantity $V(f) = \int \hat{f}^2(\mathbf{x})\mathrm{d}\mathbf{x}$, given by[2]

$$\begin{aligned} V(f) =& \int \frac{1}{N}\sum_{i=1}^{N} W_{\sigma^2}(\mathbf{x},\mathbf{x}_i)\frac{1}{N}\sum_{i'=1}^{N} W_{\sigma^2}(\mathbf{x},\mathbf{x}_{i'})\mathrm{d}\mathbf{x} \\ =& \frac{1}{N^2}\int \sum_{i,i'=1}^{N,N} W_{\sigma^2}(\mathbf{x},\mathbf{x}_i)W_{\sigma^2}(\mathbf{x},\mathbf{x}_{i'})\mathrm{d}\mathbf{x}. \end{aligned} \tag{4.26}$$

We now use the convolution theorem for Gaussians

$$\int W_{\sigma^2}(\mathbf{x},\mathbf{x}_i)W_{\sigma^2}(\mathbf{x},\mathbf{x}_{i'})\mathrm{d}\mathbf{x} = W_{2\sigma^2}(\mathbf{x}_i,\mathbf{x}_{i'}). \tag{4.27}$$

Inserting this into Eq. (4.26) gives

$$V(f) = \frac{1}{N^2}\sum_{i,i'=1}^{N,N} W_{2\sigma^2}(\mathbf{x}_i,\mathbf{x}_{i'}). \tag{4.28}$$

It can be seen that this sample based estimator involves no approximations, except the pdf estimate itself. This is an advantage compared to the Shannon entropy estimate, which has made the Renyi entropy the preferred estimator over Shannon's. The expression in Eq. (4.28) is named the *information potential* (IP) by Principe et al. [18] due to an analogy with a potential field. The Renyi quadratic entropy estimator is thus

$$\hat{h}_{R_2}(\mathbf{X}) = -\log\{V(f)\}. \tag{4.29}$$

We can also estimate the Renyi entropy of higher orders and obtain more information about the structure of the data set ($\alpha > 2$), but the algorithm becomes much more complex ($O(N^\alpha)$).

[2] $\sum_{i,i'=1}^{N,N}$ equals the double summation $\sum_{i=1}^{N}\sum_{i'=1}^{N}$

4.4.2 ISE divergence estimate

In the Renyi's quadratic entropy estimate, the use of a quadratic measure and a Gaussian Parzen window, resulted in an estimate with no other approximations than the density estimate. In a similar manner we can estimate the $ISE\{p, q\}$ between two pdfs $p(\mathbf{x})$ and $q(\mathbf{x})$ given by [18]

$$\begin{aligned} ISE\{p,q\} &= \int [p(\mathbf{x}) - q(\mathbf{x})]^2 \,\mathrm{d}\mathbf{x} \\ &= \int p^2(\mathbf{x})\mathrm{d}\mathbf{x} - 2\int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x} + \int q^2(\mathbf{x})\mathrm{d}\mathbf{x}. \end{aligned} \tag{4.30}$$

For an overview of some properties of this measure, see Section 4.3.2 on page 32. Assume we have samples $\mathbf{x}_i,\ i = 1, \ldots, N_1$ and $\mathbf{x}_j,\ j = 1, \ldots, N_2$ from $p(\mathbf{x})$ and $q(\mathbf{x})$, respectively. Estimating the two pdfs with the Parzen window method gives

$$\hat{p}(\mathbf{x}) = \frac{1}{N_1}\sum_{i=1}^{N_1} W_{\sigma^2}(\mathbf{x}, \mathbf{x}_i) \quad \hat{q}(\mathbf{x}) = \frac{1}{N_2}\sum_{j=1}^{N_2} W_{\sigma^2}(\mathbf{x}, \mathbf{x}_j). \tag{4.31}$$

Plugging this into Eq. (4.30) we get the ISE sample based estimator

$$\begin{gathered} \widehat{ISE}\{p,q\} = \\ \frac{1}{N_1^2}\sum_{i,i'=1}^{N_1,N_1} W_{2\sigma^2}(\mathbf{x}_i, \mathbf{x}_{i'}) - \frac{2}{N_1 N_2}\sum_{i,j=1}^{N_1,N_2} W_{2\sigma^2}(\mathbf{x}_i, \mathbf{x}_j) + \frac{1}{N_2^2}\sum_{j,j'=1}^{N_2,N_2} W_{2\sigma^2}(\mathbf{x}_j, \mathbf{x}_{j'}). \end{gathered} \tag{4.32}$$

4.4.3 Cauchy-Schwarz divergence estimate

The Cauchy-Schwarz divergence is given by Eq. (4.21) as

$$\begin{aligned} D_{CS}\{p,q\} &= -\log\left\{\frac{\langle p,q\rangle}{\sqrt{\langle p,q\rangle\langle q,q\rangle}}\right\} \\ &= -\log\left\{\frac{\int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x}}{\sqrt{\int p(\mathbf{x})\mathrm{d}\mathbf{x}\int q(\mathbf{x})\mathrm{d}\mathbf{x}}}\right\} \end{aligned} \tag{4.33}$$

Using the same plug in technique as for the ISE divergence estimator in Eq. (4.4.2) we can express a sample based estimator for the CS divergence as

$$\widehat{D_{CS}}\{p,q\} = -\log\left\{\frac{\frac{1}{N_1N_2}\sum_{i,j=1}^{N_1,N_2} W_{2\sigma^2}(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{\frac{1}{N_1^2}\sum_{i,i'=1}^{N_1,N_1} W_{2\sigma^2}(\mathbf{x}_i, \mathbf{x}_{i'})\frac{1}{N_2^2}\sum_{j,j'=1}^{N_2,N_2} W_{2\sigma^2}(\mathbf{x}_j, \mathbf{x}_{j'})}}\right\} \tag{4.34}$$

4.4.4 Using non-Gaussian kernels in estimation of cost functions

The use of Gaussian kernels has the advantage that the only approximation in the estimate of the cost function will be in the estimation of the pdf. In general, we search for cost functions involving products of densities, because of the properties of the Gaussian kernel. This is the main reason why quadratic measures are preferred in ITL.

Generally, all the differential versions of information theoretic quantities defined earlier can be estimated by expressing the quantities in terms of an expectation value. This may require many samples to be an accurate estimate. We now examine the Parzen window-based estimator of the inner-product $\int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x}$, since this inner-product is common in all the previously defined cost functions. Note that

$$\int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x} = E_p\{q(\mathbf{x})\}, \tag{4.35}$$

where $E_p\{\cdot\}$ denotes the expectation with respect to the density $p(\mathbf{x})$. The expectation operator may be *approximated* based on the available samples in the following way

$$E_p\{q(\mathbf{x})\} \approx \frac{1}{N_1}\sum_{i=1}^{N_1} q(\mathbf{x}_i). \tag{4.36}$$

Assume now that

$$\hat{q}(\mathbf{x}) = \frac{1}{N_2}\sum_{j=1}^{N_2} K_h(\mathbf{x}, \mathbf{x}_j), \tag{4.37}$$

where $K_h(\cdot,\cdot)$ is a non-Gaussian kernel with bandwidth h. Eq. (4.35) can now be estimated by

$$\begin{aligned}\int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x} &\approx \frac{1}{N_1}\sum_{i=1}^{N_1} \widehat{q}(\mathbf{x}_i)\\ &= \frac{1}{N_1}\sum_{i=1}^{N_1}\frac{1}{N_2}\sum_{j=1}^{N_2} K_h(\mathbf{x}_i, \mathbf{x}_j)\\ &= \frac{1}{N_1 N_2}\sum_{i,j=1}^{N_1,N_2} K_h(\mathbf{x}_i, \mathbf{x}_j)\end{aligned} \tag{4.38}$$

This is the same result as in the case where Gaussian kernels are used. The only difference is an additional approximation with regard to the expectation operator.

4.4.5 Shannon entropy estimate

We illustrate how the Shannon entropy of a pdf $f(\mathbf{x})$ may estimated using non-Gaussian kernels. The Shannon entropy may be written as an expected value

$$h(\mathbf{X}) = E_f\{-\log f(\mathbf{x})\}.$$

We have samples $\{\mathbf{x}_i\}$, $i = 1, \dots, N$ from $f(\mathbf{x})$. With the Parzen window estimator we can estimate $f(\mathbf{x})$ by $\hat{f}(\mathbf{x})$. Using the plug-in density estimator principle, we replace $f(\mathbf{x})$ with $\hat{f}(\mathbf{x})$. We now approximate the expected value by averaging over all of the samples, which gives the following estimate for the Shannon entropy of $f(\mathbf{x})$

$$\begin{aligned}\hat{h}(\mathbf{X}) =& \frac{1}{N}\sum_{i=1}^{N}\left\{-\log \hat{f}(\mathbf{x}_i)\right\}, \\ =& \frac{1}{N}\sum_{i=1}^{N}\left\{-\log \frac{1}{N}\sum_{j=1}^{N} K_h(\mathbf{x}_i, \mathbf{x}_j)\right\}. \qquad (4.39)\end{aligned}$$

The drawback of this entropy estimate is the dependency on the approximation of the expected value. With few samples of training data this may not be a good estimator for entropy.

Chapter 5

An information theoretic kernel classifier

Recently some very interesting relations between the information theoretic measures defined in the previous chapter estimated with Parzen windows satisfying Mercer's theorem, and mean vectors in *Mercer kernel spaces* has been shown. In this chapter we review this relationship, and use it to analyze a possible classifier in both the input space and the Mercer kernel space. This classifier operates implicitly in a Mercer kernel space, and we refer to it as the *standard ISE classifier.* We also mention a connection between the ISE divergence and the *graph cut.* In Part II, Section 9.3 of this thesis we present some results using this classifier on some well known data sets.

5.1 Mercer kernel theory

Mercer kernel-based learning algorithms make use of the following idea. Via a nonlinear mapping [9] [17]

$$\begin{aligned}\Phi : \mathbb{R}^d &\longrightarrow \mathcal{F}\\ \mathbf{x} &\longmapsto \Phi(\mathbf{x})\end{aligned}$$

the input data $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^d$ is mapped into a potentially much higher dimensional feature space $\mathcal{F}$. For a given learning problem one now considers the same algorithm in $\mathcal{F}$ instead of in the input space $\mathbb{R}^d$, working with [17]

$$(\Phi(\mathbf{x}_1), y_1), \ldots, (\Phi(\mathbf{x}_N), y_N) \in \mathcal{F} \times Y.$$

The learning algorithm used is usually linear, and can be expressed solely in terms of inner-product evaluations. If we use the *kernel-trick* we can calculate the inner-products in the feature space using *kernel functions.* Using kernel functions we *implicitly* execute the learning algorithm in the feature space $\mathcal{F}$. The kernel trick thus allows us to calculate inner-products in a possible very high-dimensional space.

Consider a symmetric kernel function $\rho(\mathbf{x}, \mathbf{y})$. If $\rho : d \times d \rightarrow \mathbb{R}$ is a continuous kernel of a positive integral in a Hilbert space $L_2(d)$ on a compact set $d \subset \mathbb{R}^N$, i.e

$$\forall \Psi \in L_2 : \int_d \rho(\mathbf{x}, \mathbf{y}) \Psi(\mathbf{x}) \Psi(\mathbf{y}) \mathrm{d}\mathbf{x} \mathrm{d}\mathbf{y} \geq 0. \tag{5.1}$$

This means that $\rho(\cdot, \cdot)$ is a positive semidefinite function. Then there exist a space $\mathcal{F}$ and a mapping $\Phi : \mathbb{R}^d \rightarrow \mathcal{F}$, such that by Mercer's theorem [13]

$$\rho(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle = \sum_{i=1}^{N_{\mathcal{F}}} \lambda_i \phi_i(\mathbf{x}) \phi_i(\mathbf{y}), \tag{5.2}$$

where $\langle \cdot \rangle$ denotes an inner-product, the ϕ_i's are the eigenfunctions of the kernel, the λ_i's are the corresponding eigenvalues and the dimension of the feature space $\mathcal{F}$ is $N_{\mathcal{F}} \leq \infty$ [13]. The operation in Eq. (5.2) is the "kernel trick". A kernel function that satisfies Eq. (5.1) is known as a *Mercer kernel function.* The most widely used Mercer kernel function is the radial-basis-function (RBF) [13]

$$\rho(\mathbf{x}, \mathbf{y}) = \exp \left\{ -\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2} \right\}, \tag{5.3}$$

where σ is a scale parameter to select the *bandwidth* or *width* of the RBF.

Cover showed in [2] that the probability that classes are linearly separable increases when the features are nonlinearly mapped to a higher dimensional feature space. Using the kernel trick we are able to work implicitly in very high dimensional spaces. It has been shown [19] that the Gaussian kernel has an infinite dimensional feature space, thus given any labeled data set (where points with different labels have different positions), there exists a linear hyperplane which correctly separates them in the Mercer space given by the Gaussian kernel.

The support vector machine [17] is one of the most popular Mercer kernel based learning algorithms taking advantage of the kernel trick. The basic idea behind it is to find the hyperplane between two classes which maximizes the margin between the points closest to the hyperplane. The vectors from this hyperplane to the closest points constitute the *support vectors.* If the classes are non-separable in the input space, this hyperplane and the points are calculated in a high dimensional Mercer kernel space, using the kernel trick.

5.2 Information measures in the Mercer kernel space

We will now review how some of the information theoretic measures can be expressed in the term of *mean values* in a Mercer kernel feature space. The key point to expressing ITL criteria in a Mercer kernel space is to note that for any positive semi-definite kernel function $K_h(\cdot, \cdot)$ that satisfies Mercer's theorem

$$K_h(\mathbf{x}_i, \mathbf{x}_{i'}) = \rho(\mathbf{x}_i, \mathbf{x}_{i'}) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_{i'}) \rangle. \tag{5.4}$$

The Gaussian kernel is a kernel that satisfies Mercer's theorem. We now see how the information potential defined in Eq. (4.28) on page 33 may be expressed in a Mercer kernel feature space. Using $W_{\sigma^2}(\cdot,\cdot)$ as a Mercer kernel, the IP was expressed in Section 4.4.1 as

$$V(f) = \frac{1}{N^2}\sum_{i,i'=1}^{N,N} W_{2\sigma^2}(\mathbf{x}_i,\mathbf{x}_{i'}).$$

We can now use the kernel trick to express the IP as

$$\begin{aligned} V(f) &= \frac{1}{N^2}\sum_{i,i'=1}^{N,N} \langle \Phi(\mathbf{x}_i)\Phi(\mathbf{x}_{i'})\rangle, \\ &= \left\langle \frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x}_i)\frac{1}{N}\sum_{i'=1}^{N}\Phi(\mathbf{x}_{i'})\right\rangle \\ &= \mathbf{m}^T\mathbf{m} \\ &= \|\mathbf{m}\|^2, \end{aligned} \tag{5.5}$$

where $\mathbf{m}$ is the mean of the Φ-transformed data

$$\mathbf{m} = \frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x}_i). \tag{5.6}$$

It turns out that the IP for a data set may be expressed as the squared norm of a *mean vector* of the same data set mapped ta a Mercer kernel feature space. The Renyi quadratic entropy estimate of any pdf may thus be visualized with a simple geometric description, as a mean vector in a Mercer kernel feature space.

5.2.1 ISE divergence

We will now see how the ISE divergence can be expressed in a Mercer kernel feature space. The $\widehat{ISE}\{p,q\}$ estimate

$$\int [\widehat{p}(\mathbf{x}) - \widehat{q}(\mathbf{x})]^2 \mathrm{d}\mathbf{x}$$

between two pdfs $p(\mathbf{x})$ and $q(\mathbf{x})$, was estimated in Eq. (4.30) on page 34 as

$$\widehat{ISE}\{p,q\} =$$
$$\frac{1}{N_1^2}\sum_{i,i'=1}^{N_1,N_1} W_{2\sigma^2}(\mathbf{x}_i,\mathbf{x}_{i'}) - \frac{2}{N_1N_2}\sum_{i,j=1}^{N_1,N_2} W_{2\sigma^2}(\mathbf{x}_i,\mathbf{x}_j) + \frac{1}{N_2^2}\sum_{j,j'=1}^{N_2,N_2} W_{2\sigma^2}(\mathbf{x}_j,\mathbf{x}_{j'}).$$

Similar to the calculations in Eq. (5.5) this may be expressed as

$$\begin{aligned} \widehat{ISE}\{f,g\} &= \|\mathbf{m}_1\|^2 - 2\mathbf{m}_1^T\mathbf{m}_2 + \|\mathbf{m}_2\|^2 \\ &= \|\mathbf{m}_1 - \mathbf{m}_2\|^2, \end{aligned} \tag{5.7}$$

where $\mathbf{m}_1$ and $\mathbf{m}_2$ are the mean vectors in the Mercer kernel feature space for data points drawn from $p(\mathbf{x})$ and $q(\mathbf{x})$, respectively. That is

$$\mathbf{m}_1 = \frac{1}{N_1}\sum_{i=1}^{N_1}\Phi(\mathbf{x}_i) \quad \mathbf{m}_2 = \frac{1}{N_2}\sum_{j=1}^{N_2}\Phi(\mathbf{x}_j). \tag{5.8}$$

From this we see that the ISE divergence measure has a nice geometric interpretation in the Mercer kernel feature space. It measures the the squared Euclidean distance between the information potentials of the distributions given by $p(\mathbf{x})$ and $q(\mathbf{x})$. The ISE divergence in a Mercer kernel feature space is illustrated in Fig. 5.1 on page 41, where $\mathbf{w}$ is the vector $\mathbf{m}_1 - \mathbf{m}_2$.

5.2.2 Cauchy-Schwarz divergence

For completeness, we also review an interpretation of the CS-divergence measure in a Mercer kernel feature space. The $\widehat{D_{CS}}\{p,q\}$ estimate was given in Eq. (4.34) on page 34 as

$$\widehat{D_{CS}}\{p,q\} = -\log\left\{\frac{\frac{1}{N_1N_2}\sum_{i,j=1}^{N_1,N_2}W_{2\sigma^2}(\mathbf{x}_i,\mathbf{x}_j)}{\sqrt{\frac{1}{N_1^2}\sum_{i,i'=1}^{N_1,N_1}W_{2\sigma^2}(\mathbf{x}_i,\mathbf{x}_{i'})\frac{1}{N_2^2}\sum_{j,j'=1}^{N_2,N_2}W_{2\sigma^2}(\mathbf{x}_j,\mathbf{x}_{j'})}}\right\} \tag{5.9}$$

between two pdfs $p(\mathbf{x})$ and $q(\mathbf{x})$. Again we use the kernel trick in the same way as in Eq. (5.5) and in Eq. (5.7) and can rewrite Eq. (5.9) to

$$\begin{aligned}\widehat{D_{CS}}\{p,q\} &= -\log\left\{\frac{\mathbf{m}_1^T\mathbf{m}_2}{\sqrt{\|\mathbf{m}_1\|^2\|\mathbf{m}_2\|^2}}\right\}\\ &= -\log\left\{\frac{\langle\mathbf{m}_1,\mathbf{m}_2\rangle}{\sqrt{\langle\mathbf{m}_1,\mathbf{m}_1\rangle\langle\mathbf{m}_2,\mathbf{m}_2\rangle}}\right\}\\ &= -\log\left\{\cos\angle(\mathbf{m}_1,\mathbf{m}_2)\right\}\end{aligned} \tag{5.10}$$

This means that the CS divergence measure is dependent of the cosine of the angle between the vectors $\mathbf{m}_1$ and $\mathbf{m}_2$ in Fig. 5.1 on page 41.

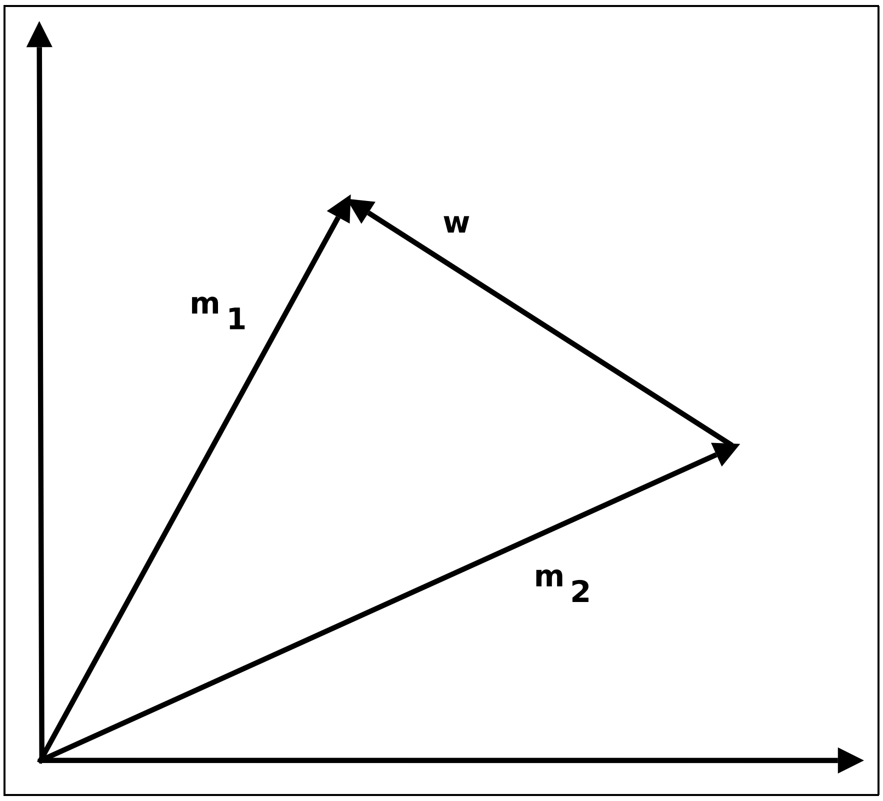

Figure 5.1: Illustration of the relationship between the mean vectors $\mathbf{m}_1$ *and* $\mathbf{m}_2$ *in the Mercer kernel feature space. The ISE divergence is given by the squared Euclidean distance between them,* $\|\mathbf{w}\|^2$*, where* $\mathbf{w} = \mathbf{m}_1 - \mathbf{m}_2$*. The CS divergence measure is related to the cosine of the angle between* $\mathbf{m}_1$ *and* $\mathbf{m}_2$*.*

5.3 The standard ISE classifier

In Fig. 5.1 in the previous section, we saw how the ISE between two distributions may be represented in terms of the Euclidean distance between the class mean vectors in the Mercer kernel feature space. In this section we use this geometric view of the ISE divergence between two classes to propose the standard ISE classifier.

We want to classify an unknown sample $\mathbf{x}_t$ to one of two classes ω_1 or ω_2. Let $\Phi(\mathbf{x}_t) = \mathbf{y}$ represent the sample in the Mercer kernel feature space. Then we may classify $\mathbf{x}_t$ using a *minimum Euclidean distance classifier* in the kernel space. That is, we simply classify $\mathbf{y}$ to the class of its closest mean vector.

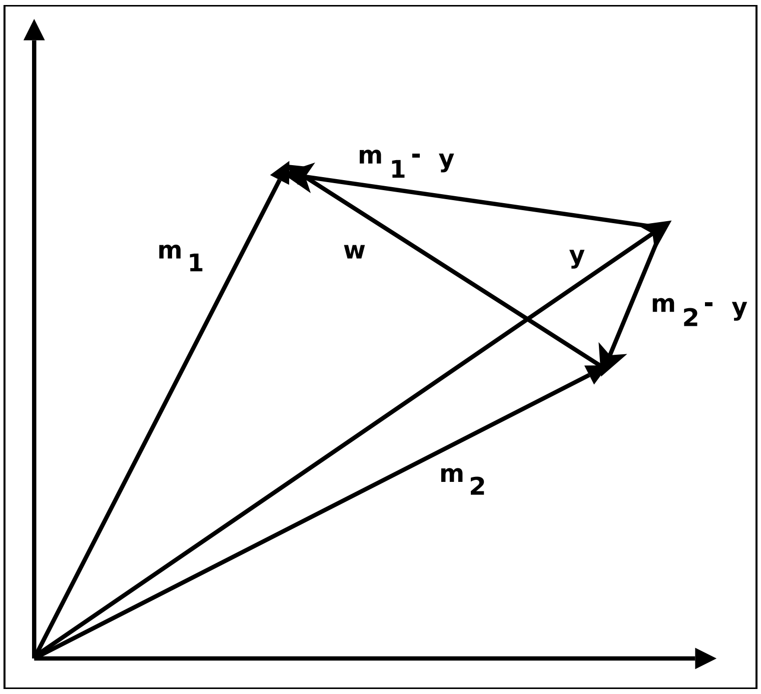

Figure 5.2: Classification in the Mercer kernel feature space.

In Fig. 5.2 the mean vectors $\mathbf{m}_1$ of class ω_1 and $\mathbf{m}_2$ of class ω_2 are illustrated. The vectors $\mathbf{m}_1 - \mathbf{y}$ and $\mathbf{m}_2 - \mathbf{y}$ are used to find which mean vector the unclassified point $\mathbf{y}$ belongs to. We may now use the following classification rule

$$\begin{aligned}
\mathbf{x_t} \to \omega_1 :\ & \|\mathbf{m}_1 - \mathbf{y}\|^2 - \|\mathbf{m}_2 - \mathbf{y}\|^2 \leq 0 \\
\Leftrightarrow\ & \|\mathbf{m}_1\|^2 - 2\mathbf{m}_1^T\mathbf{y} + \|\mathbf{y}\|^2 - \left(\|\mathbf{m}_2\|^2 - 2\mathbf{m}_2^T\mathbf{y} + \|\mathbf{y}\|^2\right) \leq 0 \\
\Leftrightarrow\ & \mathbf{m}_1^T\mathbf{y} - \mathbf{m}_2^T\mathbf{y} + b \geq 0 \\
\Leftrightarrow\ & \mathbf{w}^T\mathbf{y} + b \geq 0, \qquad (5.11)
\end{aligned}$$

where $\mathbf{w} = \mathbf{m}_1 - \mathbf{m}_2$ defines a hyperplane with $b = \frac{1}{2}[\|\mathbf{m}_2\|^2 - \|\mathbf{m}_1\|^2]$ as a threshold. The threshold b depends on the squared Euclidean norms of the mean values, which previously are shown to be equivalent to the class information potentials, and thus the class entropies. We also see that the proposed ISE classifier is a hyperplane classifier, since $\mathbf{w}$ defines a hyperplane separating two classes in a Mercer feature space. All the products in Eq. (5.11) are expressed in terms of inner-products, and may be calculated using the kernel trick as done in Eq. (5.5) on page 39. This means that we may implicitly operate in Mercer space, using kernel functions to evaluate all inner-products.

We now analyze the Mercer kernel feature space classification in terms of the Parzen window based estimators in the input space. We have

$$\mathbf{m}_1^T\mathbf{y} = \mathbf{m}_1^T\Phi(\mathbf{x}_t) = \frac{1}{N_1}\sum_{i=1}^{N_1}\Phi^T(\mathbf{x}_i)\Phi(\mathbf{x}_t) = \frac{1}{N_1}\sum_{i=1}^{N_1}W_{\sigma^2}(\mathbf{x}_t,\mathbf{x}_i) = \hat{p}(\mathbf{x}_t) \tag{5.12}$$

Likewise,

$$\mathbf{m}_2^T\mathbf{y} = \mathbf{m}_2^T\Phi(\mathbf{x}_t) = \frac{1}{N_2}\sum_{j=1}^{N_2}\Phi^T(\mathbf{x}_j)\Phi(\mathbf{x}_t) = \frac{1}{N_2}\sum_{j=1}^{N_2}W_{\sigma^2}(\mathbf{x}_t,\mathbf{x}_j) = \widehat{q}(\mathbf{x}_t) \tag{5.13}$$

Using the result in Eq. (5.5) we see that

$$\begin{aligned} b =& \frac{1}{2}[\|\mathbf{m}_2\|^2 - \|\mathbf{m}_1\|^2] \\ =& \frac{1}{2}[V_2(f) - V_1(f)]. \end{aligned} \tag{5.14}$$

Where $V_2(f) - V_1(f)$ is a measure of the difference in information potential between the classes, and thus the difference in entropy between class ω_1 and ω_2. This means that using the ISE divergence measure as a starting point, we may use the following classification rule in the input space

$$\mathbf{x}_t \to \omega_1 \quad : \quad \widehat{p}(\mathbf{x}_t) - \widehat{q}(\mathbf{x}_t) + b \geq 0, \tag{5.15}$$

where $\hat{p}(\mathbf{x}_t)$ is the Parzen window density estimate evaluated at the test point $\mathbf{x}_t$, given that the point belongs to class ω_1 and $\hat{q}(\mathbf{x}_t)$ the Parzen window density estimate given that the point belongs to class ω_2.

5.3.1 Connections to Parzen window Bayes classifier

In the two class case the ISE classifier is given by

$$\mathbf{x}_t \to \omega_1 \quad : \quad \hat{p}(\mathbf{x}_t) - \hat{q}(\mathbf{x}_t) + b \geq 0$$

$$b = \frac{1}{2}[V_2(f) - V_1(f)].$$

$V_1(f)$ and $V_2(f)$ are the information potentials of class ω_1 and ω_2. We notice that the ISE classification rule is similar to the Parzen window Bayes classification rule for equal a priori probabilities, given by

$$\mathbf{x}_t \to \omega_1 \quad : \quad \hat{p}(\mathbf{x}_t) - \hat{q}(\mathbf{x}_t) \geq 0$$

except from the threshold, $b = \frac{1}{2}[V_2(f) - V_1(f)]$. This threshold indicates that the ISE classifier is dependent on the difference in entropy or IP between the classes. If the entropy of the data in class one is larger than the entropy of class two, $V_2(f) - V_1(f) > 0$ the ISE classifier will assign the test point $\mathbf{x}_t$ to the class with largest entropy if $\hat{p}(\mathbf{x}_t) - \hat{q}(\mathbf{x}_t) = 0$. This indicates that the ISE classifier tends to classify data to the class with largest entropy or equivalently, the class with the smallest information potential. This may increase the probability of error at the cost of prioritizing the class with largest entropy. The effect of different class potentials will be investigated further in experiments.

5.3.2 Multiclass standard ISE classifier

Based on a training data set, we may define the class mean vectors $\mathbf{m}_1, \dots, \mathbf{m}_C$ for each of C classes $\omega_1, \dots, \omega_C$. We wish to classify some test sample $\mathbf{x}_t$, to the class which minimizes the ISE classification cost function in Eq. (5.7). This is achieved by measuring the squared Euclidean distance between $\Phi(\mathbf{x}_t)$ and each of the class mean vectors, and assign the test sample to the class for which the squared euclidean distance is smallest. This corresponds to the following classification rule

$$\begin{aligned}
\mathbf{x_t} \rightarrow \omega_c : \quad & \min_c \left(\|\mathbf{m}_c - \Phi(\mathbf{x}_t)\|^2 \right), \\
\Leftrightarrow \quad & \min_c \left(\langle \mathbf{m}_c, \mathbf{m}_c \rangle - 2\langle \mathbf{m}_c, \Phi(\mathbf{x}_t) \rangle + \langle \Phi(\mathbf{x}_t), \Phi(\mathbf{x}_t) \rangle \right) \\
\Leftrightarrow \quad & \min_c \left(\langle \mathbf{m}_c, \mathbf{m}_c \rangle - 2\langle \mathbf{m}_c, \Phi(\mathbf{x}_t) \rangle \right)
\end{aligned} \tag{5.16}$$

where $c = 1, \dots, C$, and

$$\begin{aligned}
\langle \mathbf{m}_c, \Phi(\mathbf{x}_t) \rangle &= \left\langle \frac{1}{N_c} \sum_{i=1}^{N_c} \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_t) \right\rangle \\
&= \left\langle \frac{1}{N_c} \sum_{i=1}^{N_c} \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_t) \right\rangle \\
&= \frac{1}{N_c} \sum_{i=1}^{N_c} W_{\sigma^2}(\mathbf{x}_t, \mathbf{x}_i) \\
&= \widehat{f}_c(\mathbf{x}_t)
\end{aligned} \tag{5.17}$$

as before,

$$\langle \mathbf{m}_c, \mathbf{m}_c \rangle = \|\mathbf{m}_c\|^2. \tag{5.18}$$

This is very similar to Mercer space k-means clustering,[26],[29] but we know the means of each cluster in this classification problem.

5.3.3 Using Non-Mercer kernels

In Section 4.4.4 we showed that the inner-product $\int p(\mathbf{x})q(\mathbf{x})\mathrm{d}\mathbf{x} = E_p\{q(\mathbf{x})\}$ can be estimated using any kind of density kernel $K_h(\cdot,\cdot)$. Notice that the ISE divergence measure is dependent on the density estimate at a test point $\mathbf{x}_t$, and a threshold given by the difference in information potential between the classes.

5.3.4 Connection to the graph cut

The set of points in an arbitrary feature space may be represented as a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the nodes, $\mathcal{V}$ are the points in the feature space, and $\mathcal{E}$ are the edges between each pair of nodes. The weight on each edge, $k(i,i')$, is a function of similarity between the nodes i and i'. Let node i and i' be represented with feature vectors $\mathbf{x}_i$ and $\mathbf{x}_{i'}$, respectively, $i,i' = 1,\ldots,N$.

An exponential function is often used as a similarity measure [9]

$$k(i,i') = \exp\left\{\frac{\|\mathbf{x}_i - \mathbf{x}_{i'}\|^2}{2\sigma_{\mathcal{G}}^2}\right\}, \tag{5.19}$$

where $\sigma_{\mathcal{G}}$ is the width of the exponential function associated with the graph G. A graph may be partitioned into two disjoint sets $\mathcal{G}_1, \mathcal{G}_2$, $\mathcal{G}_1 \cup \mathcal{G}_2 = \mathcal{V}$, $\mathcal{G}_1 \cap \mathcal{G}_2 = \emptyset$, with points $\mathbf{x}_i$, $i = 1,\ldots,N_1 \in \mathcal{G}_1$ and $\mathbf{x}_j$, $j = 1,\ldots,N_2 \in \mathcal{G}_2$. The degree of dissimilarity between these two pieces can be computed as total weight of the edges that have been removed. In graph theory, this is called the *cut* [27]

$$cut(\mathcal{G}_1, \mathcal{G}_2) = \sum_{i,j=1}^{N_1 N_2} k(\mathbf{x}_i, \mathbf{x}_j). \tag{5.20}$$

Assume that the points in $\mathcal{G}_1$ and $\mathcal{G}_2$ have distribution functions $p(\mathbf{x})$ and $q(\mathbf{x})$. Now, using the Parzen window-based estimator, another interpretation of the cut is

$$\int \hat{p}(\mathbf{x})\hat{q}(\mathbf{x})\mathrm{d}\mathbf{x} = \frac{1}{N_1 N_2}\sum_{i,j=1}^{N_1,N_2} k(\mathbf{x}_i, \mathbf{x}_j), \tag{5.21}$$

where $k(\mathbf{x}_i, \mathbf{x}_j) = W_{2\sigma^2}(\mathbf{x}_i, \mathbf{x}_j)$, and $W_{2\sigma^2}(\cdot,\cdot)$ is the Gaussian Parzen window.

The total sum of all the edges in a graph is called the *volume* of the graph

$$vol(\mathcal{G}) = \sum_{i,i'=1}^{N,N} k(x_i, x_{i'}). \tag{5.22}$$

From Section 4.4.1 we know that the information potential can be written as

$$V(f) = \frac{1}{N^2}\sum_{i,i'=1}^{N,N} k(\mathbf{x}_i, \mathbf{x}_{i'}), \tag{5.23}$$

where $k(\mathbf{x}_i, \mathbf{x}_{i'})$ is defined as in Eq. (5.21). The connection between the IP and volume of the graph is given by

$$N^2 V(f) = vol(\mathcal{G}), \tag{5.24}$$

where f denotes the distribution of the points $\mathbf{x}_i, i = 1, \ldots, N$.

The integrated squared error and graph theory

The Parzen window-based estimator for the ISE divergence is now connected to graph theory by [9]

$$\begin{aligned} \widehat{ISE}\{p,q\} &= \frac{1}{N_1^2}\sum_{i,i'=1}^{N_1,N_1} k(\mathbf{x}_i, \mathbf{x}_{i'}) + \frac{1}{N_2^2}\sum_{j,j'=1}^{N_2,N_2} k(\mathbf{x}_j, \mathbf{x}_{j'}) - \frac{2}{N_1 N_2}\sum_{i,j=1}^{N_1,N_2} k(\mathbf{x}_i, \mathbf{x}_j) \\ &= N_1^2 vol(\mathcal{G}_1) + N_2^2 vol(\mathcal{G}_2) - 2N_1N_2 cut(\mathcal{G}_1, \mathcal{G}_2). \end{aligned} \tag{5.25}$$

Chapter 6

A Laplacian ISE classifier

We know from Section 5.3 that the ISE classifier may be viewed as a hyperplane classifier. Inspired both by the SVM and the *Laplacian classifier* presented in [12], in this chapter, we modify the standard ISE classifier by introducing a weighting of inner-products in the ISE divergence measure.

The SVM is in a similar way to the ISE classifier based on finding a hyperplane to separate class data. In the SVM the task is to find the hyperplane that maximizes the margin between class data in a Mercer kernel feature space. The inner-products in this space can be computed by using the kernel trick with a Mercer kernel function. The maximization of the margin for the SVM leads to a weighting of the training data points when constructing the classifier. The points on the margin are known as the support vectors. To obtain the relevant weighting, which determines the N_{sv} support vectors, a convex optimization problem must be solved. This procedure has to select two SVM parameters and is far from straightforward.

For many data sets, low values for the overall probability density function will correspond to class boundary regions. In [12] a classifier named the *Laplacian classifier* based on the Cauchy-Schwarz divergence is presented, where the CS cost function uses weighted inner-products to emphasize the samples with small overall probability. This makes sense, since the test data points close to the class boundaries often are the most difficult to classify correctly. The Laplacian classifier does not require the optimization phase associated with the SVM, but produces similar results as the SVM in several cases [12].

The previously defined ISE classifier is now modified to a weighted version to emphasize points near the class borders in the same way as done in [12]. Next we discuss how the weighted version of the ISE cost function connects to the Bayes probability of error and the *Laplacian data matrix*. The Laplacian matrix has recently been used in many problems in clustering [9], and it may be interesting to see if it can be used in an ISE based classifier. Some classification results using the weighted ISE classifier, which we refer to as the *Laplacian ISE classifier*, are presented in Part II, Section 9.4.

6.1 Modified ISE divergence

In this section the ISE divergence previously defined by unweighted inner-products is weighted to emphasize the points in the class boundaries. Consider two data classes, ω_1 and ω_2, with corresponding probability density functions $p(\mathbf{x})$ and $q(\mathbf{x})$. Let $f(\mathbf{x}) = P_1 p(\mathbf{x}) + P_2 q(\mathbf{x})$ be the overall pdf of the data set with P_1 and P_2 as class priors. Define the weighted inner-product $\langle p, q\rangle_f \equiv \int p(\mathbf{x})q(\mathbf{x})f(\mathbf{x})^{-1}\mathrm{d}\mathbf{x}$. The cost function used in the ISE classifier is now given by

$$D_{ISE}\{p,q\} = \langle p,p\rangle_f - 2\langle p,q\rangle_f + \langle q,q\rangle_f. \tag{6.1}$$

The only difference between this and the previous version of the cost function is the inner-product weighting.

6.2 Connection to the Bayes probability of error

Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two regions in the data space. If a test sample $\mathbf{x}_t \in \mathcal{R}_1$, it will be assigned to class ω_1. Otherwise, $\mathbf{x}_t \in \mathcal{R}_2$, and it will be assigned to class ω_2. Similar to the derivation of the Bayes probability of error, the regions $\mathcal{R}_1$ and $\mathcal{R}_2$ must be determined such that the classification cost function is optimized. Assume that the classes are relatively well separated, then $f(\mathbf{x}_t) \approx P_1 p(\mathbf{x}_t)$ for $\mathbf{x}_t \in \mathcal{R}_1$ and $f(\mathbf{x}_t) \approx P_2 q(\mathbf{x}_t)$ for $\mathbf{x}_t \in \mathcal{R}_2$. Now consider each of the inner-products in Eq. (6.1)

$$\begin{aligned}
&\langle p,p\rangle_f \\
&= \int p^2(\mathbf{x}) f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x} \\
&= \int_{\mathcal{R}_1} p^2(\mathbf{x}) f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x} + \int_{\mathcal{R}_2} p^2(\mathbf{x}) f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x} \\
&\approx \frac{1}{P_1},
\end{aligned} \tag{6.2a}$$

where $\int_{\mathcal{R}_2} p^2(\mathbf{x}) f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x} \approx 0$ because we have assumed that the classes are well separated, and thus $p(\mathbf{x})$ is very small in region $\mathcal{R}_2$.

$$\begin{aligned}
&2\langle p,q\rangle_f \\
&= 2\int p(\mathbf{x})q(\mathbf{x}) f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x} \\
&= 2\left[\int_{\mathcal{R}_1} p(\mathbf{x})q(\mathbf{x}) f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x} + \int_{\mathcal{R}_2} p(\mathbf{x})q(\mathbf{x}) f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}\right] \\
&\approx 2\left[\frac{1}{P_1}\int_{\mathcal{R}_1} q(\mathbf{x})\mathrm{d}\mathbf{x} + \frac{1}{P_2}\int_{\mathcal{R}_2} p(\mathbf{x})\mathrm{d}\mathbf{x}\right].
\end{aligned} \tag{6.2b}$$

$$\begin{aligned}&\langle q,q\rangle_f\\&=\int q^2(\mathbf{x})f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}\\&=\int_{\mathcal{R}_1} q^2(\mathbf{x})f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}+\int_{\mathcal{R}_2} q^2(\mathbf{x})f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}\\&\approx\frac{1}{P_2},\end{aligned}\tag{6.2c}$$

where $\int_{\mathcal{R}_1} q^2(\mathbf{x})f^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}\approx 0$ because we have assumed that the classes are well separated, and thus $q(\mathbf{x})$ is very small in region $\mathcal{R}_1$.

Now the weighted version of the ISE divergence measure in the well separated case can be written as

$$D_{ISE_f}\{p,q\}\approx\frac{1}{P_1}-2\left[\frac{1}{P_1}\int_{\mathcal{R}_1} q(\mathbf{x})\mathrm{d}\mathbf{x}+\frac{1}{P_2}\int_{\mathcal{R}_2} p(\mathbf{x})\mathrm{d}\mathbf{x}\right]+\frac{1}{P_2}.\tag{6.3}$$

The probability of error for the two class Bayes classifier is given by

$$P_e=P_2\int_{\mathcal{R}_1} q(\mathbf{x})\mathrm{d}\mathbf{x}+P_1\int_{\mathcal{R}_2} p(\mathbf{x})\mathrm{d}\mathbf{x}\tag{6.4}$$

and we see that the weighted ISE divergence may be written as

$$\begin{aligned}D_{ISE_f}\{p,q\}\approx&\frac{1}{P_1}+\frac{1}{P_2}-\frac{2P_e}{P_1P_2}\\=&\frac{1}{P_1P_2}\left[1-2P_e\right].\end{aligned}\tag{6.5}$$

Thus, minimizing the probability of error also maximizes the weighted divergence measure in the case where the class distributions are well separated.

6.3 Kernel space and Laplacian matrix representation

In this section we review the connection between a Parzen window-based estimator for the $f^{-1}(\mathbf{x})$ weighted ISE divergence and the Laplacian data matrix. The weighted ISE divergence may be expressed as

$$\begin{aligned}D_{ISE}\{p,q\}=&\langle p,p\rangle_f-2\langle p,q\rangle_f+\langle q,q\rangle_f\\=&\int h_1^2(\mathbf{x})\mathrm{d}\mathbf{x}-2\int h_1(\mathbf{x})h_2(\mathbf{x})\mathrm{d}\mathbf{x}+\int h_2^2(\mathbf{x})\mathrm{d}\mathbf{x},\end{aligned}\tag{6.6}$$

where $h_1(\mathbf{x})=f^{-\frac{1}{2}}(\mathbf{x})p(\mathbf{x})$ and $h_2(\mathbf{x})=f^{-\frac{1}{2}}(\mathbf{x})q(\mathbf{x})$. We are given a training data set $\mathbf{x}_l, l=1,\ldots,N$. This data set consists of the class 1 data points, $\mathbf{x}_i, l=1,\ldots,N_1$, and

$\mathbf{x}_j, = 1, \dots, N_2$, the class 2 data points. Based on the data samples, define the Parzen window based estimators [12]

$$\begin{aligned}
\hat{f}(\mathbf{x}) =& \frac{1}{N}\sum_{l=1}^{N} W_{\sigma^2}(\mathbf{x}, \mathbf{x}_l), \\
\hat{h}_1(\mathbf{x}) =& \frac{1}{N_1}\sum_{i=1}^{N_1} \hat{f}^{-\frac{1}{2}}(\mathbf{x}_i) W_{\sigma^2}(\mathbf{x}, \mathbf{x}_i), \\
\hat{h}_2(\mathbf{x}) =& \frac{1}{N_2}\sum_{j=1}^{N_2} \hat{f}^{-\frac{1}{2}}(\mathbf{x}_j) W_{\sigma^2}(\mathbf{x}, \mathbf{x}_j).
\end{aligned} \tag{6.7}$$

Here, $W_{\sigma^2}(\cdot, \cdot)$ is the Parzen window. We have here assumed that a Parzen window with a Gaussian kernel, with uniform bandwidth, σ^2 is used in all estimators. Any kernel function $K_h(\cdot, \cdot)$, with bandwidth h, satisfying Mercer's theorem may be used instead of the Gaussian kernel $W_{\sigma^2}(\cdot, \cdot)$, see also Section 4.4.4 on page 35. Now we have

$$\begin{aligned}
&\int \hat{h}_1(\mathbf{x})\hat{h}_2(\mathbf{x})\mathrm{d}\mathbf{x} \\
=& \int \frac{1}{N_1}\sum_{i=1}^{N_1} \frac{W_{\sigma^2}(\mathbf{x}, \mathbf{x}_i)}{\hat{f}^{\frac{1}{2}}(\mathbf{x}_i)} \frac{1}{N_2}\sum_{j=1}^{N_2} \frac{W_{\sigma^2}(\mathbf{x}, \mathbf{x}_j)}{\hat{f}^{\frac{1}{2}}(\mathbf{x}_j)}\mathrm{d}\mathbf{x} \\
=& \frac{1}{N_1 N_2}\sum_{i,j=1}^{N_1,N_2} \frac{1}{\hat{f}^{\frac{1}{2}}(\mathbf{x}_i)\hat{f}^{\frac{1}{2}}(\mathbf{x}_j)} \int W_{\sigma^2}(\mathbf{x}, \mathbf{x}_i) W_{\sigma^2}(\mathbf{x}, \mathbf{x}_j)\mathrm{d}\mathbf{x} \\
=& \frac{1}{N_1 N_2}\sum_{i,j=1}^{N_1,N_2} \frac{W_{2\sigma^2}(\mathbf{x}_i, \mathbf{x}_j)}{\hat{f}^{\frac{1}{2}}(\mathbf{x}_i)\hat{f}^{\frac{1}{2}}(\mathbf{x}_j)},
\end{aligned} \tag{6.8}$$

where the convolution theorem for Gaussians has been used in the last step. For any pair of data points in the training data set, say $\mathbf{x}_l$ and $\mathbf{x}_{l'}$, we define the *affinity matrix* $\mathbf{K}$, such that element (l, l') equals $W_{2\sigma^2}(l, l')$. We also define a matrix $\mathbf{D} = (\hat{f}(\mathbf{x}_1), \dots, \hat{f}(\mathbf{x}_N))$. Now, all $\hat{f}^{-\frac{1}{2}}(\mathbf{x}_l)W_{2\sigma^2}(\mathbf{x}_l, \mathbf{x}_{l'})\hat{f}^{-\frac{1}{2}}(\mathbf{x}_{l'})$ can be represented by element (l, l') of the matrix $\mathbf{K}_f = \mathbf{D}^{-\frac{1}{2}}\mathbf{K}\mathbf{D}^{-\frac{1}{2}}$. The matrix $\mathbf{K}_f$ is known as the *Laplacian matrix* [12].

Each element of the matrix $\mathbf{K}$ represents an inner-product in the Mercer kernel feature space, since the Gaussian kernel satisfies the Mercer conditions mentioned in chapter 5. Now, each element in $\mathbf{K}_f$ also represents an inner-product, which we may denote

$$\langle \Phi_f(\mathbf{x}_l), \Phi_f(\mathbf{x}_{l'}) \rangle = \hat{f}^{-\frac{1}{2}}(\mathbf{x}_l)W_{2\sigma^2}(\mathbf{x}_l, \mathbf{x}_{l'})\hat{f}^{-\frac{1}{2}}(\mathbf{x}_{l'}). \tag{6.9}$$

Which gives

$$\begin{aligned}&\int \hat{h}_1(\mathbf{x})\hat{h}_2(\mathbf{x})\mathrm{d}\mathbf{x}\\ =&\frac{1}{N_1N_2}\sum_{i,j=1}^{N_1,N_2}\langle\Phi(\mathbf{x}_i),\Phi(\mathbf{x}_j)\rangle\\ =&\left\langle\frac{1}{N_1}\sum_{i=1}^{N_1}\Phi_f(\mathbf{x}_i),\frac{1}{N_2}\sum_{j=1}^{N_2}\Phi_f(\mathbf{x}_j)\right\rangle\\ =&\langle\mathbf{m}_{1f},\mathbf{m}_{2f}\rangle,\end{aligned}\tag{6.10}$$

where $\mathbf{m}_{1f}=\frac{1}{N_1}\sum_{i=1}^{N_1}\Phi_f(\mathbf{x}_i)$ and $\mathbf{m}_{2f}=\frac{1}{N_2}\sum_{j=1}^{N_2}\Phi_f(\mathbf{x}_j)$ are the class mean vectors after the mapping to the Mercer kernel feature space. This non-linear mapping is given by $\Phi_f(\cdot)$. The same analysis may bedone for $\int \hat{h}_1^2(\mathbf{x})\mathrm{d}\mathbf{x}$ and $\int \hat{h}_2^2(\mathbf{x})\mathrm{d}\mathbf{x}$. Now the weighted ISE classifier can be written in the same way as the unweighted version in Eq. (5.16). The only difference is that the data mapping is now to a different Mercer kernel feature space, given by the eigenvectors of the Laplacian matrix $\mathbf{K}_f$ instead of the space given by the eigenvectors of the affinity matrix $\mathbf{K}$. We repeat the classifier rule for a test point $\mathbf{x}_t$

$$\mathbf{x_t}\rightarrow\ \min_c\left(\langle\mathbf{m}_c,\mathbf{m}_c\rangle_f-2\langle\mathbf{m}_c,\Phi(\mathbf{x}_t)\rangle_f\right),\tag{6.11}$$

where $c=1,\ldots,C$, are the class labels and

$$\begin{aligned}\langle\mathbf{m}_c,\Phi(\mathbf{x}_t)\rangle_f&=\left\langle\frac{1}{N_c}\sum_{i=1}^{N_c}\Phi_f(\mathbf{x}_i),\Phi_f(\mathbf{x}_t)\right\rangle\\ &=\left\langle\frac{1}{N_c}\sum_{i=1}^{N_c}\Phi_f(\mathbf{x}_i)^T\Phi_f(\mathbf{x}_t)\right\rangle\\ &=\frac{1}{N_c}\sum_{i=1}^{N_c}f^{-1}(\mathbf{x}_i)W_{\sigma^2}(\mathbf{x}_t,\mathbf{x}_i)\end{aligned}\tag{6.12}$$

as before, but in a different Mercer space,

$$\langle\mathbf{m}_c,\mathbf{m}_c\rangle_f=\|\mathbf{m}_c\|^2.\tag{6.13}$$

6.3.1 Illustration of weights

To illustrate the effect of the weighting of the data we created two classes. Class 1 is represented with 150 samples from a dense Gaussian distribution with mean $[0, 0]^T$. Class 2 is represented with 150 samples from a circle distribution with the same mean. In Fig. 6.1 the samples are plotted with the 10 points having largest weights marked with star symbols. We notice that points with the largest weights all are on the borders of the circle distribution, which is expected since the circle points have a much more sparse distribution.

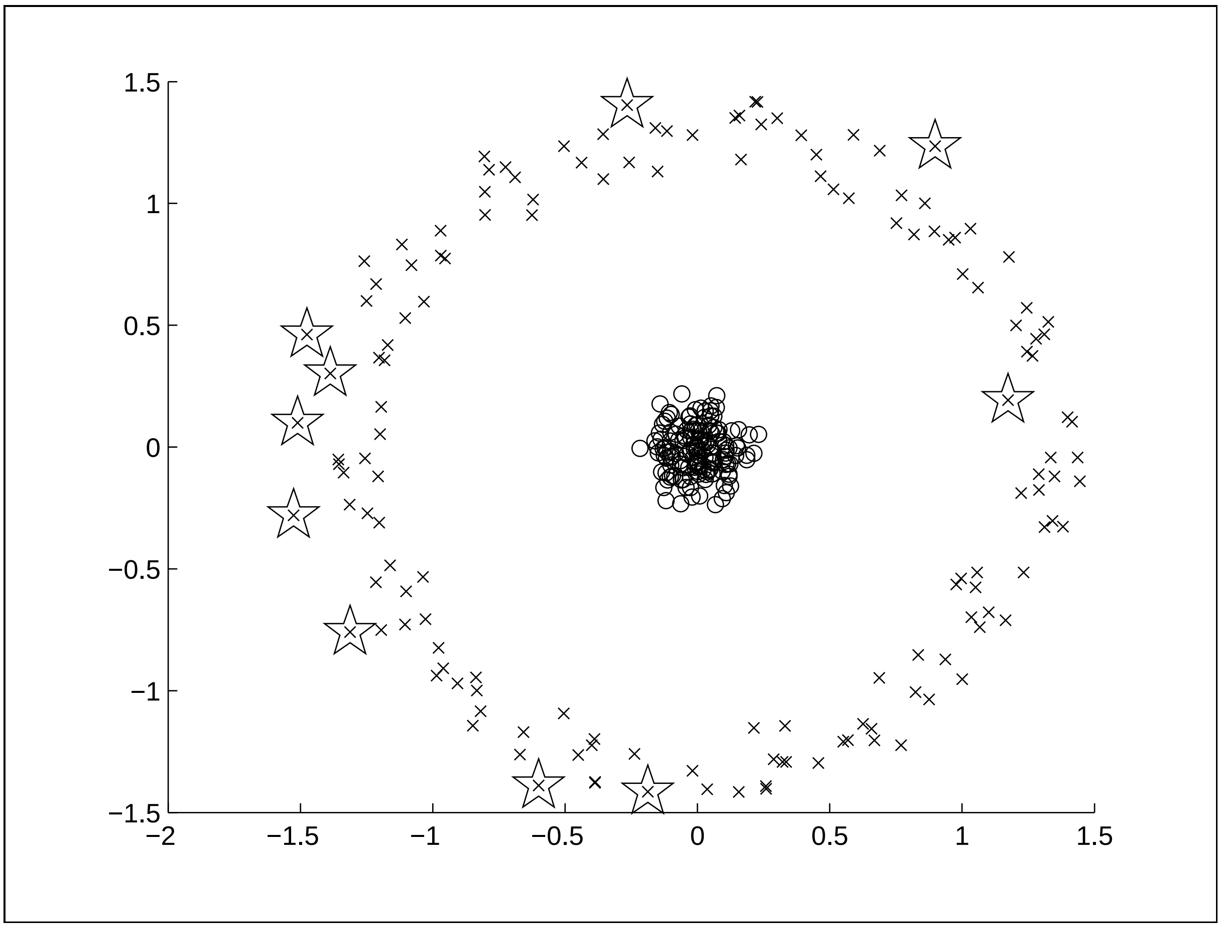

Figure 6.1: Note that all of the 10 largest weighted points belong to the boundaries of the circle distribution.

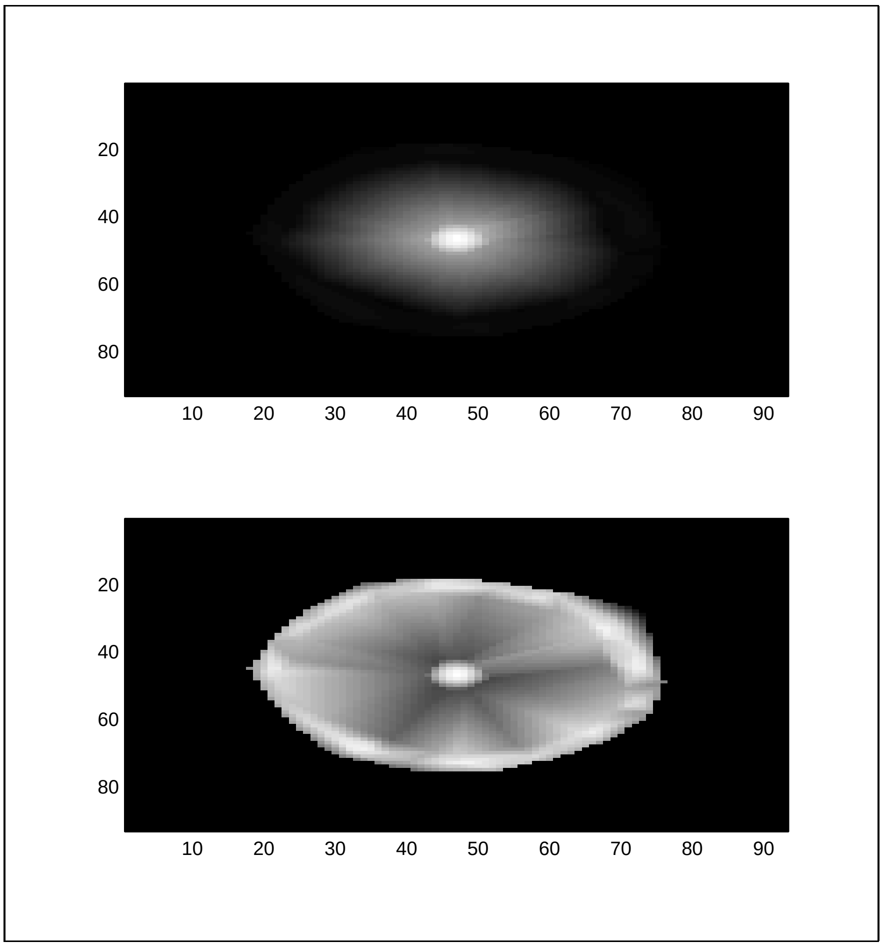

Figure 6.2: Illustrated shape of the density estimates of the two classes. In the top figure the samples are not weighted, but in the lower the samples are weighted with the inverse of the overall probability for each point.

In Fig. 6.2 we see the effect of applying weights to the sample data. The distributions illustrated in the lower figure clearly emphasizes the sparse points in the ring data, compared to the unweighted estimate in the upper figure.

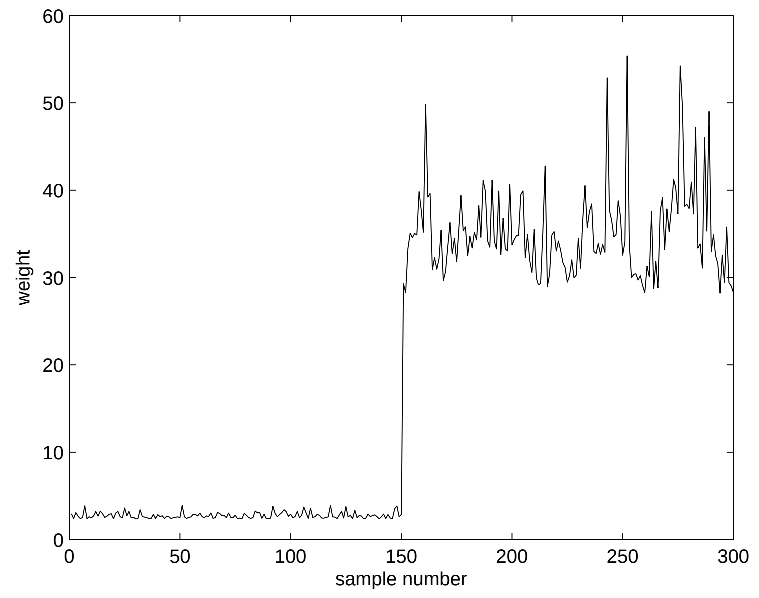

Figure 6.3: Plot of the weights for each sample. The first 150 points are from the Gaussian distribution and the last 150 points are from the circle distribution.

In Fig. 6.3 we see the weights for each data sample in class 1 and class 2. The first 150 weights belong to the dense Gaussian distributed data in class 1, while the last 150 weights belong to the clearly emphasized circle data points.

Chapter 7

Spectral ISE classifiers

Spectral methods are popular especially in clustering methods. Spectral methods are based on an affinity matrix, containing pairwise relationships between the samples, and depend on the spectral properties of this matrix. This matrix is eigendecomposed to find a more useful data representation of the original data. Until recently, only the points used in the affinity matrix have been possible to represent in a kernel feature space. This is probably the main reason spectral methods rarely are used to classify new test samples. By using the *Nyström routine* [31], however, the mapping of new points to the kernel feature space is now possible. In the previous chapters we have used the ISE classifier by operating in the Mercer kernel feature space implicitly, by evaluation of inner-products. We know that the ISE divergence measure is related to the class mean vectors in a Mercer kernel feature space, and to the squared Euclidean distance between the mean vectors and the test samples in this space. Creating an affinity or Laplacian matrix with the training data, and eigendecompose it, we can find the class means operating directly in an approximate Mercer kernel space with the projected training samples.

If the data set has outliers, it may be beneficial to use the class median vectors instead of the means in the approximated Mercer kernel space. With the Nyström routine we can project the test samples to the same space, calculate distances and thus evaluate the ISE divergence directly in the space spanned by either the affinity or Laplacian matrix.

In this chapter we assume that the affinity and Laplacian matrix elements are inner-products created with Mercer kernels, unless some other kernel type is specified. Based on the eigendecomposition of an affinity or Laplacian matrix and projection of training data and samples onto the C dominant eigenvectors, we now propose spectral versions of the standard and Laplacian ISE classifier. We refer to the spectral ISE classifier based on the eigendecomposition of the affinity matrix as the *spectral ISE classifier* and the spectral ISE classifier using the Laplacian matrix as the *spectral Laplacian ISE classifier.*

Some results using the spectral ISE classifier are presented in Part II, Section 9.5, and results using the spectral Laplacian ISE classifier are presented in Part II, Section 9.6.

7.1 Mapping of data to a Mercer based feature space

An approximation of the nonlinear mapping of the training data $\Phi(\mathbf{x}_l), l = 1, \cdots, N$, from input space to the Mercer kernel space, using the C largest eigenvalues and corresponding eigenvectors of the kernel matrix $\mathbf{K}$, is accomplished with [13], [24]

$$\begin{aligned} &\Phi : R^d \rightarrow \mathcal{F} \\ &\mathbf{x}_l \rightarrow \Phi(\mathbf{x}_l) \approx \left[\sqrt{\lambda_1} e_{1l}, \sqrt{\lambda_2} e_{2l}, \ldots, \sqrt{\lambda_C} e_{Cl}\right]^T, \quad l = 1, \cdots, N, \end{aligned} \tag{7.1}$$

where e_{ml} denotes the lth element of the mth eigenvector of $\mathbf{K}$ and λ_m is the corresponding eigenvalue, where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_C$. It can be shown [13] that in the ideal case with C clusters of the training data corresponding to C different classes that are "infinitely" far apart, the eigendecomposition of the affinity matrix results in C point clusters, mutually orthogonal to each other situated on the C first principal axes in the kernel space [13].

When using the affinity matrix $\mathbf{K}$ to create the basis which the data is projected on, this is the same as performing a C-dimensional kernel PCA on the training data [24]. The affinity matrix $\mathbf{K}$ and the Laplacian matrix $\mathbf{K}_f$ are created with training data samples as described in Section 6.3 on page 49.

We can compute the C-dimensional vector projection of a test sample $\mathbf{x}_t$ into the subspace spanned by the C eigenvectors of the kernel matrix with [6]

$$\Phi(\mathbf{x}_t) \approx \left(\sqrt{N} \sum_{i=1}^{N} \alpha_i^j \kappa(\mathbf{x}_i, \mathbf{x}_t)\right)_{j=1}^{C}, \tag{7.2}$$

where $\alpha^j = \lambda_j^{-\frac{1}{2}} \mathbf{e}_j$ is given by the corresponding eigenvector and eigenvalue of the kernel matrix. $\kappa(\mathbf{x}_i, \mathbf{x}_t)$ is a Mercer kernel function computing the inner-products between the new test sample $\mathbf{x}_t$ and all $i = 1, \ldots, N$ samples in the kernel matrix. The computation in Eq. (7.2) is also known as the Nyström routine. The Mercer space spanned by the Laplacian matrix is not the same as the Mercer space spanned by the kernel matrix. The procedure to map the training data and new samples is however the same, except that we eigendecompose $\mathbf{K}_f$, not $\mathbf{K}$, in Eq. (7.2) and Eq. (7.1).

7.1.1 Illustration of mappings to Mercer space

In Fig. 7.1 we see the same circle and Gaussian distributions as in Fig. 6.3 on page 54, after the projection to space spanned by the eigenvectors corresponding to the two largest eigenvalues of the Laplacian matrix $\mathbf{K}_f$. We see that the two classes seem to be distributed along two clearly separable lines in this space. This seems to be close to the ideal case for separable data. In this two-class case, we expect the data to be situated in two clusters, mutually orthogonal along the two first principal axes in the space of the Laplacian matrix.

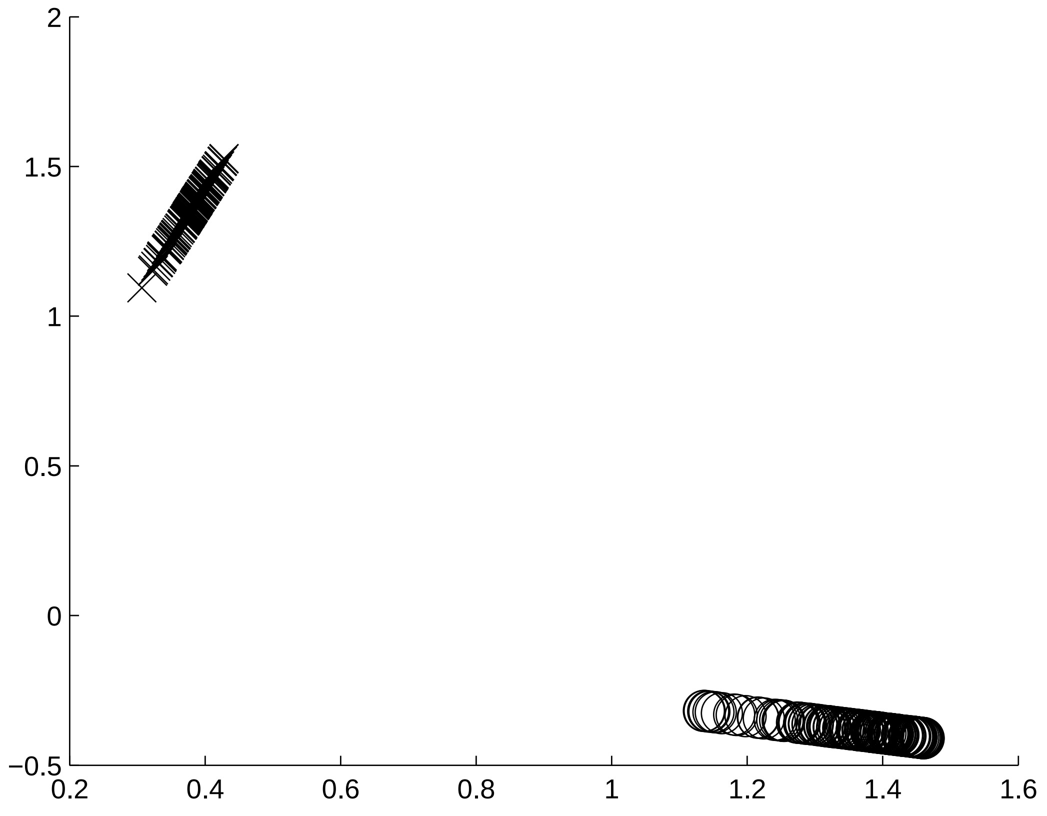

Figure 7.1: Gauss and circle distributions in an approximate Mercer space, given by the two principal eigenvectors of the Laplacian matrix $\mathbf{K}_f$. Samples from the Gaussian distribution are labeled with $\bigcirc$, and circle distribution samples are labeled with $\times$.

In Fig. 7.2 we see the same distributions as in Fig. 6.3 after the projection to the space spanned by the eigenvectors corresponding to the two largest eigenvalues of the affinity matrix **K**. Note that all training points that belong to the circle distribution seem to be mapped to origo, while the points that belong to the Gaussian distribution are spread around more uniformly.

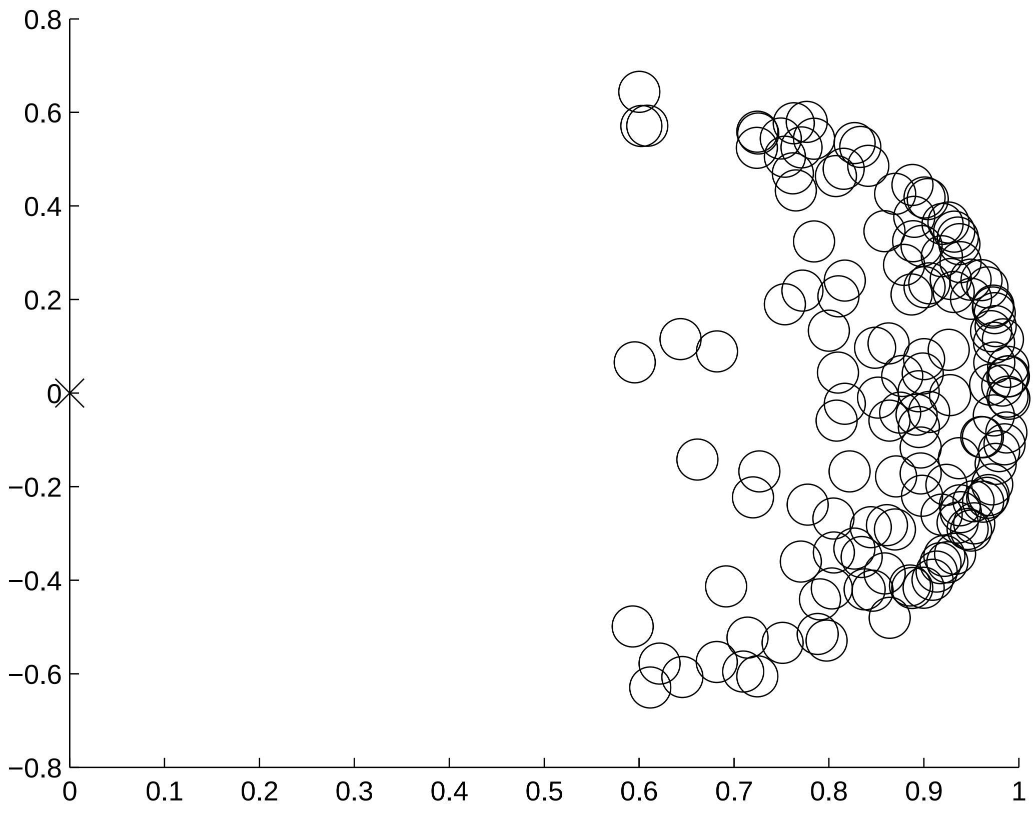

Figure 7.2: Gauss and circle distributions in an approximate Mercer space, given by the affinity matrix **K**. *Samples from the Gaussian distribution are labeled with* ○*, and circle distribution samples are labeled with* $\times$.

7.2 Spectral versions of the ISE classifiers

7.2.1 The spectral ISE classifier

In this section we discuss how we can develop spectral classifiers based on the ISE divergence cost function. Assume we have a training data set with labels from C different classes, $\omega_c, c = 1, \ldots, C$. From this data set we now construct the kernel matrix $\mathbf{K}$. This matrix is then eigendecomposed and the training data is projected to the space spanned by the C dominant eigenvectors of $\mathbf{K}$ using Eq. (7.1). Using the projected training data, we can now find the mean vectors of each class in the approximated kernel space. If the data set has outliers, it may be useful to use the median vectors instead of the mean vectors. With Eq. (7.2) we project each of the test samples to the same space as the training data, and measure the squared Euclidean distance between the sample and each of the class means. Finally, each sample is classified to the class where the distance is smallest. To summarize the steps:

- Find the affinity matrix $\mathbf{K}$ using training data $\mathbf{x}_l,\ l = 1, \ldots, N$
- Eigendecompose $\mathbf{K}$ and compute $\Phi(\mathbf{x}_l) \approx \left[\sqrt{\lambda_1} e_{1l}, \sqrt{\lambda_2} e_{2l}, \ldots, \sqrt{\lambda_C} e_{Cl}\right]^T,\ l = 1, \ldots, N.$
- Find the mean or median vectors of the projected data, $\mathbf{m}_c$ for class $\omega_c,\ c = 1, \ldots, C,$
- for i=1:number of test points to classify

  1 Map $\mathbf{x}_i$ to the approximate kernel space with Eq. (7.2)

  2 Find the squared Euclidean distances, $d_c,\ c = 1, \ldots, C$, between $\mathbf{m}_c$ and $\Phi(\mathbf{x}_i)$

  3 Classify: $\Phi(\mathbf{x}_i) \in \omega_c$ if $d_c < d_k,\ \forall k \neq c$

7.2.2 The spectral Laplacian ISE classifier

This follows the same routine as the spectral ISE classifier, except that instead of eigendecomposing the affinity matrix $\mathbf{K}$, we now use the Laplacian matrix $\mathbf{K}_f$.

Part II

Analysis and experiments

Chapter 8

Kernel selection

All our versions of the ISE based classifier are highly dependent on density estimation, since they are derived using Parzen windowing. Assume we have a set of data $\mathbf{x}_1, \ldots, \mathbf{x}_N$ generated i.i.d according to some unknown distribution, where this distribution describes data from one specific class. We then need to find the density estimate for the data set. This can often be a problem, both because of the curse of dimensionality, and because distributions don't always possess a density [17]. The Parzen window density estimates are dependent of kernel size and kernel type. Thus, if we can find the optimal kernel for our data set for the Parzen window density estimation, we have an appropriate kernel for the classifier. There exist many types of kernels which can be used in density estimation. A short summary is given in Section 3.2.4. It has been proved [30] that the Epanechnikov kernel gives a better density estimate then the Gaussian kernel, in terms of the number of data points needed to get a good estimate. This gives us a good reason to check if the ISE classifiers may benefit from using this kernel, even if it does not satisfy Mercer's theorem. In this chapter we will use some artificial distributions and analyze the effect of different kernel functions and bandwidths on the classification rates and density estimates. We need to see if we can get good classification results, even when the density estimates are far from exact.

8.1 Effect of kernel bandwidth and kernel type

In this section we aim to demonstrate how the density estimates and the different versions of the ISE classifier may behave using different kernels and bandwidths on an artificial data set. We want to check if some of the versions of the ISE classifier are more robust, i.e. give good classification results over a wider range of kernels then others. We also check two non-Mercer kernels which often are used in density estimation, the square and the Epanechnikov kernels.

We use the same data set as previously in Section 6.3.1. Class ω_1 is represented with 150 samples from a dense Gaussian distribution, with mean $[0, 0]^T$. Class ω_2 is represented with 150 samples from a circular shaped distribution, with the same mean as ω_1. The two classes are illustrated again in Fig. 8.1

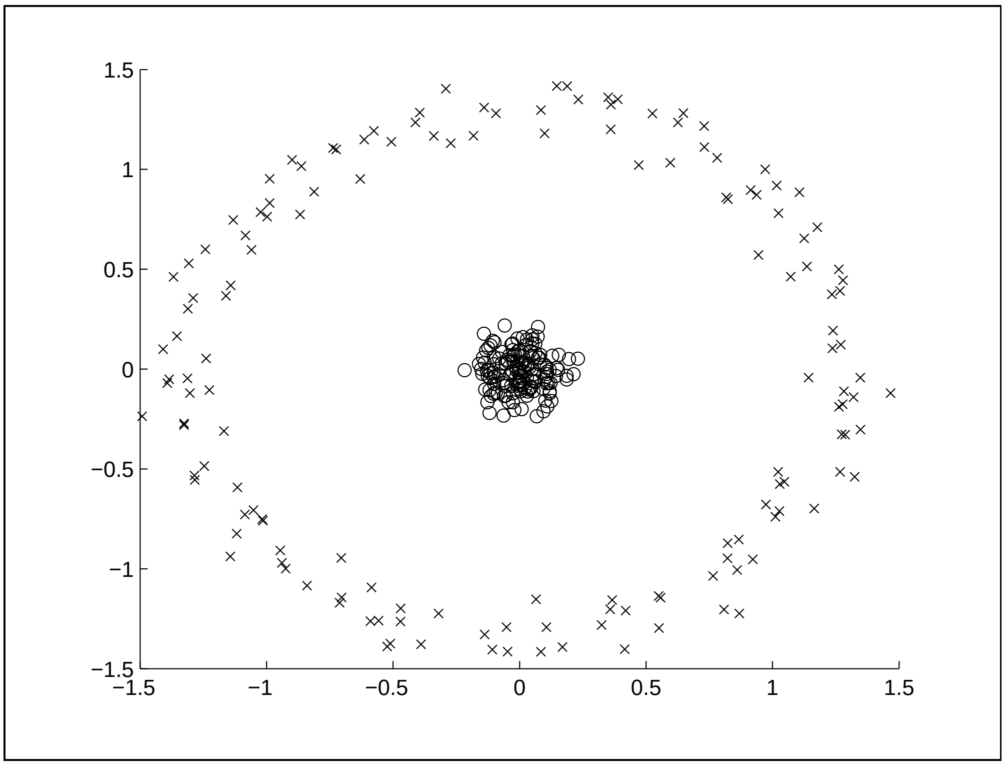

Figure 8.1: ω_1 samples illustrated with $\circ$ symbols, ω_2 illustrated with $\times$ symbols.

To test the different kernel types and bandwidths we use 20 and 50 points for test and training from each class, respectively. The test points are then classified using the different classifiers with different kernel types over a range of kernel bandwidths. In Table 8.1, Table 8.2 and Table 8.3 following, we give the range of kernel bandwidths which gives 100% correct classification rates for the different classifiers and kernels. Since the class data does not have any extreme outliers, we don't expect the spectral median versions to be much different from the mean versions, and we only include the spectral classifiers using mean vectors in the ISE divergence measure.

Table 8.1: Gaussian kernel

Classifier	bandwidth range
Standard ISE	0.30-0.76
Laplacian ISE	0.09-0.91
Spectral ISE means	0.07-0.45
Spectral Laplacian ISE means	0.10-0.42
Bayes	0.01-0.60

We see from Table 8.1 that for the Gaussian kernel the broadest range of bandwidths is achieved using the the Laplacian ISE classifier. All classifiers perform well in the kernel size range, 0.30-0.42. The standard ISE classifier seems to start working properly at a slightly larger kernel size then the others. The spectral versions of the ISE classifier performs well, but seems to have a little narrower bandwidth range, compared with the classifiers working implicitly in a Mercer space.

The shape of the unweighted and weighted density estimates using a Gaussian kernel with bandwidth 0.80 is given in Fig. 8.2. Notice that the shape of the top figure seems to be dominated by the dense Gaussian distribution in the center, and all classifiers using this density estimate are unable to separate the two classes clearly. The weighted data points used in the bottom figure reduce the dominant shape of the Gaussian distribution enough, compared to the more sparsely distributed circle shape, to let the weighted Laplacian ISE classifier correctly classify all test samples.

In Fig. 8.3 the shape of the unweighted and weighted density estimates using a Gaussian kernel with bandwidth 0.30 is illustrated. Compared with the shapes in Fig. 8.2, the circle distribution is now good enough separated from the Gaussian distribution to let all classifiers correctly classify all of the test samples. In the bottom figure with weighted data, we clearly see the structure where the circle distribution is emphasized.

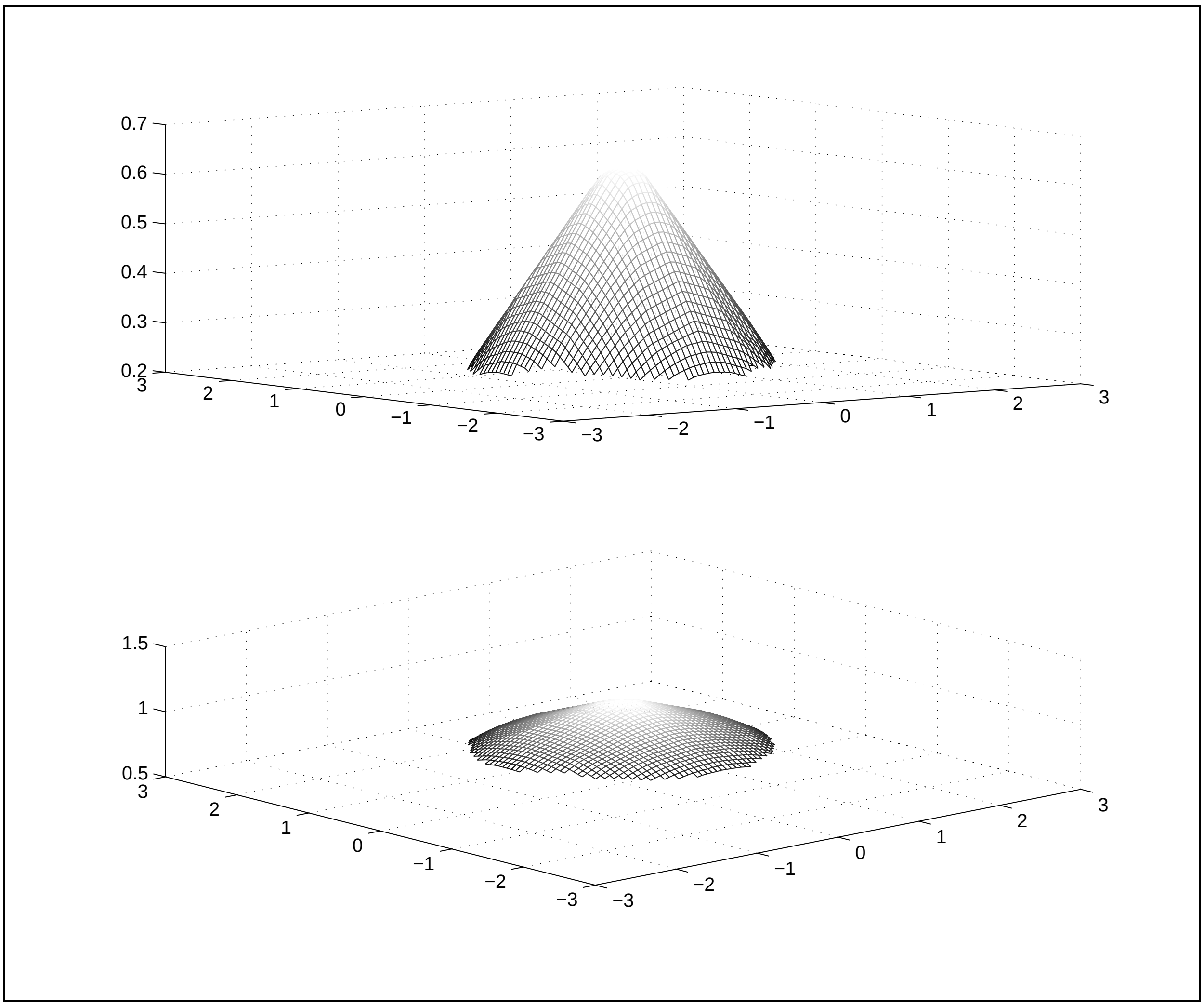

Figure 8.2: Shapes of density estimates using Gaussian kernel with bandwidth, 0.80. Unweighted estimate in the upper plot. Laplacian weights applied to the points in the estimate in the lower plot. This kernel width gives a 100% classification rate only for the Laplacian ISE classifier.

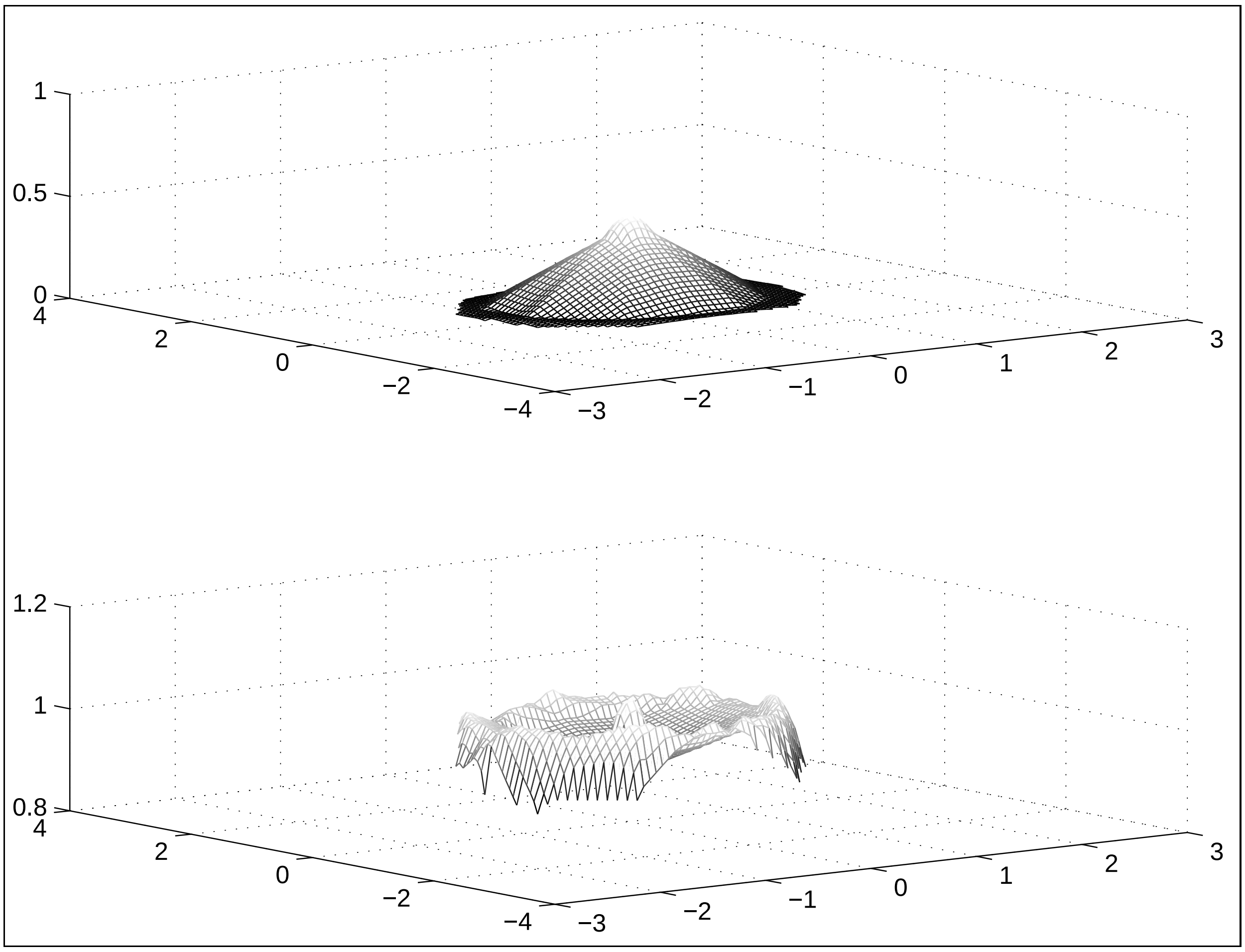

Figure 8.3: Shape of density estimate using Gaussian kernel with bandwidth, 0.30. Unweighted estimate in the upper plot. Laplacian weights applied to the points in the estimate in the lower plot. This kernel width gives a 100% classification rate for all classifiers.

Table 8.2: Epanechnikov kernel

Classifier	bandwidth range
Standard ISE	0.09-2.23
Laplacian ISE	0.08-1.94
Spectral ISE means	0.02-1.27
Spectral Laplacian ISE means	0.24-1.30
Bayes	0.08-1.39

We know that even if the Epanechnikov kernel does not map the data to a Mercer space, it is known to give good density estimates. From Table 8.2 we see that all classifiers work well over an even wider range of kernel sizes then for the Gaussian kernel. Again we note that the implicit versions of the ISE classifier seems to have a wider kernel bandwidth range then the spectral classifiers. This time the standard ISE classifier has the widest kernel bandwidth range with 100% classification rate.

Table 8.3: Square kernel

Classifier	bandwidth range
Standard ISE	0.07-1.32
Laplacian ISE	0.11-1.32
Spectral ISE means	0.02-0.91
Spectral Laplacian ISE means	0.23-1.18
Bayes	0.11-1.06

The results using a square kernel in Table 8.3 are similar to the previous, but the bandwidth ranges are generally smaller then for the Gaussian and Epanechnikov kernel. Again the standard ISE classifier has the widest kernel bandwidth range with 100% classification rate.

Summary We note that for this artificial data set, the Epanechnikov kernel seems to have the broadest range of kernel bandwidths for all different versions of the ISE classifier. This may be useful when we don't know, or are unable to use cross-validation on our training data to select the optimal kernel size. The Gaussian kernel also produces good results. The simple square kernel works well, but it has the smallest range of usable bandwidths. Even if the Epanechnikov and square kernel are non-Mercer kernels, and thus does not map to a Mercer space, it seems possible to find estimates of ISE divergence in the spaces they map to. The Bayes classifier works well over similar bandwidth ranges to the ISE classifiers.

Chapter 9

Classification experiments

9.1 Introduction

This part of the thesis reports experiments done with the different versions of the ISE classifier discussed in Part I. Because of the similarity with the well-known Bayes classifier, this is the main classifier which we choose to compare the results with. Some experiments also include results using other Mercer space based classifiers, particularly the SVM and the Laplacian classifier.

To reduce numerical errors when calculating inner-products, affinity matrices and density estimates, we removed the scaling h^{-d} from Eq. (3.12) on page 17 in front of all kernel evaluations with bandwidth h and data dimensionality d.

The purpose of the experiments is to see how different implementations of the ISE classifier behaves on some popular benchmark data sets. In the derivation of the standard ISE classifier we noted the standard ISE classifier may favor the large entropy class compared to the Parzen window based Bayes classifier. Theoretically, the ISE classifiers has a separating hyperplane in the kernel space, that is shifted away from the class with highest entropy[1] compared with the separating hyperplane for the Bayes classifier. We want to check if the standard ISE classifier tends to favor the classes with high entropy compared to the Bayes classifier on some real data sets. This is why we include the confusion matrices and calculated information potentials. We also want to find out what happens when the different ISE classifiers are unable to achieve high classification rates, by looking at how the some of the training data and samples are projected to approximated Mercer spaces.

9.2 Selection of data sets and classification methods

The data sets used in this study are selected from the UCI-repository [16] and the Rätsch [20] data sets. The selected Rätsch data sets are Banana(2,400,4900), Thyroid(5,140,75), Ringnorm(20,400,7000) and Twonorm(20,400,7000), where the numbers in parenthesis are

[1]When we refer to entropy and entropy estimates in this chapter, we mean the Renyi quadratic entropy estimate.

the dimensionality, the size of the training data set and the test data set, respectively. Each set has 100 realizations. To reduce the computation time for the Ringnorm and Twonorm data sets we pick 500 samples from each of the 100 realizations as test data instead of 7000 samples. The data sets have zero mean and unit standard deviation for each feature. The results in the tables and confusion matrices are average classification results and standard deviations when classifying each of the 100 test realizations. For all Rätsch data sets we have used the training set for training and the test sets for testing.

The selected UCI data sets are Wine(13,178), Iris(4,150), WBC(30,569) (Wisconsin breast cancer), Ionosphere(34,351), Pima(8,768) and Pendigits(16,1091). The numbers in parenthesis are the dimensionality, and the number of samples. The Pendigits data consists of samples representing integers 0,1 and 2 selected from the original test data set (3498 samples). For all UCI data sets we have normalized the standard deviation to one for each feature, since the classifiers use spherical kernel functions, and to use a method comparable to the one used in the Laplacian classifier, described in [12]. All UCI test and training data sets were created by splitting a random permutation of the data set in two halves over 100 trials, where 1/3 of the data set was used for testing and 2/3 used for training.

All the UCI and Rätsch data sets are also classified with our implementation of a Parzen window based Bayes classifier, for comparison. We refer to the Parzen window based Bayes classifier simply as the Bayes classifier from now on. In all experiments we have found the best kernel size using three-fold cross validation over a range of kernel sizes on the training data set, and selected the kernel with highest classification rate.

In the following sections we will present and discuss results obtained on the selected data sets, using the standard ISE, Laplacian ISE, spectral ISE and spectral Laplacian ISE classifiers. Unless otherwise specified we have chosen to use a Gaussian kernel in the calculations. This is because it is a Mercer kernel, and when using cross validation in training to find the best kernel size it does not matter much which kernel type we use.

9.3 Standard ISE

In this section we discuss results found when using the standard ISE classifier discussed in Chapter 5. Classification results found with our implementation of the Bayes classifier, SVM results obtained from [21] and results found using the Laplacian classifier abbreviated with CS, described in [12] are also included for comparison. The Laplacian classifier is trained in the same way as we have done. The SVM used was trained to find the parameters C and σ (C is the regularization constant and σ the width of the RBF kernel used) with a five-fold cross validation on five realizations of each data set [21].

If one class from a data set has a small ratio of information potential compared to the other classes, this class will have a relatively large entropy. We estimate the information potentials for each class and compare the confusion matrices for the standard ISE and the Bayes classifier, to find out if the standard ISE classifier tend to classify more samples to the classes with large entropy.

Table 9.1 contains classification rates in percent with standard deviations for the Rätsch data sets. The standard ISE classifier performs well, and similar to the other classifiers, with an exception for the Ringnorm data set, where it has a low classification rate. We note that the only case where the standard ISE classifier is able to beat our Bayes classifier is the Twonorm data set. The Banana set seems to be difficult for all classifiers.

Data	Bayes	ISE	SVM	CS
Banana	87.7±1.2	87.6±0.9	89.2±0.7	89.4±0.5
Ringnorm	96.9±0.8	76.6±17.5	98.3±0.1	No data
Twonorm	97.1±0.7	97.2±0.7	97.0±0.6	97.4±0.2
Thyroid	95.5±2.3	94.9±2.5	95.2±2.2	95.7±2.2

Table 9.1: Average classification rates for Rätsch data sets

In Table 9.2 the ratios of information potentials for each of the classes in the Rätsch data sets are listed. The ratios with bold fonts, represent the classes with relatively highest entropy.

Data	ω_1	ω_2
Banana	**45.7%**	54.3%
Ringnorm	**49.3%**	50.7%
Twonorm	**49.7%**	50.3%
Thyroid	**16.6%**	83.4%

Table 9.2: Class data information potentials for Rätsch data sets

Table 9.3 list the average classification rates in percent, with standard deviations for the selected UCI data sets in the same way as in Table 9.1. For the Wine, Iris and Pendigits data sets, the standard ISE classifier performs slightly worse than the Bayes classifier. The standard ISE classifier is slightly better than the Bayes for the Pima and WBC data set, and notably better for the Ionosphere data set.

Data	Bayes	ISE	SVM	CS
Wine	96.6±2.3	95.4±2.4	97.5±1.7	97.3±1.4
Iris	94.3±3.0	93.0±3.3	95.7±2.0	94.5±2.1
WBC	95.9±1.4	96.3±1.2	96.9±0.7	97.1±0.7
Ionosphere	86.3±2.8	94.1±2.3	94.1±1.2	92.5±1.7
Pendigits	99.0±0.5	98.2±0.6	99.6±0.2	98.9±0.4
Pima	71.8±2.4	72.6±2.4	76.8±1.5	73.9±1.7

Table 9.3: Average classification rates for UCI data sets.

In Table 9.4 the estimated ratios of class information potentials are listed. The ratios in bold fonts are the classes with lowest IP, and thus highest entropy. We will later in this section use the confusion matrices for the standard ISE and Bayes classifiers for these data sets to see if the standard ISE classify more points erroneously to the class with highest entropy, as suggested in Section 5.3.1

Data	ω_1	ω_2	ω_3
Wine	34.5%	**26%**	39.5%
Iris	42.6%	31.6%	**25.7%**
WBC	71.8%	**28.2%**	
Ionosphere	58.1%	**41.9%**	
Pima	55.1%	**44.9%**	
Pendigits	30.0%	**28.6%**	41.4%

Table 9.4: Average class information potentials for UCI data sets

9.3.1 Confusion matrices Standard ISE and Bayes

In this section we compare the confusion matrices obtained by classifying with the standard ISE and the Bayes classifier. We want to check if there is any correspondence between the class entropies and which class has most classification errors, when comparing the standard ISE and the Bayes classifier.

About the confusion matrices Each row of a confusion matrix denote the correct class label, while each column denote the predicted label from the classifier. As an example, element (1,2) of a confusion matrix contains the amount of samples that belong to class 1, but are predicted to belong to class 2. The trace of a confusion matrix contains the amount of correctly estimated samples.

Banana

In Table 9.5 we note that the standard ISE classifies more points to class ω_1, with relatively larger entropy, than the Bayes classifier.

ISE	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	1941.6	252.9
ω_2	285.5	2420.0

Bayes	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	1874.7	319.8
ω_2	201.3	2504.2

Table 9.5: Average confusion matrices Banana data set.

Ringnorm

We note in the confusion matrices for the Ringnorm data set in Table 9.6 that the standard ISE has a much lower classification rate than the Bayes classifier. For this data set, class ω_1 has slightly larger entropy than class ω_2. This does not cause the standard ISE to classify more points to class ω_1, compared with the Bayes classifier.

ISE	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	132.19	115.97
ω_2	0.84	251.00

Bayes	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	236.25	11.91
ω_2	3.42	248.42

Table 9.6: Average confusion matrices Ringnorm data set.

Twonorm

For the Twonorm data set, class ω_1 has a slightly higher entropy than class ω_2. The standard ISE classifier should now classify more points to class ω_1 than the Bayes classifier. From Table 9.7 we see that this is not the case.

ISE	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	243.20	6.25
ω_2	7.75	242.80

Bayes	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	243.12	6.33
ω_2	8.20	242.35

Table 9.7: Average confusion matrices Twonorm data set.

Thyroid

The Thyroid data set has a relatively large entropy difference between the classes, with relatively large entropy in class ω_1. In Table 9.8 we note that slightly more points are classified to class ω_1 for the standard ISE classifier than for the Bayes classifier.

ISE	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	20.45	1.86
ω_2	1.94	50.75

Bayes	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	20.19	2.12
ω_2	1.26	51.43

Table 9.8: Average confusion matrices Thyroid data set.

Wine

For the Wine data set in Table 9.9, the standard ISE classifier seems to take a small amount of samples from class ω_1 and ω_3 and classify to class ω_2, compared with the Bayes classifier. From Table 9.4 we see that class ω_2 has largest entropy of the classes.

ISE	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	19.72	0.15	0
ω_2	1.05	21.68	1.29
ω_3	0	0.25	15.86

Bayes	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	19.81	0.03	0
ω_2	1.10	21.89	0.80
ω_3	0	0.10	16.27

Table 9.9: Average confusion matrices Wine data set.

Iris

For the Iris data set in Table 9.10, the standard ISE classifier takes a small amount of samples from class ω_1 and ω_2 and classify as ω_3, compared with the Bayes classifier. From Table 9.4 we see that class ω_3 has largest entropy.

ISE	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	16.11	0.01	0.44
ω_2	0	15.53	1.16
ω_3	0	1.91	14.84

Bayes	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	16.76	0.17	0
ω_2	0	15.60	1.12
ω_3	0	1.5700	14.78

Table 9.10: Average confusion matrices Iris data set

WBC

For the WBC data set in Table 9.11, class ω_2 has relatively much larger entropy than class ω_1. The standard ISE has a higher classification rate than the Bayes classifier for this data set, but it does not seem to move samples from the class with relatively lower entropy.

ISE	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	65.98	4.44
ω_2	2.59	117.99

Bayes	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	65.19	5.89
ω_2	1.90	118.02

Table 9.11: Average confusion matrices WBC data set

Ionosphere

From Table 9.4 we note that class ω_2 has larger entropy than class ω_1, and the standard ISE classifier in Table 9.12 seems to add more samples to class ω_2 when compared with the Bayes classifier. For this data set this gives a higher classification rate for the standard ISE classifier.

ISE	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	71.55	3.41
ω_2	3.53	38.51

Bayes	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	73.39	1.40
ω_2	14.68	27.53

Table 9.12: Average confusion matrices Ionosphere data set

Pima

For the Pima data set in Table 9.13 we note that the standard ISE classifier has more samples assigned to class ω_2 with largest entropy, than the Bayes classifier. Again this gives the standard ISE classifier a slightly higher classification rate than our Bayes classifier.

ISE	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	123.09	42.90
ω_2	27.17	62.85

Bayes	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	138.50	27.55
ω_2	44.74	45.21

Table 9.13: Average confusion matrices Pima

Pendigits

For the selected classes from Pendigits, we know from Table 9.4 that class ω_2 has largest entropy. Thus we expect the standard ISE classifier to classify more points from class ω_1 and class ω_3 to class ω_2 than the Bayes classifier. In Table 9.14 we see that this is the case for class ω_1, but not class ω_2.

ISE	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	121.09	1.00	0
ω_2	0.22	116.14	4.90
ω_3	0.07	0.49	121.09

Bayes	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	121.58	0	0
ω_2	0	118.89	2.99
ω_3	0	0.58	120.96

Table 9.14: Average confusion matrices Pendigits

9.3.2 Summary of classification results for the standard ISE classifier

The standard ISE classifier seems to perform very well and similar to our Bayes classifier, but is only able to beat our implementation of a surprisingly good Bayes classifier for 4 out of 10 data sets. The Ringnorm data set is the only set where the standard ISE performs notably worse than the Bayes classifier. For WBC, Ionosphere, Pima and Twonorm the standard ISE classifier has slightly higher classification rates than the Bayes classifier. For each data set we compared the difference in entropy between the classes and the confusion matrices for the standard ISE and Bayes classifier. If we exclude the Ringnorm data set, the difference in entropy seems to let the standard ISE draw points from low entropy classes to higher entropy classes for the Wine, Iris, Ionosphere, Pendigits, Pima, Banana and Thyroid data sets (7 of 9 data sets), compared with the Bayes classifier. For the Twonorm and WBC data sets this was not the case. It is hard to explain exactly why the standard ISE classifier doesn't behave as expected for some data sets.

9.4 Laplacian ISE

In this section we present average classification rates using the version of the ISE classifier discussed in Chapter 6, operating implicitly in the Mercer space spanned by the eigenvectors of the Laplacian matrix. The weighting of the training data samples changes the class entropies, so we choose not to compare the different Laplacian ISE confusion matrices with the Bayes confusion matrices. The average confusion matrices for the data sets using the Laplacian ISE classifier may be found in the appendix. For some selected data sets we also plot the weights for the training data in each class, to check if some class samples are weighted more than others. If some classes have uniform weights for the training samples, this implies that the Laplacian ISE classifier should give the same results as the standard ISE classifier.

In Table 9.15 we list the average classification rates with standard deviations in percent, using the Laplacian ISE classifier on the selected Rätsch data sets. To reduce the computation time we reduced the amount of training and test data to 100 and 50 samples in each realization for Banana, Ringnorm and Twonorm for this classifier. The listed results should still give a good indication of the classifier performance. On these data sets the weighting induced by the Laplacian ISE does not seem to influence the results significantly in a positive manner, compared to the standard ISE classifier.

Data	Rate
Banana	86.2±5.0
Ringnorm	76.7±22.1
Twonorm	96.3±2.7
Thyroid	94.6±2.9

Table 9.15: Average Laplacian ISE classification rates for Rätsch data sets.

In Fig. 9.1 we illustrate the two classes in a typical training set for the Banana data set. The points marked with star symbols represent some of the largest weights in the data set. The classes in this data set are very difficult to separate because they are distributed in several overlapping clusters. The points representing the largest weights seems to be situated in the outer borders of the data set.

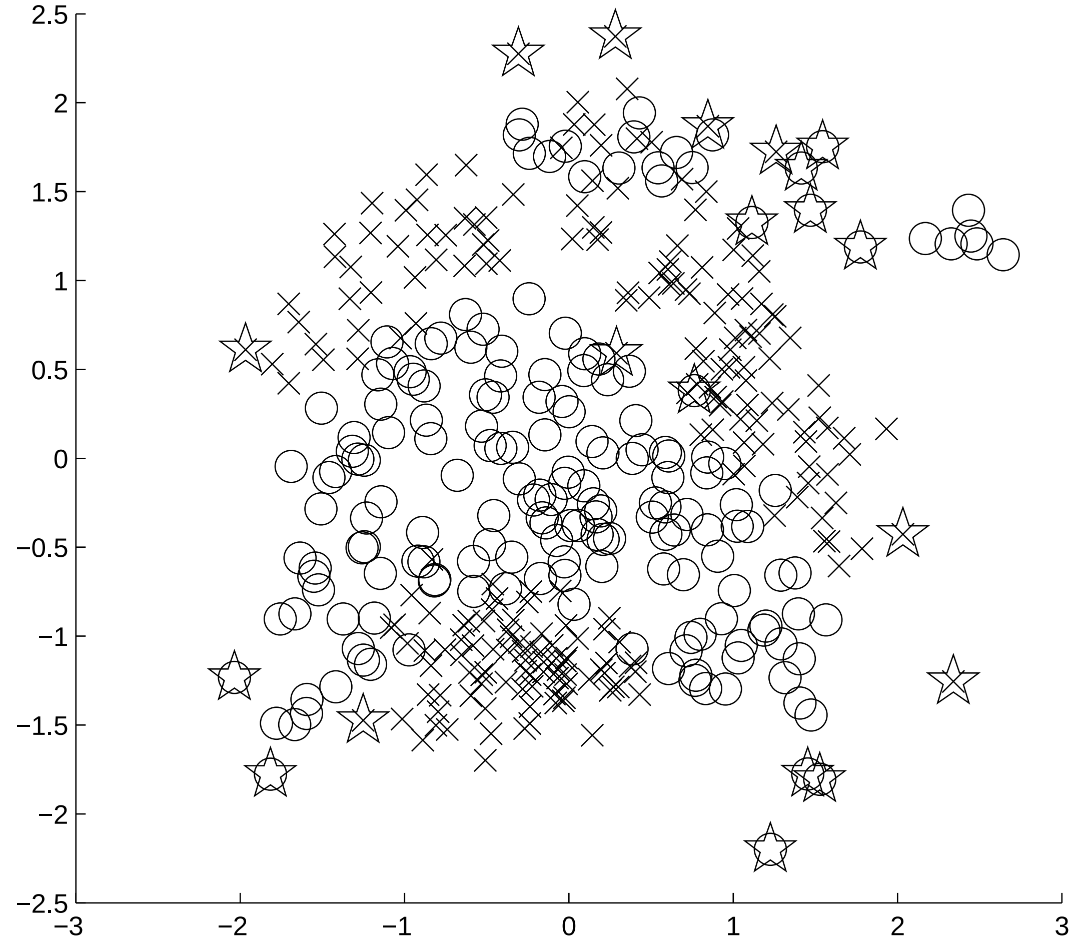

Figure 9.1: Plot of a sample training data set from the Banana data set. Class ω_1 and ω_2, are illustrated with $\times$ and $\bigcirc$ respectively. Some of the points with largest weights are plotted with star symbols.

In Fig. 9.2 we illustrate the weights for a typical training data set from the Banana data set. Each figure illustrate the weight assigned to each sample within a class. We note that of the most of the samples are weighted similarly, but for a few samples in each class the weights are much larger than the others, e.g the sample with a weighing of 400 in class ω_2 compared to most of the others which seem to have weights in the interval 10-100. The largest weights in this figure corresponds to the star symbols in Fig. 9.1. The weighting of data points does not seem improve the classification rate on this data set.

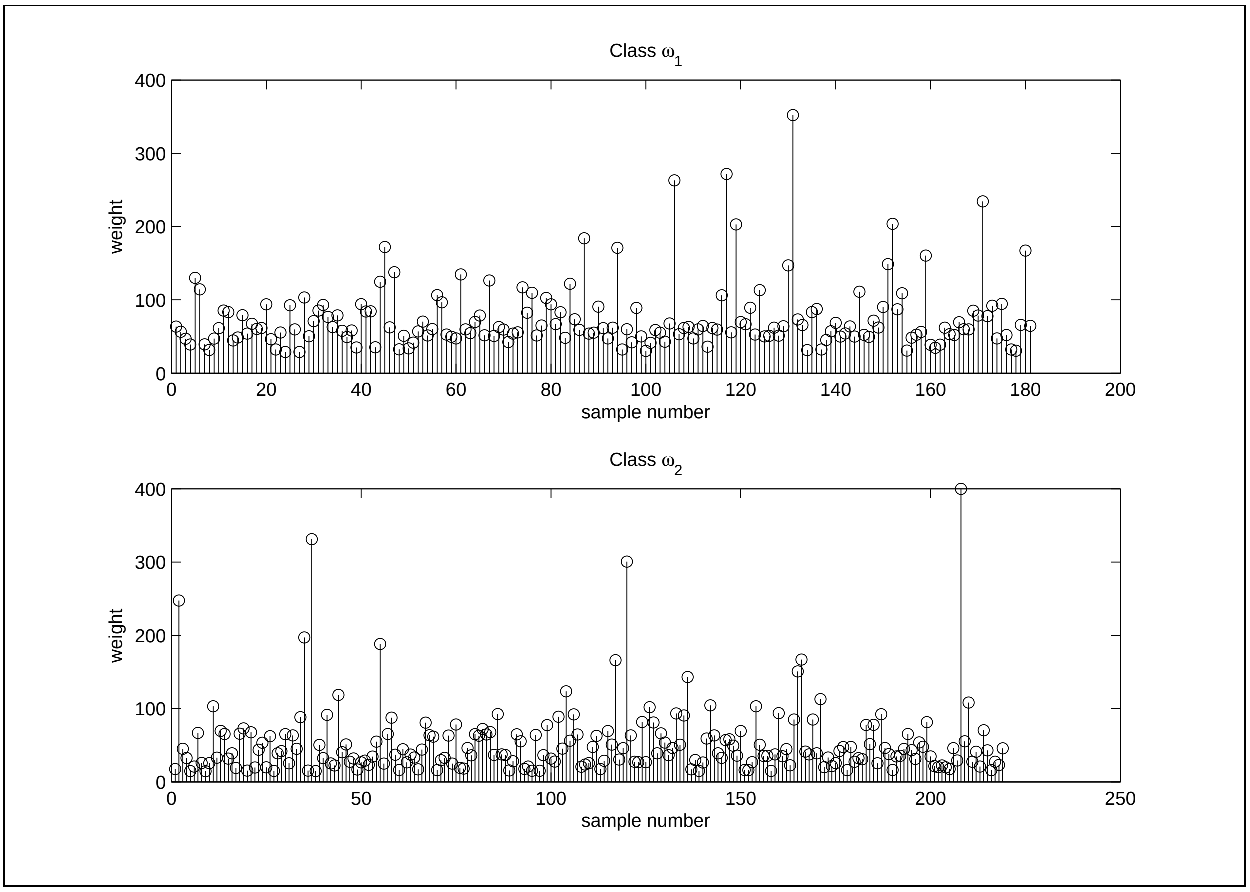

Figure 9.2: Sample weights for the Banana training data set. Top and bottom figures illustrate typical weights for training samples from class ω_1 and ω_2, respectively.

In Table 9.16 we list the average classification rates with standard deviations, using the Laplacian ISE classifier on the selected UCI data sets. We note that the results are very similar to the standard ISE rates. For Iris, Wine, WBC and Pendigits we achieve slightly higher classification rates with the Laplacian ISE classifier than for the standard ISE classifier. The Ionosphere data set has notably worse classification rates with the Laplacian ISE classifier, 89.0% versus 94.1% for the standard ISE classifier. The Pima data set has slightly worse classification rate for the Laplacian ISE classifier versus the standard ISE classifier.

Data set	ISE
Wine	95.6±2.1
Iris	94.1±2.6
WBC	96.6±1.2
Ionosphere	89.0±2.4
Pendigits	98.5±0.6
Pima	72.4±2.4

Table 9.16: Average classification rates using the Laplacian ISE classifier on UCI data sets.

In Fig. 9.3 we illustrate the weights for a typical training data set from the Wine data set. Each figure illustrate the weight assigned to each sample within a class. The weights seem to emphasize some of the points within each class a bit more than the others, but mostly they are quite similar. For this data set the Laplacian induced weighting seems to give a slightly better classification rate.

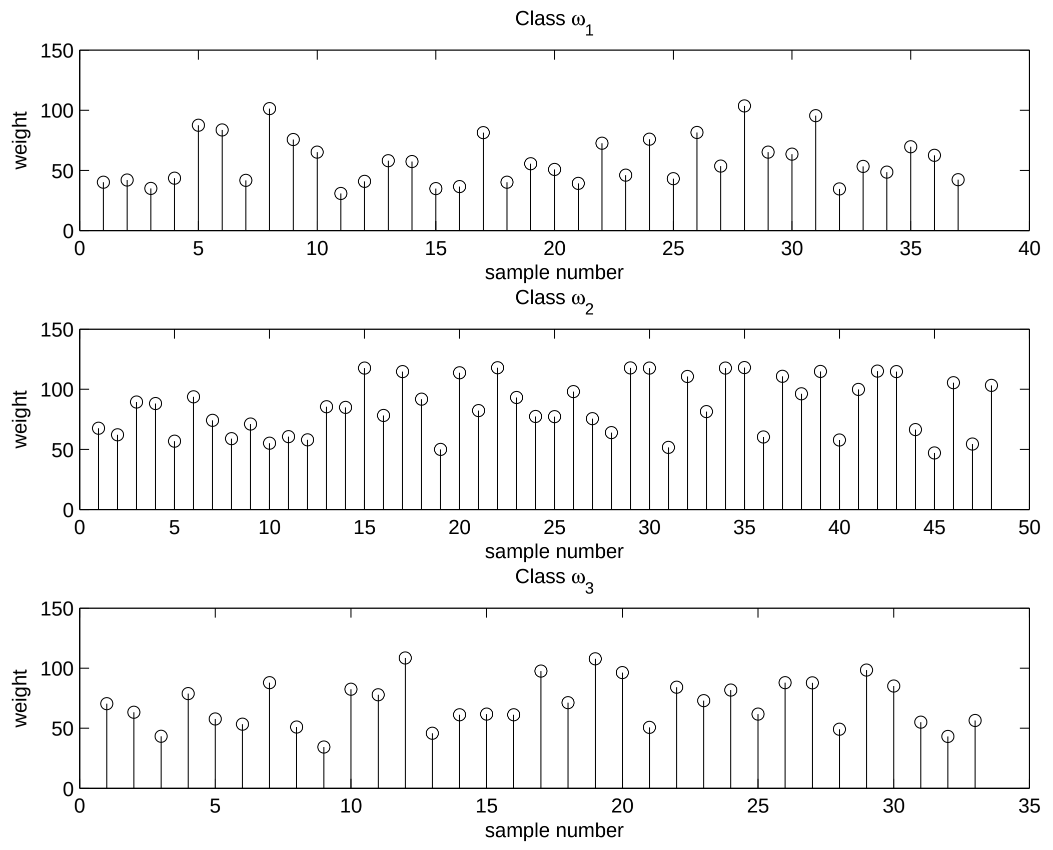

Figure 9.3: Sample weights for the Wine training data set. Top, middle and bottom figures illustrate typical weights for training samples from class ω_1, ω_2 and ω_3, respectively.

In Fig. 9.4 we illustrate the weights for a typical training data set from the Iris data set. Each figure illustrate the weight assigned to each sample within a class. We note that for this training data set each class have a few points that have relatively large weights compared to the others.

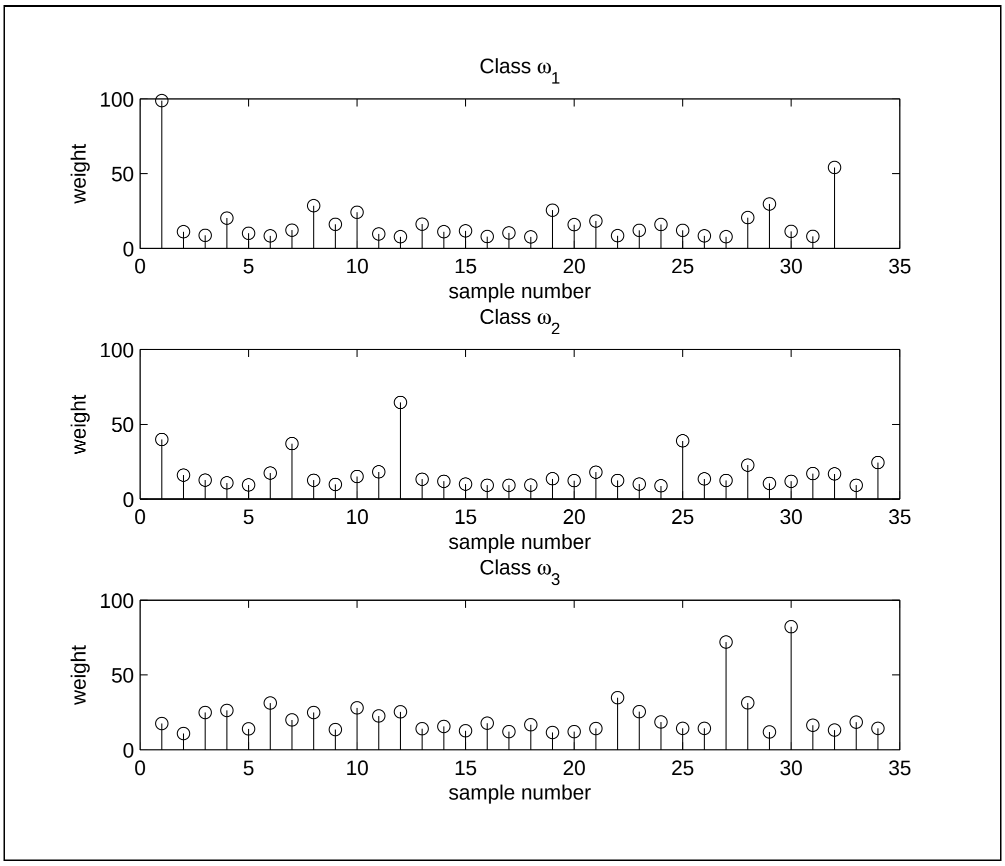

Figure 9.4: Sample weights for the Iris training data set. Top, middle and bottom figures illustrate typical weights for training samples in class ω_1, ω_2 and ω_3, respectively.

In Fig. 9.5 we plotted dimension 2 vs dimension 3 of the same training set used in Fig. 9.4. The 15 points with the largest weights are marked with a star symbol. The points with the largest weights all seem to lie in the border of a class cluster. Emphasizing these points seems to increase the separability of the different classes since the Laplacian ISE classifier performs slightly better than the standard ISE classifier for this data set.

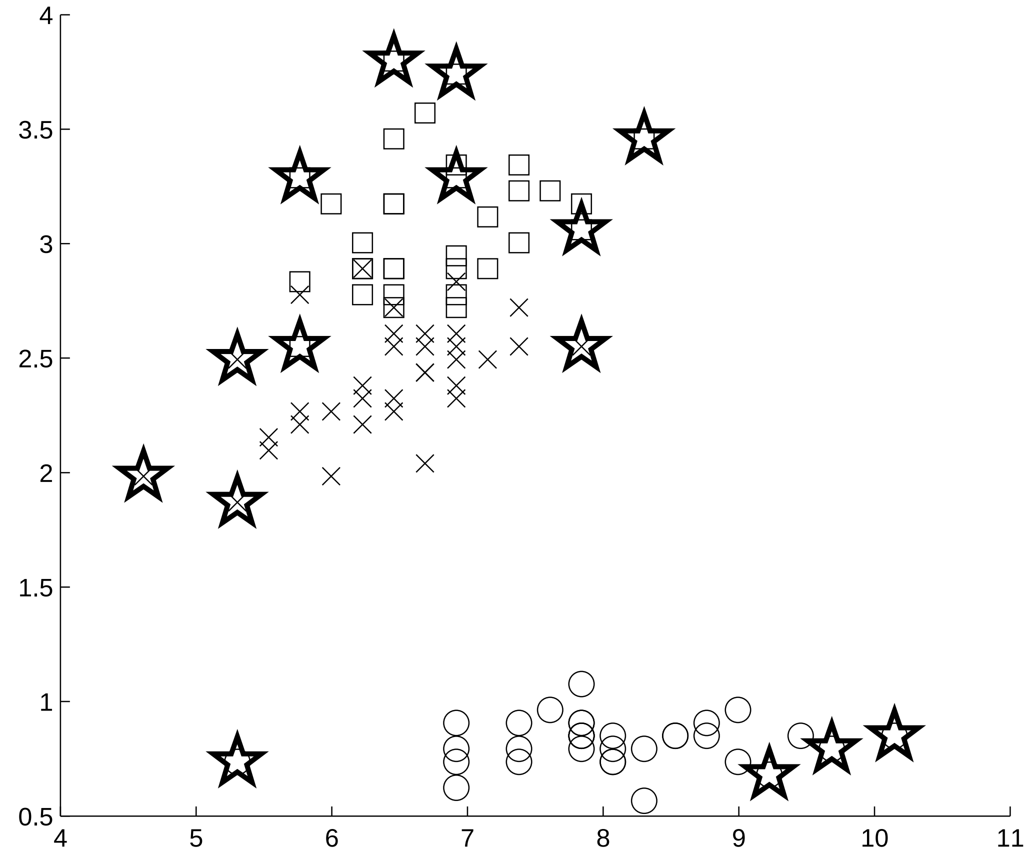

Figure 9.5: Illustration of a typical Iris training data set. We plot dimension 2 versus 3 with the 15 points with largest weights marked with star symbols. Class ω_1, ω_2 and ω_3 samples are illustrated with $\bigcirc$, $\times$ and $\square$, respectively.

9.4.1 Summary of classification results for the Laplacian ISE classifier

We note that weighting the training data as described in Chapter 6 actually gave slightly worse results for the Rätsch data set. This may be because the training data comes from classes with much overlap between the borders of the class clusters, as illustrated for the Banana data set in Fig. 9.1. This seems to give largest weights to the points in the outer border of the whole data set, and may not increase the separability of the classes. For the UCI data sets, we get slightly better classification rates, except for the Ionosphere and Pima data sets. Looking at the samples in dimension 2 versus 3 for the Iris data set in Fig. 9.5 the largest weights seem to be in the different cluster borders of quite separable classes. This may indicate that the Laplacian induced weighting of data helps the classifier when the class borders are not overlapping, but it is hard to conclude from the few data sets we have looked at.

9.5 Spectral ISE

In this section we present results using the spectral ISE classifier discussed in Chapter 7. We include the classification rates using the mean vectors in the approximate Mercer kernel space given by the principal eigenvectors of the training data affinity matrix. The results with median vectors are very similar, indicating that the data sets have few outliers. Classification results and confusion matrices using the median version of the spectral ISE classifier are included in the appendix for reference. We also illustrate how some of the data sets with low classification results are projected to the approximate Mercer kernel space. The illustration of mappings where the spectral classifiers performs well are postponed to the discussion of the spectral Laplacian ISE classifier in the next section.

In Table 9.17 we list the average classification rates with standard deviations using the mean version of the spectral ISE classifier on selected Rätsch data sets. We note that for the Banana data set this classifier seems to fail and that the Ringnorm data set has slightly higher classification rates than the standard ISE classifier. The other data sets have classification rates slightly lower, but similar to the standard ISE classifier.

Data	Mean rate
Banana	54.2 ± 6.2
Ringnorm	77.3 ± 1.8
Twonorm	97.3 ± 0.6
Thyroid	93.5 ± 2.4

Table 9.17: Average classification rates for some Rätsch data sets using the spectral ISE classifier.

In Fig. 9.6 we illustrate the mapping of the Banana data set projected to the approximate Mercer kernel space given by the two principal eigenvectors of the affinity matrix which gives a classification rate of 54.2%. Note that the training data samples for both classes are clustered together around origo and almost impossible to separate from each other. In Fig. 9.1 in the previous section, we saw that the class clusters for the Banana data set in the input space also were highly overlapping. The test samples illustrated with $*$ symbols are spread far a way from the class mean vectors, but the spectral ISE classifier fails, because no test point is distinctly closer to one of the two class means.

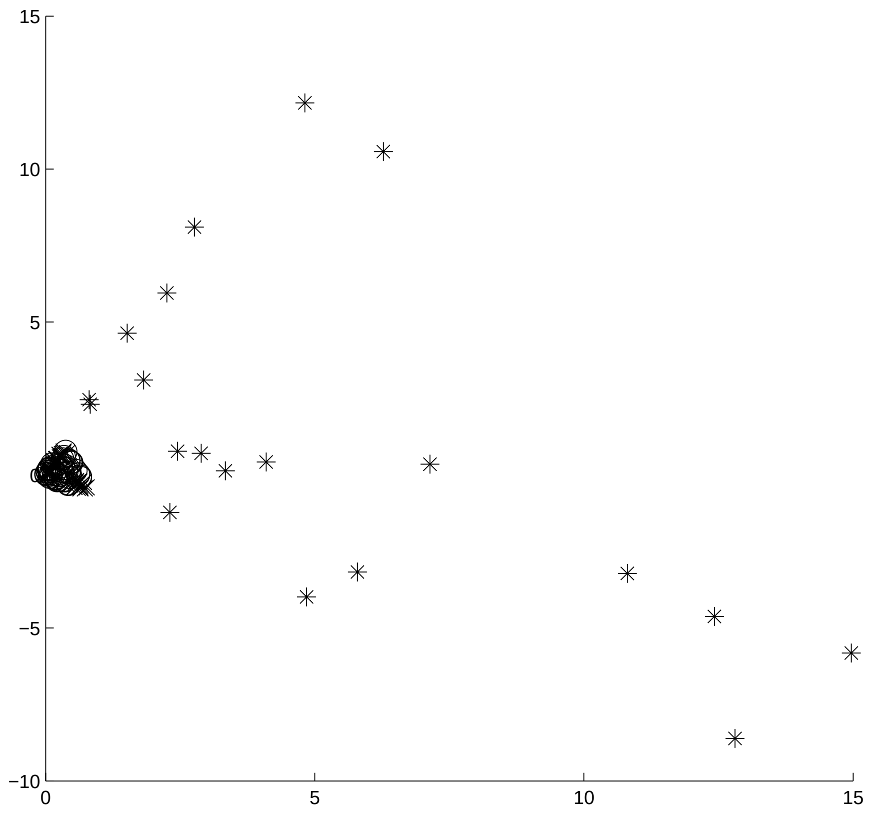

Figure 9.6: Illustration of the data mapping given by the affinity matrix created with training data from the Banana data set. We projected 50 training samples from each of class ω_1 illustrated with $\bigcirc$, and 50 from class ω_2 illustrated with $\times$. Mapping of 20 test samples is illustrated with $*$ symbols.

In Table 9.18 we list the average classification rates with standard deviations using the spectral ISE classifier on selected UCI data sets. We get lower classification rates, but similar to the standard ISE classifier.We illustrate the mapping of data for the Ionosphere and Iris data sets in Fig. 9.7 and Fig. 9.8, respectively.

Data set	Mean rate
Wine	95.1±2.5
Iris	81.1±6.8
WBC	90.0±2.3
Ionosphere	70.6±3.4
Pendigits	84.1±2.1
Pima	69.6±2.5

Table 9.18: Average classification rates for some UCI data sets using the spectral ISE classifier.

In Fig. 9.7 we illustrate the mapping of data projected onto the two principal eigenvectors of the affinity matrix for the Ionosphere data set. We marked the mean points of each class from the training data with large bold symbols. The class means seem to end up close to each other around origo, with test samples spread far away from the class means. This makes it difficult for the spectral ISE classifier to separate the two classes, since the distances from each test point to each class mean are very similar.

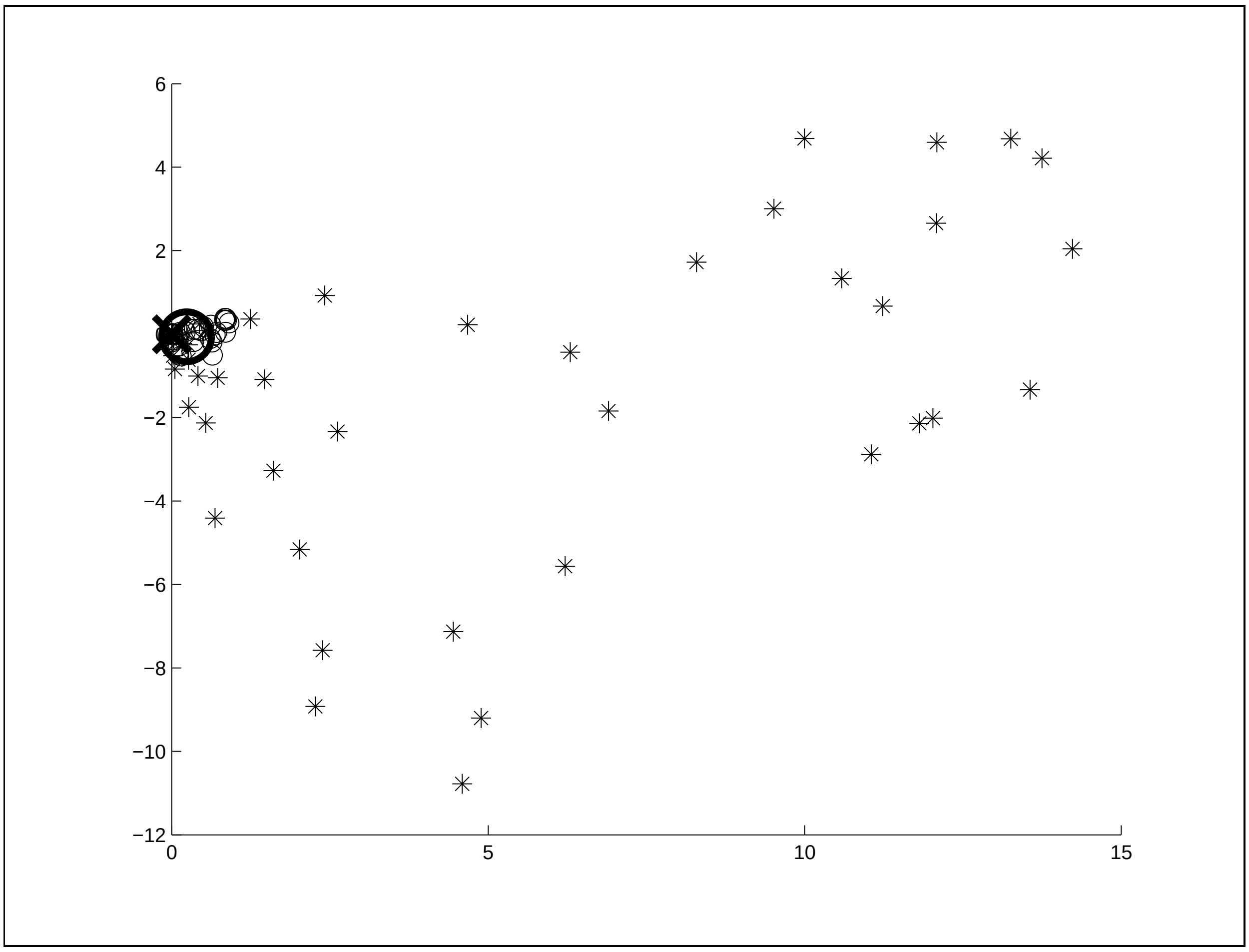

Figure 9.7: Illustration of the data mapping given by the affinity matrix created with training data from the Ionosphere data set. We projected 50 training samples from each of class ω_1 illustrated with $\bigcirc$, and 50 from class ω_2 illustrated with $\times$. Mapping of test samples is illustrated with $*$ symbols.

In Fig. 9.8 we illustrate the mapping of a typical training data projected onto the three principal eigenvectors of the affinity matrix for the Iris data set. We marked the mean points of each class with large bold symbols. Note that the data set is clearly not linearly separable for all classes, and this is probably why the ISE classifier seems to have some problems.

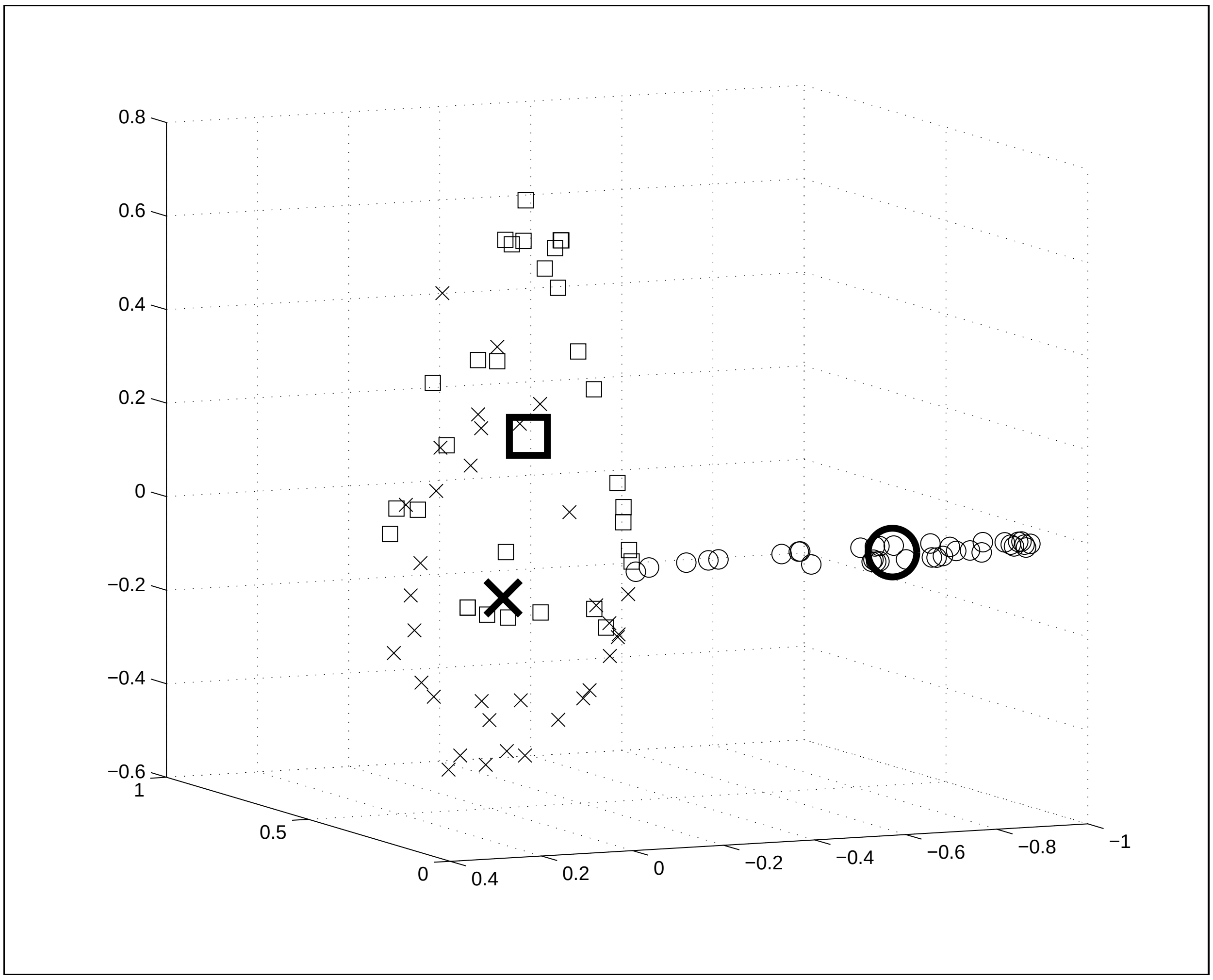

Figure 9.8: Illustration of the data mapping given by the affinity matrix created with training data from the Iris data set. We projected typical training samples from each of class ω_1, ω_2 and ω_3 and illustrated with $\bigcirc$, $\times$ and $\square$ symbols, respectively. The mean points of each class are marked with large bold symbols.

9.5.1 Summary of classification results for the spectral ISE classifier

The results obtained with the spectral ISE classifier are similar to the results obtained with the standard ISE classifier, but with a lower classification rate in most cases. Note that for the Ringnorm and Twonorm data sets the classification rates are slightly better using the spectral ISE classifier compared with the standard ISE classifier. Why the spectral ISE classifier seems to work better for these two data sets is hard to say. The lower classification rates may be explained with the fact that the spectral ISE classifier works in an approximated Mercer space, while the standard ISE classifier works implicitly in a Mercer space. When the spectral ISE classifier fails, it seems to be because almost all training points are mapped to clusters where the mean vectors are close to each other. When a test sample is far away from closely grouped mean vectors, the distance between the test point and each of the class means are very similar, and the classifier seems to be more likely to make an error.

9.6 Spectral Laplacian ISE

In this section we present results using the spectral Laplacian classifier discussed in Chapter 7. For the same reason as previously we focus on the version using the mean vectors in an approximate Mercer space. Results obtained with the median version are included in the appendix along with confusion matrices for both versions. We also illustrate how some of the data sets with good classification rates are mapped to the space spanned by the principal eigenvectors of their Laplacian matrices.

In Table 9.19 we note that the Banana set is difficult to classify correctly when using the spectral Laplacian classifier. The Ringnorm data set is the big surprise here, and we will illustrate in Fig. 9.9 why we achieve so high classification rate for this data set. The Twonorm data set now has a slightly higher classification rate than for the spectral classifier, while the Thyroid has a slightly lower classification rate.

Data	Mean rate
Banana	58.3±5.3
Ringnorm	98.0±0.7
Twonorm	97.5±0.7
Thyroid	92.9±2.3

Table 9.19: Average classification rates for Rätsch data sets, using the spectral Laplacian ISE classifier.

In Fig. 9.9 we illustrate the mapping of a typical training data projected onto the two principal eigenvectors of the Laplacian matrix for the Ringnorm data set. We illustrate the mean points of each class with large bold symbols. Note that each mean point is far from the other. The test samples are mapped along the same line as the training data, and it is easy to see which of the class means most samples are closest to.

In Table 9.20 we list the average classification rates in percent with standard deviations obtained with the mean version of the spectral Laplacian classifier. We note that for Wine and Iris we achieve better results with the spectral Laplacian classifier than with the spectral ISE classifier, so weighting of data points increase the separability of the data in these data sets. For Ionosphere and Pima we get worse results, so weighting the training samples does not give a positive effect for these data sets.

In Fig. 9.10 we illustrate the mapping of training data and some samples for the Wine data set. It seems from the figure that we get three distinct clusters, one for each class with clearly separated mean vectors. The Laplacian ISE classifier is able to assign the test points to the correct mean cluster in most cases, and we achieve high classification rates.

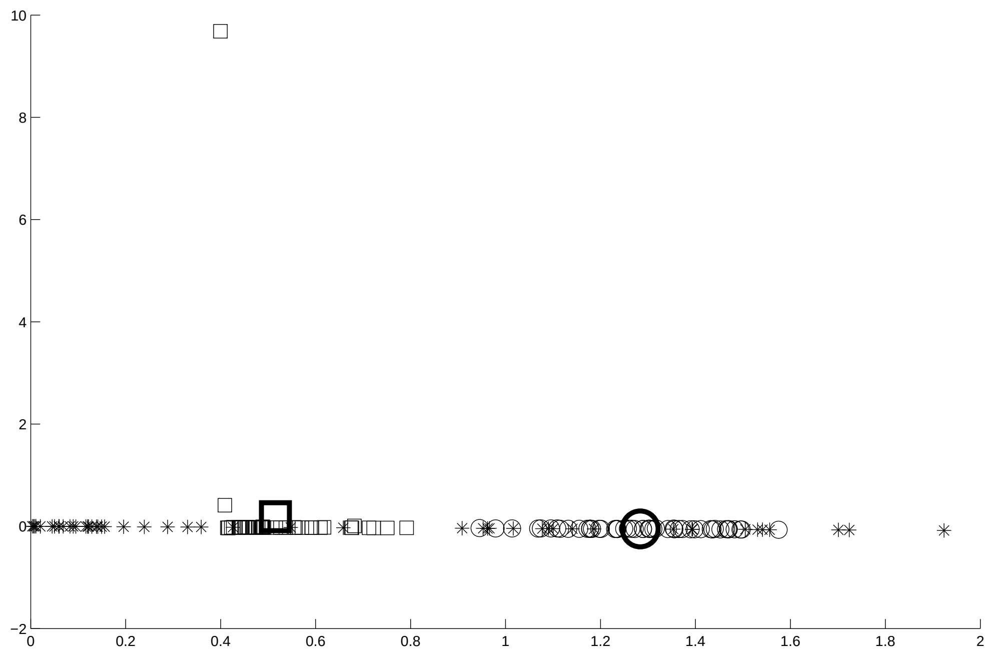

Figure 9.9: Illustration of the data mapping given by the Laplacian matrix created with training data from the Ringnorm data set. We projected typical training samples from each of class ω_1 and ω_2 and and illustrated with $\bigcirc$ and $\Box$ symbols, respectively. The mean points of each class are marked with large bold symbols. Some projected test samples are illustrated with $*$ symbols.

Data	Mean rate
Wine	97.7±1.8
Iris	85.2±4.8
WBC	78.9±2.2
Ionosphere	57.5±10.7
Pendigits	79.4±3.9
Pima	68.3±2.7

Table 9.20: Average classification rates for UCI data sets, using the spectral Laplacian ISE classifier.

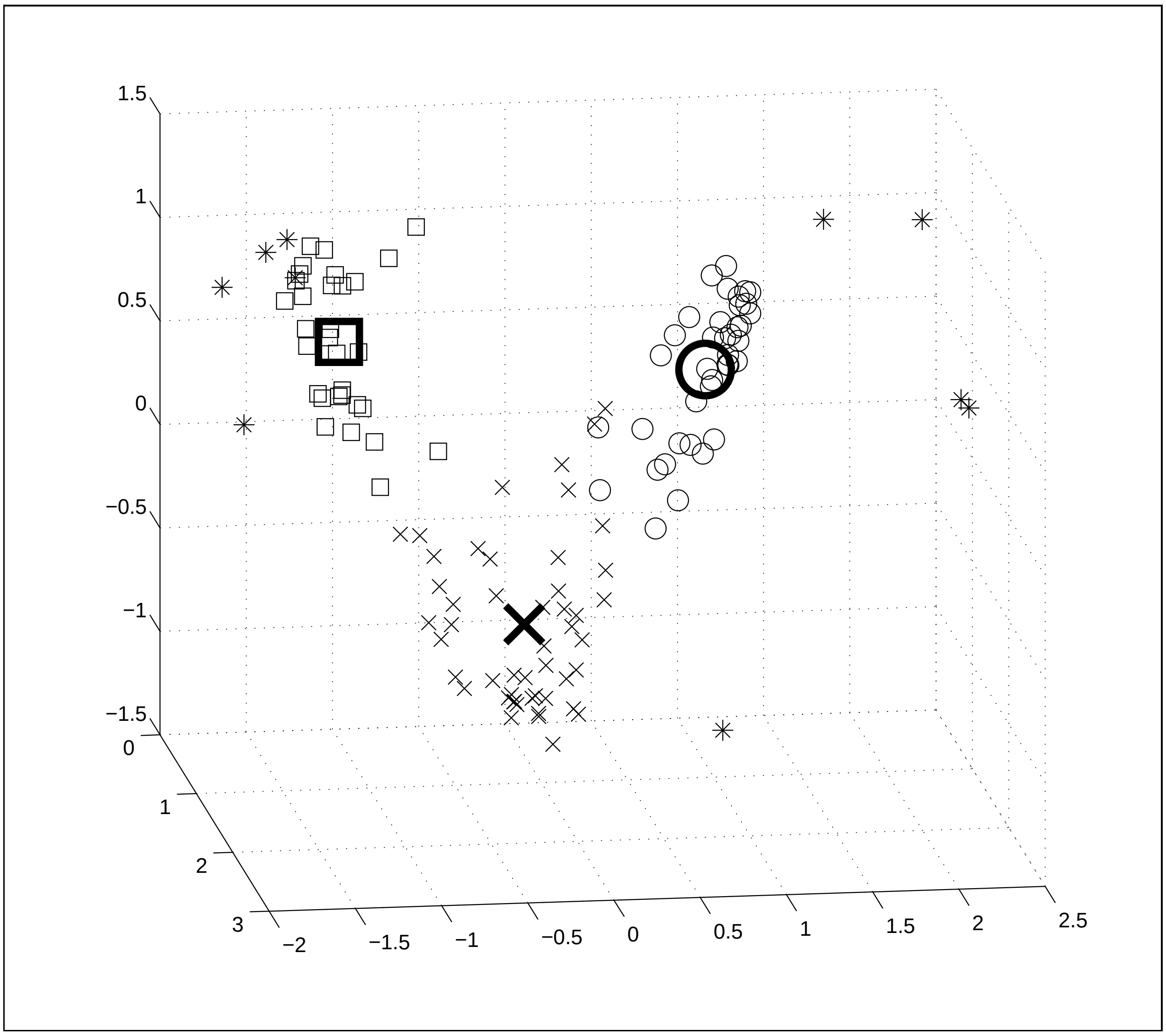

Figure 9.10: Illustration of the data mapping given by the Laplacian matrix created with training data from the Wine data set. We projected typical training samples from each of class ω_1, ω_2 and ω_3 and illustrated with $\bigcirc$, $\square$ and $\times$ symbols, respectively. The mean points of each class are marked with large bold symbols. Some projected test samples are illustrated with $*$ symbols.

9.6.1 Summary of classification results for the spectral Laplacian ISE classifier

We sometimes get better results using the spectral Laplacian ISE classifier than for the spectral ISE classifier, but overall the results are inferior to the other classifiers. Generally all the spectral methods seems to give slightly worse results than the methods working implicitly in Mercer space.

Part III

Conclusion

Chapter 10

Conclusion

In this thesis we have provided a study of many of the relatively new concepts used in information theoretic learning. We also reviewed background information necessary to understand basic density estimation and pattern classification. New classifiers based on the information theoretic ISE divergence measure and kernel methods, using both weighted and unweighted data, are investigated. Relations between an ISE divergence based classifier operating implicitly in a Mercer kernel space and the well known Parzen window based Bayes classifier are studied. We found that by using unweighted data the ISE classifier is comparable to the Bayes classifier with slightly different properties. This classifier seems to prioritize the classes with highest entropy compared to the Bayes classifier on several popular data sets, but not all. We use the spectral properties of the data affinity and Laplacian matrix, to propose and investigate ISE based classifiers working directly in approximated Mercer kernel spaces. We found that in most cases the spectral versions of the ISE classifier perform slightly worse than the versions working implicitly in Mercer spaces.

10.1 Further work

- In this thesis we have used the same single bandwidth kernel size for all classes within a data set. This is fine if the classes have the same type of distribution, but this is probably not a realistic situation. It should be rather simple to extend the classifiers discussed in this thesis to allow for different kernel sizes for each class. We can also check several other Mercer and Non-Mercer kernels to try to find out if some types of data distributions benefit from a particular kernel type.

- It may be interesting to investigate how the classifiers perform with different kernels when the kernel size is selected by some reference rule, e.g. Silverman's rule in Eq. 3.13 on page 17, since we can't always afford to do cross-validation.

- The spectral ISE classifier and the spectral Laplacian classifier both work in spaces spanned by a number of eigenvectors corresponding to the number of classes in each data set. It may be interesting to see if including a few more eigenvectors can improve the classifiers performance.

- The weighting of data points used in the Laplacian ISE classifiers seems to give largest weights to points on the class borders. For large data sets it may be useful to train the classifiers by selecting a small portion of the points with the largest weights, within each class. This may reduce the computation time for the classifiers, without reducing the classification rates significantly.

- Since we have developed spectral classifiers which use the median vectors in an approximated Mercer space, it should be interesting to test how they behave compared to the same classifiers using mean vectors on data sets which clearly contains outliers. An analysis of how the different classifiers behave on data sets with outliers should also be done.

Appendix A

Laplacian ISE classifier

Classification results for the Laplacian ISE classifier not listed in the main part of the thesis.

A.1 Average confusion matrices Laplacian ISE classifier

	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	18.57	4.26
ω_2	2.63	24.54

Table A.1: Average confusion matrix Banana data set.

	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	12.94	11.40
ω_2	0.26	25.40

Table A.2: Average confusion matrix Ringnorm data set.

	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	18.61	3.70
ω_2	0.35	52.34

Table A.3: Average confusion matrix Thyroid data set.

	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	24.66	0.97
ω_2	0.86	23.51

Table A.4: Average confusion matrix Twonorm data set.

	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	66.84	4.02
ω_2	2.44	117.70

Table A.5: Average confusion matrix WBC data set

	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	73.64	1.80
ω_2	11.06	30.50

Table A.6: Average confusion matrix Ionosphere data set.

	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	127.41	39.63
ω_2	31.02	57.94

Table A.7: Average confusion matrix Pima data set.

	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	19.68	0	0
ω_2	1.22	21.20	1.41
ω_3	0	0.04	16.45

Table A.8: Average confusion matrix Wine data set.

	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	16.44	0.20	0.10
ω_2	0	14.96	1.00
ω_3	0	1.63	15.67

Table A.9: Average confusion matrix Iris data set.

	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	121.89	0.04	0.04
ω_2	0.01	116.52	5.37
ω_3	0	0.180	120.95

Table A.10: Average confusion matrix Pendigits data set.

Appendix B

Spectral ISE classifier

Classification results for the spectral ISE classifier not listed in the main part of the thesis.

Data	Median rate
Banana	56.59±5.58
Ringnorm	76.69±1.92
Twonorm	97.28±0.58
Thyroid	93.35±2.70

Table B.1: Average classification rates for some Rätsch data sets using the spectral ISE classifier with median vectors.

Data set	Median rate
Wine	95.5±2.4
Iris	81.3±6.3
WBC	89.2±2.9
Ionosphere	68.9±8.4
Pendigits	83.0±2.0
Pima	69.2±2.3

Table B.2: Average classification rates for some UCI data sets using the spectral ISE classifier with median vectors.

B.1 Average confusion matrices Spectral ISE

Means	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	1431.69	1489.02
ω_2	762.78	1216.51

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	1524.60	1412.56
ω_2	669.87	1292.97

Table B.3: Average confusion matrices for the Banana data set.

Means	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	134.82	113.34
ω_2	0	251.84

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	131.61	116.55
ω_2	0	251.84

Table B.4: Average confusion matrices for the Ringnorm data set.

Means	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	243.73	5.72
ω_2	8.00	242.55

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	243.79	5.66
ω_2	7.95	242.60

Table B.5: Average confusion matrices for the Twonorm data set.

Means	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	19.56	2.15
ω_2	2.75	50.54

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	19.50	2.18
ω_2	2.81	50.51

Table B.6: Average confusion matrices for the Thyroid data set.

Means	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	19.94	1.29	0
ω_2	0	20.61	0
ω_3	0	1.63	16.53

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	19.94	1.2	0
ω_2	0	20.82	0
ω_3	0	1.51	16.53

Table B.7: Average confusion matrices for the Wine data set.

Means	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	15.85	0	0
ω_2	0.06	12.99	3.8
ω_3	0.61	4.02	12.67

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	15.93	0	0
ω_2	0.01	12.9	3.94
ω_3	0.58	4.11	12.53

Table B.8: Average confusion matrices for the Iris data set.

Means	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	60.82	9.51
ω_2	9.6	111.07

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	57.45	7.71
ω_2	12.97	112.87

Table B.9: Average confusion matrices for the WBC data set.

Means	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	46.24	5.33
ω_2	29.06	36.37

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	45.87	6.93
ω_2	29.43	34.77

Table B.10: Average confusion matrices for the Ionosphere data set.

Means	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	113.34	6.27	2.30
ω_2	0	82.40	38.60
ω_3	0	10.87	111.22

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$	$\hat{\omega_3}$
ω_1	113.01	5.12	3.78
ω_2	0	70.65	50.35
ω_3	0	2.64	119.45

Table B.11: Average confusion matrices for the Pendigits data set.

Means	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	130.61	37.03
ω_2	40.75	47.61

Medians	$\hat{\omega_1}$	$\hat{\omega_2}$
ω_1	126.27	41.37
ω_2	37.51	50.85

Table B.12: Average confusion matrices for the Pima data set.

Appendix C

Spectral Laplacian ISE classifier

Classification results for the spectral Laplacian ISE classifier not listed in the main part of the thesis.

Data	Median rate
Banana	58.8±3.9
Ringnorm	98.0±0.7
Twonorm	97.4±1.0
Thyroid	92.7±2.6

Table C.1: Average classification rates for Rätsch data sets, using the spectral Laplacian ISE classifier with median vectors.

Data	Median rate
Wine	98.0±1.9
Iris	85.1±4.8
WBC	78.9±2.2
Ionosphere	61.6±14.8
Pendigits	80.9±2.7
Pima	68.3±2.6

Table C.2: Average classification rates for UCI data sets, using the spectral Laplacian ISE classifier with median vectors.

C.1 Average confusion matrices for the spectral Laplacian ISE classifier.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	1546.59	1396.16
ω_2	647.88	1309.37

Medians	ω_1	ω_2
$\hat{\omega}_1$	1576.08	1401.74
$\hat{\omega}_2$	618.39	1303.79

Table C.3: Average confusion matrices for the Banana data set.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	239.83	1.46
ω_2	8.33	250.38

Medians	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	239.73	1.35
ω_2	8.43	250.49

Table C.4: Average confusion matrices for the Ringnorm data set.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	244.34	5.11
ω_2	7.50	243.05

Median	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	243.71	5.74
ω_2	7.32	243.23

Table C.5: Average confusion matrices for the Twonorm data set.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	19.22	2.21
ω_2	3.09	50.48

Medians	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	19.03	2.20
ω_2	3.28	50.49

Table C.6: Average confusion matrices for the Thyroid data set.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$	$\hat{\omega}_3$
ω_1	19.87	0.51	0
ω_2	0.06	22.89	0.21
ω_3	0	0.62	15.84

Medians	$\hat{\omega}_1$	$\hat{\omega}_2$	$\hat{\omega}_3$
ω_1	19.85	0.39	0
ω_2	0.08	23.04	0.16
ω_3	0	0.59	15.89

Table C.7: Average confusion matrices for the Wine data set.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$	$\hat{\omega}_3$
ω_1	17.11	0	0
ω_2	0.04	12.18	3.28
ω_3	0	4.08	13.31

Medians	$\hat{\omega}_1$	$\hat{\omega}_2$	$\hat{\omega}_3$
ω_1	17.13	0	0
ω_2	0.02	12.15	3.3
ω_3	0	4.11	13.29

Table C.8: Average confusion matrices for the Iris data set.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	49	18.24
ω_2	22.12	101.64

Medians	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	49.09	18.21
ω_2	22.03	101.67

Table C.9: Average confusion matrices for the WBC data set.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	40.16	14.15
ω_2	35.56	27.13

Medians	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	32.69	1.85
ω_2	43.03	39.43

Table C.10: Average confusion matrices for the Ionosphere data set.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	130.10	36.54
ω_2	44.67	44.69

Medians	$\hat{\omega}_1$	$\hat{\omega}_2$
ω_1	130.26	36.38
ω_2	44.66	44.70

Table C.11: Average confusion matrices for the Pima data set.

Means	$\hat{\omega}_1$	$\hat{\omega}_2$	$\hat{\omega}_3$
ω_1	110.75	3.06	6.12
ω_2	0	58.00	65.32
ω_3	0	0.68	121.07

Medians	$\hat{\omega}_1$	$\hat{\omega}_2$	$\hat{\omega}_3$
ω_1	108.35	9.95	1.63
ω_2	0	65.39	57.93
ω_3	0	0.34	121.41

Table C.12: Average confusion matrices for the Pendigits data set.

Bibliography

[1] P. M. Atkinson and P. Lewis, *Geostatistical classification for remote sensing: an introduction*, Computers Geosciences (2000), no. 26, 361–371.

[2] T. M. Cover, *Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition*, IEEE Transactions on Electronic Computers **EC-14** (1965), no. 3, 326–334.

[3] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley series in telekommunications, John Wiley & Sons, Inc, 1991.

[4] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2 ed., John Wiley Sons, Inc, 2001.

[5] D. Erdogmus and J. C. Principe, *From Linear Adaptive Filtering to Nonlinear Information Processing*, IEEE Signal Processing Magazine **23** (2006), no. 6, 14–33.

[6] M. Girolami, *Orthogonal Series Density Estimation and the Kernel Eigenvalue Problem*, Neural Computation (2002), no. 14, 669–688.

[7] R. V. Hartley, *Transmission of information*, Tech. report, Bell Systems Technical Journal, 1928.

[8] T. Hastie, R. Tibshirani, and J.Friedman, *The Elements of Statistical Learning;Data Mining, Inference, and Prediction*, Springer Series in Statistics, Springer, 2001.

[9] R. Jenssen, *An Information Theoretic Approach to Machine learning*, Ph.D. thesis, University of Tromsø, May 2005.

[10] R. Jenssen, D. Erdogmus, K. E. Hild II, J. C. Principe, and T. Eltoft, *Information Cut for Clustering using a Gradient Descent Approach*, Pattern Recognition (2006), no. 40, 796–806.

[11] R. Jenssen, D. Erdogmus, J. C. Principe, and T. Eltoft, *Information Theoretic Angle-Based Spectral Clustering: A Theoretical Analysis and an Algorithm*, Proc. Int'l Joint Conference on Neural Networks (IJCNN2006), July 2006, pp. 4904–4911.

[12] R. Jenssen, D. Erdogmuz, J. C.Principe, and T. Eltoft, *The Laplacian Classifier*, IEEE Transactions on Signal Processing (2005).

[13] R. Jenssen, J. C. Principe, D. Erdogmus, and T. Eltoft, *The Cauchy-Schwarz divergence and Parzen windowing: Connections to graph theory and Mercer kernels*, Journal of the Franklin Institute **343** (2006), no. 6, 614–629.

[14] M.C. Jones, J.S. Marron, and S.J. Sheater, *A Brief Survey of Bandwidth Selection for Density Estimation*, Journal of the American Statistical Association **91** (1996), no. 433, 401–407.

[15] D. Martens, B. Baesens, T. Van Gestel, and J. Vanthienen, *Comprehensible credit scoring models using rule extraction from support vector machines*, European Journal of Operational Research (2007), no. 183, 1466–1476.

[16] P. M. Murphy and D. W. Aha, *UCI Repository of machine learning databases irvine, ca: University of California, Department of Information and Computer Science.*, http://www.ics.uci.edu/ mlearn/MLRepository.html, 1994.

[17] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, *An Introduction to Kernel-Based Learning Algorithms*, IEEE Transactions on neural networks **12** (2006), no. 2, 181–201.

[18] J. C. Principe, J. W. Fisher, and D. Xu, *Unsupervised Adaptive Filtering*, vol. 1, S. Haykin(Ed.), ch. Information Theoretic Learning, pp. 265–317, John Wiley & Sons, Inc, New York, 2000.

[19] F. Pérez-Cruz and O. Bousquet, *Kernel Methods and Their Potential Use in Signal processing*, IEEE Signal Processing Magazine (2004), 57–65.

[20] G. Rätsch, *Ensemble Learning Methods for Classification*, (1998), Diploma thesis (in german).

[21] G. Rätsch, T. Onoda, and K.-R. Müller, *Soft Margins for AdaBoost*, Machine Learning **42** (2001), 287–320.

[22] A. Rényi, *On measures of information and entropy*, Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics and Probability 1960, 1960, pp. 548–561.

[23] C. Sanchez-Avila and R. Sanchez-Reillo, *Two different approaches for iris recognition using Gabor filters and multiscale zero-crossing representation*, Pattern Recognition (2005), no. 38, 231 – 240.

[24] B. Schölkopf and A. Smola, *Nonlinear Component Analysis as a Kernel Eigenvalue Problem*, Neural Computation (1998), no. 10, 1299–1319.

[25] C. E. Shannon, *A Mathematical Theory of Communication*, Tech. Report 27, Bell Systems Technical Journal, 1948.

[26] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, ch. 6, pp. 140–194, Cambridge University Press, 2006.

[27] J. Shi and J. Malik, *Normalized Cuts and Image Segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000), no. 8, 888–905.

[28] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*, Chapman & Hall, London, 1986.

[29] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 3 ed., Academic Press, 2006.

[30] M. P. Wand and M. C. Jones, *Kernel Smoothing*, Chapman & Hall/CRC, 1995.

[31] C. K. I. Williams and M. Seeger, *Using the Nyström Method to Speed Up Kernel Machines*, NIPS, 2000, pp. 682–688.