CLASSIFICATION RULES IN STANDARDIZED PARTITION SPACES

Revised Version of the thesis
of the same title that was submitted at the
UNIVERSITY OF DORTMUND
DORTMUND
JULY, 2002
revised
SEPTEMBER, 2002

By
Ursula Maria Garczarek
from Ulm

Acknowledgements

I received a lot of support of many kinds in accomplishing this work. I want to express my gratitude to all the people and organizations that contributed.

- Thanks for financial support:
  to the Deutsche Forschungsgemeinschaft, Sonderforschungsbereich 475.

Some of the results of this thesis have been preprinted as technical reports (Sondhauss and Weihs, 2001b,c; Garczarek and Weihs, 2002) in the Sonderforschungsbereich 475.

- Thanks for support in realizing the simulation study in Chapter 6:
  to Karsten Lübke.

- Thanks for proofreading and comments:
  to Thorsten Bernholt, Anja Busse, Martina Erdbrügge, Uwe Ligges, Stefan Rüping, Christin Schäfer, and Winfried Theis.

- Thanks for maintaining my computer:
  to Uwe Ligges and Matthias Schneider.

- Thanks for help on R and LaTeX:
  to Uwe Ligges, Brian Ripley, Winfried Theis,

- Thanks for discussions on various aspects of statistics and/or machine learning:
  to the members of project A4 of the SFB475: Peter Brockhausen, Thomas Fender, Ursula Gather, Thorsten Joachims, Katharina Morik, Ursula Robers, Stefan Rüping and Claus Weihs,
  and also to Claudia Becker, Thorsten Bernholt, Manfred Jaeger, Joachim Kunert, Tobias Scheffer, Detlef Steuer, and Winfried Theis.

- Thanks for support in managing the administrative aspects of life:
  to Anne Christmann, Peter Garczarek, Myriel Gelhaus, Nicole Radtke, Winfried Theis, Claus Weihs, and Thorsten Ziebach.

- Thanks for support in daily life:
  to Anja Busse, Claudia Becker, Anne Christmann, Martina Erdbrügge, Peter Garczarek, Myriel Gelhaus, Martin Kappler, Rolf Klein, Peter Koschinski, Uwe Ligges, Heinz-Jürgen Metzger, Michael Meyners, Detlef Steuer, Winfried Theis, and Claus Weihs.

Over and above: thanks for love and patience to Peter Garczarek and Myriel Gelhaus, groundwork of everything else.

Dortmund, Germany — Ursula Garczarek
July, 2002

Table of Contents

Abstract

In this work we propose to represent classification rules in a so-called standardized partition space. This offers a unifying framework for the representation of a wide variety of classification rules.

Standardized partition spaces can be utilized for any processing of classification rules that goes beyond their primal goal, the assignment of objects to a fixed number of classes on the basis of observed predictor values.

An important secondary goal in many classification problems is to gain a deeper understanding of the relationship between the predictors and the class membership. A standard approach in this respect is to look at the induced partitions by means of the decision boundaries in the predictor space. The comparison of partitions generated by different classification methods for the same problem is performed to detect procedure-invariantly confirmed relations. In general, such comparisons serve for the representation of the different potential relationships methods can model. In high-dimensional predictor spaces, and with respect to a wide variety of classifiers with largely diverse decision boundaries, such a comparison may be difficult to understand.

In these cases, we propose not to use the predictor space as a constant entity for the comparison but to standardize decision boundaries and regions — and to compare the classification rules with respect to the different patterns that are generated by placing examples from some test set in these standardized regions.

This is possible for any rule that bases its assignment on some transformation of observed predictor values such that these transformations asses the membership of the corresponding object in the classes. The main difficulty is an appropriate scaling of membership values such that membership values of different classification methods are comparable in size. We present a solution to this problem.

Given appropriately scaled membership values we can rank classification rules not only with respect to their ability to assign objects correctly to classes but also with respect to their ability to quantify the membership of objects in classes reliably. We introduce measures for the evaluation of the performance of classification rules in this respect.

In designed simulation study we realize a comparative study of classification rules to demonstrate the possibilities of the suggested method.

Another important secondary goal is the combination of the evidence on the class membership with evidence from other sources. For that purpose, we also propose to scale membership values into the standardized partition space. Only in this case, the goal of the scaling is not the comparability to other classification rules, but the compatibility with the information from other sources. Therefore, we present an additional scaling procedure that weights current membership information of objects in classes with information about past memberships in dynamic domains. In a simulation study, various combination strategies are compared.

The stone mind

Hogen, a Chinese Zen teacher, lived alone in a small temple in the country. One day four traveling monks appeared and asked if they might make a fire in his yard to warm themselves and sleep for the night.

While they were building the fire, Hogen heard them arguing about subjectivity and objectivity. He joined them and said: "There is a big stone. Do you consider it to be inside or outside your mind?"

One of the monks replied "From the Buddhist viewpoint everything is an objectification of the mind, so I would say that the stone is inside my mind."

"Your head must be very heavy," observed Hogen, "if you are carrying around a stone like that in your mind."

(taken from: 101 Zen Stories)

Introduction

In the days of data mining, the number of competing classification techniques is growing steadily. These techniques are developed in different scientific communities and on the basis of diverse theoretical backgrounds.

The aim of the theoretical part of this thesis is to introduce the main ideas of some of these frameworks for classification to show that all of them are well-founded, and nevertheless justifiable and questionable at the same time. As a consequence, general superiority of one approach over the other can only be shown and will only be valid within frameworks that consider the same criteria of performance to be important.

For specific tasks, though, it is important to develop strategies to rank the performance of classification rules to help data analysts in choosing an appropriate classification method. In standard comparative studies, the term 'performance' is largely restricted to the misclassification rate, because this can be most easily formalized and measured. This is unsatisfactory, because 'performance' of a classification rule can stand for much more than only its ability to assign objects correctly to classes. This is, however, the only aspect that is measured by the misclassification rate. Misclassification rates do not cover the variety of demands on classification rules in practice.

In many practical applications of classification techniques it is important to understand the interplay of predictors and class membership. Any classification method finds its own coding for this interplay. Therefore, the comparison and ranking of performance for different methods with respect to their ability to support the analysis of the relationship between class membership and predictor values becomes very difficult. The important feature of the method introduced in this thesis is the possibility of a standardized representation of the classifiers quantitative assessment of membership.

The two main communities involved in classification for data mining applications are statisticians and machine learners. We describe classification as it is approached in these communities. The initial point of this work is determined by the characteristics of data mining applications and the definition of the classification task in decision theoretic terms. These are presented in Chapter 1.

In Chapter 2 the statistical view on decision problems is described. We introduce two statistical frameworks in which decision problems can be solved, namely the Bayesian approach in Section 2.2.1 and the so-called frequentist approach in Section 2.2.2. The strategies to learn 'optimal' classification rules in these frameworks are outlined. To exemplify these approaches, we derive the notation and the implementation of linear and quadratic discriminant classifiers and a continuous naïve Bayes classifier.

In Chapter 3 we describe the general approach to "learning" in machine learning. We embed the learning of classification rules in the corresponding

framework of concept learning. We devise a structure for comparable representations of methods for concept learning and classification rule learning. To link the statistical approach to the machine learning approach, the classifiers introduced in Chapter 2 will be represented that way. We present various aspects of performance that are important for concept learning, and different theoretical frameworks that define good learning in these terms, namely the probably approximately correct learning and the structural risk minimization principle. To exemplify techniques from the machine learning community, the representation and the learning of a decision tree classifier and a support vector machine classifier will be developed.

Chapter 4 summarizes the classification rule learning and the concept learning. The demonstration of the various approaches in chapters 3 and 2 will show that though the underlying principles of appropriate learning in these frameworks are very different, and therefore classification rules gained with methods following these principles may be very different, the basic type of the rules is often similar: rules base their assignment of objects into classes on some quantification of the membership in the classes and assign to the class with highest membership.

This technical similarity yields the basis for standardized partition spaces. The general idea and the implementation will be presented in Chapter 5. We will provide performance measures that quantify the ability of classification rules not only to assign objects correctly to classes but also to provide a more differentiated assessment of higher or lower membership of individual objects in classes. Original membership values are measured on different scales – we

will standardize these so that scaled membership values reflect the classifiers characteristic of classification and give a realistic impression of the classifiers performance.

We demonstrate the comparison of classification rules in standardized partitions spaces in a designed simulation study using statistical experimental design in Chapter 6. Thereby, we introduce another aspect that is of importance for the comparison of classification rules: the use of informative data for the detection of strengths and weaknesses of classification methods. We will introduce a visualization of scaled membership values that can be used to analyze the influence of predictors on the class membership.

In Chapter 7 we will show that an appropriate scaling of membership values into the standardized partition space is not only useful for the comparison of classification rules and the interpretation of the relationship between predictor values and class membership, but also for the combination of evidence from different sources. Current membership information of objects in classes and information about past memberships can be elegantly combined and allow for the dynamization of static classification rules and the incorporation of background knowledge on dynamic structures.

We conclude with a summary and discussion of the thesis in Chapter 8.

Chapter 1

Preliminaries

1.1 Statistics, Machine Learning, and Data Mining

Historically, statistics has its origins as accountancy for the state in the 18th century, while the theory of machine learning has its origins in the beginnings of artificial intelligence and cognitive science in the mid-1950s. Statistics has shifted towards the study of distributions for finding information in data that helps to solve a problem. Research in *artificial intelligence* focussed in the 60ties more on the representation of domain knowledge, and drifted slightly apart from research in *machine learning* where researchers were more interested in general, domain independent methods. Nowadays, machine learning plays again a central role in artificial intelligence. According to the American Association for Artificial Intelligence (AAAI, 2000-2002): *Machine learning refers to a system capable of the autonomous acquisition and integration of knowledge.*

Data mining started as a subtopic of machine learning in the mid-1990s.

The term mainly refers to the secondary analysis of large databases and became a hot topic, because development in database technology led to huge databases. Owners of these databases view these as resources for information, and data mining is concerned with the search for patterns and relationships in the data by building models. Of course, the task to *discover knowledge in data* is much older than the beginning of data mining, in statistics it is well known mainly under the name *explorative data analysis*, other names are *knowledge discovery in databases*, *knowledge extraction*, *data archeology*, *data dredging* , and so on....

New for statisticians is the size of the databases to work on: it goes into the terabytes — millions or billions of records. Interdisciplinary team work thus seems to be a good strategy to tackle data mining tasks, consisting of at least machine learners, statisticians, and database experts. The different technical languages and approaches to data analysis, yet, make team work difficult, and often work is done in parallel and not together.

One aim of this work is to improve the mutual understanding. This thesis focusses on one within the many tasks of data mining: classification. Mainly statisticians and machine learners have contributed solutions to classification problems — in parallel. We will present different approaches to the task. Similarities and differences are worked out.

The many methods to solve classification problems raise another problem: to decide for a given application, which method to use. Not only differ machine learning and statistics in their ways to approach the classification task, but also in their ways to evaluate methods, and to define, what a good method is. In

classification the compromise is to reduce good to mean having a high prediction accuracy — but even prediction accuracy can be defined and evaluated in many different ways. The second aim of this work is to provide a tool that can be used to evaluate classification methods from statistics and machine learning with respect to more goodness aspects, the *standardized partition space*.

One major endeavor in the presentation of classification approaches and in the comparison of classification rules is to be precise about assumptions. This has two reasons:

1. Being precise about assumptions is a basic necessity of interdisciplinary work.

   Within communities some assumptions are so common that they are deemed common sense such that their justification is not questioned and one feels no need to trace their effects. Or — knowledge about justification and effect is presumed from former discussions within the community. These non-stated a-priori assumptions are one cause of major misunderstandings when crossing the borders of communities.

2. In data mining applications many common assumptions are violated and one has to be aware of the consequences thereof.

   Data mining is concerned mainly with secondary data analysis, where data has been collected according to some strategy serving some other purpose, but not with regard to the data analysis. In consequence, common assumptions about the "data generating process" are violated: In

particular the assumption that data items have been sampled independently and from the same distribution does not hold. Data may be collected with "selection bias", such that they are not representative for an intended population for future events, and so on. Hand (1998) gives an excellent overview of these problems and their consequences.

A first a-priori of this work stated clearly: we can not escape assumptions when learning. It is the general problem of inductive inference that any learning has to assume some relation between the past and the future. The way we set up this relation will always influence the outcome of our learning process. One fights windmills when fighting assumptions – one better is aware of them to be able to change them.

1.2 Basic Notation

The following notation for sets and their elements is used throughout this work:

Abbr.	**Description**
$\mathbb{N}$	Set of natural numbers. If possible, we use letters from the middle of the Latin alphabet i, j, k, n, m to denote natural numbers, small letters for counting indices capital letter for the end-points, e.g. $n = 1, ..., N$.
$\mathbb{R}$	Set of real numbers. If possible, we use letters from the end of the Latin alphabet u, v, w, x, y, z to denote real numbers.
$\mathbb{R}^+$	Set of positive real numbers.
$\mathbb{R}_0^+$	Set of non-negative real numbers.

Abbr.	Description
$(a, b), [a, b] \subseteq \mathbb{R}$	open and closed interval on $\mathbb{R}$.
$(a, b], [a, b), \subset \mathbb{R}$	left open and right open interval on $\mathbb{R}$
$\mathbb{R}^K, K \in \mathbb{N}$	Set of K-dimensional vectors of real numbers. We mark vectors with arrows, e.g. $\vec{x} \in \mathbb{R}^K$. In general, $\vec{x}$ can be a row vector or a column vector. If it is necessary to distinguish row vectors from column vectors, row vectors will be marked with a single quotation mark as in $\vec{x}'$.
$\mathbb{R}^{K \times N}, K, N \in \mathbb{N}$	Set of $K \times N$-dimensional matrices of real numbers. We use bold-face capital letters for matrices, e.g. $\mathbf{M} \in \mathbb{R}^{K \times N}$. We note transposed matrices with a single quotation mark, as in $\mathbf{M}' \in \mathbb{R}^{N \times K}$.
$\mathbb{Q}^K \subset \mathbb{R}^{K \times K}$	Set of quadratic matrices, $\mathbf{Q} \in \mathbb{Q}^K$.
$\mathbb{Q}^K_{[pd]} \subset \mathbb{Q}^K$	Set of positive definite matrices.
$\mathbb{Q}^K_{[D]} \subset \mathbb{Q}^K$	Set of diagonal matrices, where only diagonal elements are non-zero.
e.g. $\mathbf{U} \subset \mathbb{R}$	Subspaces of the real numbers. These sets will always be denoted by bold-face letters, preferably capital Latin letters.
e.g. $\mathbf{\Omega}$, $\mathbf{S}$	Set of arbitrary elements. These sets will always be denoted by bold-face letters, preferably capital and from the Greek alphabet, or as special case, $\mathbf{S}$. If possible, elements are noted with the corresponding small letters.
e.g. $\lvert\mathbf{\Omega}\rvert \in \mathbb{N} \cup \infty$	(possibly infinite) potency of set $\mathbf{\Omega}$.

1.3 Decision Problems

Decision theory is concerned with the problem of decision making. Decision theory is part of many different scientific disciplines. Typically, it is seen to be a research area of mathematics, as a branch of game theory. It plays an important role in Artificial Intelligence, e.g. in planning (Blythe, 1999) and expert systems (Horvitz et al., 1988). In statistical decision theory one is concerned with decisions under uncertainty, when there exists statistical knowledge that can be used to quantify these uncertainties.

Standard formulations of decision problems use the following three basic elements:

Definition 1.3.1 (Basic elements of Decision Problems).

Decision problems are formulated in terms of

state	some true state from a set of states of the world, $s \in \mathbf{S}$,
action	some chosen action from a set of possible actions $a \in \mathbf{A}$,
loss	a loss function $L(\circ, \circ) : \mathbf{S} \times \mathbf{A} \to \mathbb{R}$.

The loss $L(s, a)$ quantifies the loss one experiences if one chooses action a while the "true state of the world" is s.

The decision maker would like to choose the action with minimum loss, with the difficulty that she is uncertain about the true state of the world.

In this work, one special instantiation of a decision problem plays the central role: the classification problem.

Definition 1.3.2 (Classification Problem).

In a classification problem we want to assign some object $\omega \in \mathbf{\Omega}$ into one out of G classes $\mathbf{C} := \{1, ..., G\}$. And we assume, any $\omega \in \mathbf{\Omega}$ belongs to one and only one of these classes $c(\omega) \in \mathbf{C}$. We define the basic elements of classification problems to be:

state	the true class $c(\omega) \in \mathbf{C}$ for the given $\omega \in \mathbf{\Omega}$,
action	the assigned class $\hat{c}(\omega) \in \mathbf{C}$,
loss	some loss function $L(\circ, \circ) : \mathbf{C} \times \mathbf{C} \to \mathbb{R}$.

The most common loss function for classification problems is the so-called *zero-one loss* $L^{[01]}(\circ, \circ) : \mathbf{C} \times \mathbf{C} \to \{0, 1\}$ where any wrong assignment is penalized with one:

$$L^{[01]}(c, \hat{c}) := \begin{cases} 0 & \text{if } c = \hat{c}, \\ 1 & \text{otherwise.} \end{cases} \tag{1.3.1}$$

In statistics, the point estimation problem is another important decision problem. Some solutions of the classification problem involve solutions of point estimation problems.

Definition 1.3.3 (Point estimation).

In general point estimation the goal is to decide about the value $\hat{g}$ of some function $g(\circ) : \Theta \to \Gamma \subseteq \mathbb{R}^M$, $M \in \mathbb{N}$, of the unknown parameter $\theta \in \Theta$ of some distribution $P_{V|\theta}$. We call $g(\theta)$ the *estimand*, and the action $\hat{g}$ the "estimate". For convenience of the presentation we will assume $\Theta \subseteq \mathbb{R}^M$, and $g : \Theta \to \Theta$

to be the identity mapping $g(\theta) \equiv \theta$. With this, we define a point estimation problem to consist of

state	the parameter $\theta \in \Theta \subseteq \mathbb{R}^M$
action	the estimate $\hat{\theta} \in \Theta$
loss	some loss function $L(\circ, \circ) : \Theta \times \Theta \rightarrow \mathbb{R}$,

The most common loss function in point estimation is the *quadratic loss function* $L^{[q]}$:

$$L^{[q]}(\theta, \hat{\theta}) \quad := \quad \left(\theta - \hat{\theta}\right) \mathbf{Q} \left(\theta - \hat{\theta}\right)', \tag{1.3.2}$$

with $\mathbf{Q} \in \mathbb{Q}^K_{[pd]}$. That means $\mathbf{Q}$ is some $K \times K$ positive definite matrix. Another loss function for point estimation problems is the *ϵ-zero-one loss function* $L^{[\epsilon 01]}(\circ, \circ) : \mathbf{C} \times \mathbf{C} \rightarrow \{0, 1\}$ for some (and may be arbitrarily small) $\epsilon \in \mathbb{R}^+$:

$$L^{[\epsilon 01]}(\theta, \hat{\theta}) \quad := \quad \begin{cases} 0, & \text{if } \theta \in B_\epsilon(\hat{\theta}), \\ 1 & \text{otherwise,} \end{cases} \tag{1.3.3}$$

where $B_\epsilon(\hat{\theta})$ is a ball of radius ϵ in Θ centered at $\hat{\theta}$.

These definitions of the classification problem and the point estimation problem form the decision theoretic framework for the statistical approach to classification rule learning.

Chapter 2

Statistics

We will describe statistical solutions to decision theoretic problems, based on the presentations in Berger (1995) and Bernardo and Smith (1994), but also Mood et al. (1974), Gelman et al. (1995), and Lehmann (1983). We will describe two main decision theoretic schools, namely the Bayesian and the frequentist or (classical) school. For a current overview and references to the many more statistical frameworks, we refer to Barnett (1999).

2.1 Basic Definitions

In Definition 1.3.1, we follow Berger (1995) in assuming a certain "true state of the world", as this is common and convenient. We want to focus on similarities and differences of certain perspectives on decision problems, such that a common definition of the basic elements seems inevitable. From a puristic Bayesian point of view, though, we would rather prefer to define potentially observable, yet currently unknown events upon some assumed external, objective reality. Therefore, Bernardo and Smith (1994) define slightly different, but related basic elements of a decision problem. This is just one exemplary

case of the basic difficulty that the clarity of a certain perspective on a problem may suffer from an "embracing" presentation of the problem for substantially different perspectives. We try to keep this shortcoming as small as possible.

According to the Bayesian theory of probability, one can quantify the uncertainty about the true state of the world. This results in the specification of a probability space $(\mathbf{S}, \mathcal{A}, P_S)$ such that the distribution P_S quantifies the subjective probabilities of the events that the world is in a specific state $s \in A$ for all members A of a algebra $\mathcal{A}$ of subsets of $\mathbf{S}$.

Definition 2.1.1 (Prior distribution).

In Bayesian decision theory the uncertainty about the potential state of the world $s \in \mathbf{S}$ prior to a statistical investigation is quantified in a probability space $(\mathbf{S}, \mathcal{A}, P_S)$ with a spefified *prior distribution* P_S of the uncertain quantity S.

[**Notation**] Throughout this work, we have two types of random quantities: those that are formally defined as random variables in both schools, frequentist or Bayesian, and the "uncertain quantities" that can be represented by random variables only in the Bayesian school. In texts, we will avoid to represent these quantities by capital letters. In Bayesian formulae, though, where they are processed as random variables, they are naturally notated with capital letters.

With the three elements in Definition 1.3.1, and optionally the specification of a prior distribution as in Definition 2.1.1 a so-called *no-data decision problem* is formulated.

To obtain information about the true state of the world a statistical investigation is performed. This investigation is deemed a conceptual random experiment with elementary events $\omega \in \mathbf{\Omega}$ and corresponding probability space $(\mathbf{\Omega}, \mathcal{A}, P)$. The events of this experiment are described by some random variable $X(\circ) : \mathbf{\Omega} \to \mathbf{X} \subseteq \mathbb{R}$, or by some random K-dimensional row vector $\vec{X}(\circ) : \mathbf{\Omega} \to \mathbf{X} \subseteq \mathbb{R}^K$, $K \in \mathbb{N}$, and $\mathbf{X}$ in this context is called *sample space*.

[**Notation**] We will denote real valued random quantities always with capital letters, preferably Latin letters from near the end of the alphabet. As an exception, we will use V to denote general random quantities, real-valued variables or vectors. Also, we use the corresponding small letter to denote a value of the random quantity $V = v, v \in \mathbf{X}$. We only consider discrete and continuous random variables and possibly mixed vectors of discrete and continuous random variables. Whenever defining a function of a random variable, we assume it to be measurable. The distribution of a random quantity $V \in \mathbf{X}$ is denoted by P_V, and the corresponding density with $p(v), v \in \mathbf{X}$. We write $V \sim P_V$ to say, V is a random quantity with distribution P_V. Ignoring subtleties with respect to sets of measure zero, we use the terms "distribution" and "density" interchangeable. In case of discrete random variables, we define $\mathbf{X}$ to be minimal in the sense, that only values with positive probability are included: $\mathbf{X} = \left\{ v \in \mathbb{R}^K : p(v) > 0 \right\}$. We note corresponding integrals for integrable functions $f : \mathbf{X} \to \mathbb{R}$ with respect to the counting measure and the Lebesgue measure both with

$$\int_{\mathbf{X}} f(v)p(v)d\mu(v) \quad = \quad \begin{cases} \sum_{v \in \mathbf{X}} f(v)p(v) & \text{discrete case,} \\ \int_{\mathbf{X}} f(v)p(v)dv & \text{continuous case.} \end{cases} \tag{2.1.1}$$

If the distribution of V is dependent on some other quantity q, we note this by $P_{V|q}$ and $p(v|q), v \in \mathbf{X}$.

Of course, the outcome of the statistical investigation has to be informative about the true state of the world s, or, in other terms, the probability distribution $P_{\vec{X}}$ of $\vec{X}$ has to depend upon s, denoted by $P_{\vec{X}|s}$, $s \in \mathbf{S}$. This defines

"being informative" mainly from the perspective of the frequentist school in statistics (see Section 2.2.2). From the Bayesian perspective being informative is translated the other way round: we know how our uncertainty prior to the statistical investigation — coded in P_S — changes after observing $\vec{x}$. The changed uncertainty is quantified in the so-called *a-posteriori distribution* $P_{S|\vec{x}}$, and $P_{S|\vec{x}}$ and P_S differ at least for some $\vec{x}\in\mathbf{X}$. The update mechanism for determining $P_{S|\vec{x}}$ for all $\vec{x}\in\mathbf{X}$ in the Bayesian context is derived from the specification of the joint distribution $P_{\vec{X},S}$ of S and $\vec{X}$ that can be decomposed in the a-priori distribution P_S and the *sampling distribution* which we also call the *informative distribution* $P_{\vec{X}|s}$ for all $s\in\mathbf{S}$ (see Section 2.2.1).

Therefore, it is no offence to the Bayesian formalism to describe the model assumptions common to the Bayesian and the frequentist approaches on the basis of $P_{\vec{X}|s}$. These model assumptions are comprised in a statistical model:

Definition 2.1.2 (Statistical Model).

A statistical model $\mathbf{\Lambda}$ codes the distributional assumptions about the distribution of a random vector $\vec{X}$ with counterdomain $\mathbf{X}$ in a triple

$$\mathbf{\Lambda} := \left(\mathbf{X}, \mathcal{A}, \mathcal{P}_{\vec{X}|\mathbf{S}}\right),$$

where $(\mathbf{X}, \mathcal{A})$ is some measurable space, and $\mathcal{P}_{\vec{X}|\mathbf{S}} := \left\{P_{\vec{X}|s}, s\in\mathbf{S}\right\}$ is a corresponding class of distributions.

Definition 2.1.3 (Statistical Investigation).

A statistical investigation consists of the data $\vec{x}$ and a representation of the information content in $\vec{x}$ on $s\in\mathbf{S}$. This is coded in a statistical model $\mathbf{\Lambda}$ that

specifies the distributional assumptions about the data generating process of $\vec{x}$ in dependence on $s \in \mathbf{S}$. And – optionally – the uncertainty about s is quantified in a prior distribution P_S. A statistical investigation thus consists of

information	some observed data $\vec{x}$,
representation	$\mathbf{\Lambda} : \left(\mathbf{X}, \mathcal{A}, \mathcal{P}_{\vec{X}\mid\mathbf{S}}\right)$ and optionally some prior distribution P_S.

We will indicate the origin of the density of a random variable $\vec{X}$ within a certain statistical model by writing $p(\vec{x}|s, \mathbf{\Lambda})$.

The density of the sampling distribution $p(\vec{x}|s, \mathbf{\Lambda})$ is sometimes perceived as a function $l_{\vec{x}}(\circ|\mathbf{\Lambda}) : \mathbf{S} \to \mathbb{R}_0^+$ in s for fixed $\vec{x}$:

$$l_{\vec{x}}(s|\mathbf{\Lambda}) \quad := \quad p(\vec{x}|s, \mathbf{\Lambda}), s \in \mathbf{S}. \tag{2.1.2}$$

It is then called *likelihood function*.

On the basis of the formalization of decision problem in Chapter 1 and the definition of a statistical investigation, we can now define a scheme for a comprised representation of various statistical approaches to classification that we will use to clarify their similarities and distinctions:

Definition 2.1.4 (Scheme for Statistical Decision Problems). In this work, *statistical decision problem* are instantiated by specifying the basic elements of a decision problem 1.3.1 and modelling a statistical investigation 2.1.3.

state	some true state from a set of states of the world, $s \in \mathbf{S}$,
action	some chosen action from a set of possible actions $a \in \mathbf{A}$,
loss	a loss function $L(\circ, \circ) : \mathbf{S} \times \mathbf{A} \rightarrow \mathbb{R}$,
information	some observed data $\vec{x}$,
representation	some statistical $\mathbf{\Lambda} : \left(\mathbf{X}, \mathcal{A}, \mathcal{P}_{\vec{X}\mid\mathbf{S}}\right)$ and optionally some prior distribution P_S.

We do so to define the statistical classification problem and the statistical point estimation problem.

Definition 2.1.5 (Statistical Classification Problem).

In a statistical classification problem, we want to assign some object $\omega \in \mathbf{\Omega}$ into one and only one class $\hat{c} \in \mathbf{C}$. And we assume, any $\omega \in \mathbf{\Omega}$ belongs to one and only one of these classes $c(\omega) \in \mathbf{C}$. We take measurements $\vec{x}(\omega)$ on the object. The measured entities are often called *predictors*, the corresponding space $\mathbf{X}$ the *predictor space*. The information content of $\vec{x}(\omega)$ on $c(\omega)$ is coded in a statistical model $\mathbf{\Lambda}$ that specifies distributional assumptions about the data generating processes for the measurements on objects belonging to each of the classes $c \in \mathbf{C}$, the *class of informative distributions*. In addition, the uncertainty about the true class $c(\omega)$ of the object ω prior to the measurement may be coded in the prior distribution P_C. In its comprised form, the statistical classification problem reads as follows:

state	the true class $c(\omega) \in \mathbf{C}$ for the given $\omega \in \mathbf{\Omega}$,
action	the assigned class $\hat{c}(\omega) \in \mathbf{C}$,
loss	the zero-one loss function $L = L^{[01]}$,
information	some measurements $\vec{x}(\omega)$,
representation	$\mathbf{\Lambda} : \left(\mathbf{X}, \mathcal{A}, \mathcal{P}_{\vec{X}\mid\mathbf{C}}\right)$ and optionally some prior distribution P_C.

Definition 2.1.6 (Statistical Point Estimation Problem).

We set up the statistical point estimation problem by assuming $\vec{x}_n \in \mathbf{X} \subseteq \mathbb{R}^K$, $n = 1, ..., N$, to be the realization of a random sample of size N from a population with distribution $P_{\vec{X}|\theta} \in \mathcal{P}_{\vec{X}|\Theta}$ for some $\theta \in \Theta \subseteq \mathbb{R}^M$. That is our sample space is given by $\mathbf{X}^N := \times_{n=1}^N \mathbf{X}$, and corresponding product measurable space $\left(\mathbf{X}^N, \mathcal{A}^N\right)$. The joint density $p(\vec{x}_1, ..., \vec{x}_N|\theta)$ of a random sample factors as follows:

$$p(\vec{x}_1, ..., \vec{x}_N|\theta) = \prod_{n=1}^{N} p(\vec{x}_n|\theta).$$

The corresponding product distribution $\bigotimes_{n=1}^N P_{\vec{X}_n} = \bigotimes_{n=1}^N P_{\vec{X}}$ is denoted by $P_{\vec{X}}^N$. In other words, the random vectors $\vec{X}_1, \vec{X}_2, ..., \vec{X}_N$ are identically and independently distributed (**i.i.d.**), according to $P_{\vec{X}|\theta}$ for some $\theta \in \Theta$. The scheme of the statistical point estimation problem reads as follows:

state	the parameter $\theta \in \Theta \subseteq \mathbb{R}^M$,
action	the estimate $\hat{\theta} \in \Theta$,
loss	some loss function $L(\circ, \circ) : \Theta \times \Theta \to \mathbb{R}$,
information	$\vec{x}_n \in \mathbb{R}^K$, $n = 1, ..., N$,
representation	$\mathbf{\Lambda}$: $\left(\mathbf{X}, \mathcal{A}, \mathcal{P}_{\vec{X}|\Theta}\right)^N := \left(\mathbf{X}^N, \mathcal{A}^N, \left\{P_{\vec{X}|\theta}^N, \theta \in \Theta\right\}\right)$ and optionally some prior distribution P_θ.

A reasonable method to decide on an action in a statistical decision problem is to look at the "expected" loss of making a decision, dependent on the statistical investigation, and to minimize it.

2.2 Derivations of Expected Loss

We will present two standard types of "expected" loss, defined according to the Bayesian and to the frequentist approach to statistical inference.

[**Notation**] Here and in the following, we will denote with $\mathbf{E}(g(V))$ the expected value of some integrable function $g(V)$ with respect to the distribution P_V of some random variable $V \in \mathbf{X}$:

$$\mathbf{E}(g(V)) \quad := \quad \int_{\mathbf{X}} g(v)p(v)d\nu(v).$$

If the distribution of V depends on some other quantity q this will be denoted either with $\mathbf{E}(g(V)|q)$ or with $\mathbf{E}_{V|q}(g(V))$, whichever seems to be more useful in a given situation.

2.2.1 Bayesian Approach

In the Bayesian context, P_S quantifies the uncertainty we have about the state of the world. This enables Bayesians to quantify the expected loss of some action $a \in \mathbf{A}$ in a decision problem by the weighted average loss with respect to the prior distribution P_S:

$$\rho\left(a, P_S\right) := \mathbf{E}\left(L(S,a)\right) = \int_{\mathbf{S}} L(s,a)p(s)d\mu(s). \tag{2.2.1}$$

After the investigation, given the observed data $\vec{x}$, the uncertainty about the state of the world is changed according to simple probability calculus applied on the distributional assumptions in $\mathbf{\Lambda}$. In $\mathbf{\Lambda}$ the full Bayesian probability model is defined, that is the joint probability distribution of the unknown state

of the world and the yet unobserved data of the statistical investigation:

$$\begin{aligned} p(s, \vec{x}|\mathbf{\Lambda}) &= p(s|\vec{x}, \mathbf{\Lambda})p(\vec{x}|\mathbf{\Lambda}) \\ &= p(\vec{x}|s, \mathbf{\Lambda})p(s|\mathbf{\Lambda}). \end{aligned}$$

Typically this is done by specifying the prior distribution P_S and the sampling distribution $P_{\vec{X}|s}$. We denote the joint probability and its decomposition by $P_{S,\vec{X}} = P_S \otimes P_{\vec{X}|S}$.

The update-mechanism on the basis of the full Bayesian model results in:

$$p(s|\vec{x}, \mathbf{\Lambda}) = \frac{p(\vec{x}|s, \mathbf{\Lambda})p(s|\mathbf{\Lambda})}{p(\vec{x}|\mathbf{\Lambda})} \tag{2.2.2}$$

$$= \frac{p(\vec{x}|s, \mathbf{\Lambda})p(s|\mathbf{\Lambda})}{\int_{\mathbf{S}} p(\vec{x}|\tilde{s}, \mathbf{\Lambda})p(\tilde{s}|\mathbf{\Lambda})d\mu(\tilde{s})}. \tag{2.2.3}$$

The updated uncertainty is thus quantified in $P_{S|\vec{x},\mathbf{\Lambda}}$, the so-called *posterior* distribution.

When deciding for $a \in \mathbf{A}$, we now expect a weighted average loss with respect to the posterior distribution $P_{S|\vec{x},\mathbf{\Lambda}}$:

$$\rho\left(a, P_S|\vec{x}, \mathbf{\Lambda}\right) = \mathbf{E}\left(L(S, a)|\vec{x}, \mathbf{\Lambda}\right) = \int_{\mathbf{S}} L(s, a)p(s|\vec{x}, \mathbf{\Lambda})d\mu(s).$$

Appropriate Bayesian modelling is an art of its own, elaborated extensively in Bernardo and Smith (1994). The Bayesian school is often criticized for the use of prior distributions, when there is not some *knowledge* (considered objective) that justifies to possibly favor one state over the other. The influence of a specific subjective prior on the final decision is from the perspective of many scientists objectionable. For large data sets, though, there is not really a problem: under appropriate (and typically fulfilled) conditions, when using the Bayesian update mechanism for more and more data, the influence of the prior

distribution vanishes (Bernardo and Smith, 1994, Section 5.3). For moderate sample sizes there exists a vast number of literature on how to define so-called *non-informative* priors. Bernardo and Smith (1994, p. 357) call the search for non-informative priors that "let the data speak for themselves" or represent "ignorance" the search for a Bayesian "Holy Grail" . We share their view that the search for a kind of "objective" prior is misguided, as *Put bluntly: data cannot ever speak for themselves;* (Bernardo and Smith, 1994, p. 298). With careful consideration, what the unknown quantity of interest in an investigation really is, what can be achieved are "minimally informative" priors, chosen relative to the information which can be supplied by a particular experiment. The corresponding theory of *reference priors* is presented in Bernardo and Smith (1994, Section 5.4).

In case of the statistical classification problem the reference prior and the non-informative priors from any principle to derive non-informative priors coincide:

$$p(c) \;:=\; 1/G, \text{ for all } c \in \mathbf{C} = \{1, ..., G\}\,. \tag{2.2.4}$$

In return to the effort of careful modelling, the basic Bayesian principle for making a decision is very simple and straightforward:

Principle 2.2.1 (Conditional Bayesian Principle).

In any statistical decision problem, always choose an action $a \in \mathbf{A}$ *that minimizes the Bayesian expected loss given the current knowledge:*

$$a^{[B]}(L) \;:=\; \arg\min_{a \in \mathbf{A}} \left\{ \rho(a, P_{S|\vec{x},\mathbf{\Lambda}}) \right\} \tag{2.2.5}$$

The word "conditional" emphasizes the fact that with the conditional Bayesian principle, we are looking for the best decision conditional on a given body of acquired knowledge, subject to the specific, observed data $\vec{x}$. This is one of the major differences to the frequentist approach, as you will see in Section 2.2.2.

Applying the conditional Bayesian principle to the statistical classification problem set up in Definition 2.1.5 results in assigning the object ω with measurement $\vec{x}(\omega)$ to the class with lowest error probability (cp. (2.2.7)) and the highest posterior probability (cp. (2.2.8)):

$$\begin{aligned}
c^{[B]} &= \arg\min_{c\in\mathbf{C}} \left\{\rho\left(c, P_{C|\vec{x},\mathbf{\Lambda}}\right)\right\} \\
&= \arg\min_{c\in\mathbf{C}} \left\{\mathbf{E}\left(L^{[01]}(C,c)|\vec{x},\mathbf{\Lambda}\right)\right\} \qquad (2.2.6)\\
&= \arg\min_{c\in\mathbf{C}} \left\{\int_{\mathbf{C}} L^{[01]}(\tilde{c},c)p(\tilde{c}|\vec{x},\mathbf{\Lambda})d\mu(\tilde{c})\right\} \\
&= \arg\min_{c\in\mathbf{C}} \left\{\sum_{\substack{\tilde{c}\in\mathbf{C}\\ \tilde{c}\neq c}} p(\tilde{c}|\vec{x},\mathbf{\Lambda})\right\} \\
&= \arg\min_{c\in\mathbf{C}} \left\{1 - p(c|\vec{x},\mathbf{\Lambda})\right\} \qquad (2.2.7)\\
&= \arg\max_{c\in\mathbf{C}} \left\{p(c|\vec{x},\mathbf{\Lambda})\right\}. \qquad (2.2.8)
\end{aligned}$$

With the factorization

$$p(c|\vec{x},\mathbf{\Lambda}) = \frac{p(c,\vec{x}|\mathbf{\Lambda})}{\sum_{g\in\mathbf{C}} p(g,\vec{x}|\mathbf{\Lambda})} = \frac{p(c|\mathbf{\Lambda})p(\vec{x}|c,\mathbf{\Lambda})}{\sum_{g\in\mathbf{C}} p(g,\vec{x}|\mathbf{\Lambda})},$$

and because the denominator does not depend on c, we present the final assignment as it is most often cited:

$$\begin{aligned}
c^{[B]} &= \arg\max_{c\in\mathbf{C}} \left\{p(c|\vec{x},\mathbf{\Lambda})\right\} \qquad (2.2.9)\\
&= \arg\max_{c\in\mathbf{C}} \left\{p(c|\mathbf{\Lambda})p(\vec{x}|c,\mathbf{\Lambda})\right\}. \qquad (2.2.10)
\end{aligned}$$

Concerning the statistical point estimation problem, the posterior uncertainty about the true parameter $\theta \in \Theta$ is coded in $P_{\theta|\vec{x},\mathbf{\Lambda}}$. If one wants to decide for a single best estimate $\hat{\theta}$, this depends on the specified loss.

For the quadratic loss 1.3.2 the best estimate is the mean of the posterior distribution (see Bernardo and Smith, 1994):

$$\begin{aligned} \hat{\theta}^{[B]}(L^{[q]}) &= \arg\min_{\hat{\theta}\in\Theta} \mathbf{E}\left(L^{[q]}(\theta,\hat{\theta})|\vec{x},\mathbf{\Lambda}\right) && (2.2.11)\\ &= \mathbf{E}\left(\theta|\vec{x},\mathbf{\Lambda}\right). && (2.2.12) \end{aligned}$$

The Bayes estimate with respect to the ϵ-zero-one loss given in (1.3.3) for $\epsilon \to 0$, $\epsilon > 0$, is the mode of the posterior distribution (see Bernardo and Smith, 1994, and compare with (2.2.6) and (2.2.8)):

$$\begin{aligned} \hat{\theta}^{[B]}(L^{[\epsilon 01]}) &= \lim_{\epsilon\to 0^+} \arg\min_{\hat{\theta}\in\Theta} \mathbf{E}\left(L^{[\epsilon 01]}(\theta,\hat{\theta})|\vec{x},\mathbf{\Lambda}\right) && (2.2.13)\\ &= \lim_{\epsilon\to 0^+} \arg\max_{\hat{\theta}\in\Theta} \int_{B_{\epsilon(\hat{\theta})}} p(\theta|\vec{x},\mathbf{\Lambda})d\mu(\theta). && (2.2.14) \end{aligned}$$

The mode of the posterior distribution is also called the *maximum a-posteriori estimate (**MAP**)* of θ.

The conditional Bayesian principle is an implementation of another basic principle of statistical inference, the so-called *likelihood principle*:

Principle 2.2.2 (Likelihood Principle).

Given a decision problem 1.3.1, all the information about the state of the world that can be obtained from a statistical investigation is contained in the likelihood function $l_{\vec{x}}(\circ|\mathbf{\Lambda})$ for the actual observation $\vec{x}$. Two likelihood functions (from the same or different investigations) in statistical models $\mathbf{\Lambda}_1$ and

$\mathbf{\Lambda}_2$ *contain the same information about the state of the world, if they are proportional to each other:*

$$\frac{l_{\vec{x}_1}(s|\mathbf{\Lambda}_1)}{l_{\vec{x}_2}(s|\mathbf{\Lambda}_2)} \equiv r, r{\in}\mathbb{R}, \textit{ for all } s{\in}\mathbf{S}.$$

The maximum-likelihood point estimation follows another implementation of the likelihood principle:

Definition 2.2.1 (Maximum Likelihood Estimation).

Let $\hat{\theta}^{[ML]}{\in}\Theta$ be the parameter value under which the data $\vec{x}$ would have had the highest probability of arising:

$$\hat{\theta}^{[ML]} = \arg\max_{\theta\in\Theta} l_{\vec{x}}(\theta|\mathbf{\Lambda}),$$

If this parameter value exists, then according to the maximum likelihood principle, this is the best estimate of θ.

Note that this principle makes no use of any definition of loss, and has not been derived as an optimal solution to a decision problem. Yet, maximum-likelihood estimation is widely used, because it has high intuitive appeal and often the resulting estimators have desirable properties also from the perspective of other approaches. In particular asymptotically, that is for $N \to \infty$ in Definition 2.1.6, the maximum-likelihood estimation can be justified from the Bayesian perspective (see Gelman et al., 1995), as well as from the frequentist perspective (see Lehmann, 1983).

2.2.2 Frequentist approach

In the frequentist school, we think of the state of the world as being unknown but fixed. The information in the data $\vec{x}$ of the statistical investigation is modelled by assuming $\vec{x}$ to be the realization of some random vector $\vec{X}$, distributed according to $P_{\vec{X}|s}$ depending on s. According to the frequentist school probabilities are *the limiting values of relative frequencies in indefinitely long sequences of repeatable (and identical) situations* (Barnett, 1999, p.76). Despite the difficulties when formalizing this "notion" of probability to some "definition" (see also Barnett, 1999, Section 3.3) it is a notion that motivates many statistical procedures. This is also true in statistical decision theory where from a frequentist point of view the goal is no longer only to decide for the best action given the one observed $\vec{x}$ at hand, which is targeted when applying the conditional Bayesian principle. A frequentist wants to decide for some action following some *decision rule* that minimizes what she expects to lose if this rule was repeatedly used with varying random $\vec{X}\in\mathbf{X}$.

Definition 2.2.2 (Decision Rule).

Let $\mathbf{A}$ be the action space of some decision problem and $\mathbf{X}$ be the sample space of some related statistical investigation.

A non-randomized decision rule is a function

$$\delta(\circ) : \mathbf{X} \rightarrow \mathbf{A}.$$

If $\vec{x}\in\mathbf{X}$ is the observed value of the sample information, then $\delta(\vec{x})\in\mathbf{A}$ is the action that will be taken. The space of all non-randomized decision rules will be denoted by $\mathbf{\Delta}$.

A randomized decision rule

$$\delta^*(\circ,\circ) : \mathbf{X} \times \mathcal{A}_{\mathbf{A}} \rightarrow [0,1]$$

is — for given $\vec{x}\in\mathbf{X}$ — a probability distribution $P_{A|\vec{x}}$ with corresponding measurable space $(\mathbf{A}, \mathcal{A}_{\mathbf{A}})$. If $\vec{x}\in\mathbf{X}$ is the observed value of the sample information, then $\delta^*(\vec{x}, \mathbf{B})$ is the probability that some action $a\in\mathbf{B} \subseteq \mathcal{A}_{\mathbf{A}}$ will be taken. The space of all non-randomized decision rules will be denoted by $\mathbf{\Delta}^*$.

Decision rules for estimation problems are called *estimators* and decision rules for classification *classification rules*.

The expected loss in terms of the frequentist approach is the hypothetical average loss, if we used the decision rule for the outcomes of the indefinitely number of repetitions of the statistical investigation given constant true state s:

Definition 2.2.3 (Frequentist Expected Loss, Risk Function).

The risk function of a randomized decision rule $\delta\in\mathbf{\Delta}^*$ is defined by

$$R(s,\delta) \;=\; \mathbf{E}_{\vec{X}|s}\left(\mathbf{E}_{\vec{A}}\left(L(s,\delta(\vec{X},A))|\vec{X}\right)\right), \tag{2.2.15}$$

which results for a non-randomized decision rule $\delta\in\mathbf{\Delta}$ in

$$R(s,\delta) \;=\; \mathbf{E}_{\vec{X}|s}\left(L(s,\delta(\vec{X}))\right). \tag{2.2.16}$$

Now the difficulty is that we still do not really know, what is the best decision rule and resulting action is, because the expected loss $R(s, \delta)$ – analogously to the individual loss $L(s, a)$ – is dependent on the unknown true state of the world $s \in \mathbf{S}$.

Actually, we can set up a no-data decision problem, where decision rules are actions, and the risk function is the loss function. And the frequentist wants to decide upon the best decision rule.

Of course, the best rule would minimize $R(s, \delta)$ for all values of $s \in \mathbf{S}$ among all $\delta \in \mathbf{\Delta}^*$. Such a rule is called the *uniformly best rule* on $\mathbf{\Delta}^*$. Yet, for most problems it is not possible to find such a rule. Sometimes on a restricted set of rules $\delta \in \mathbf{\Delta}^{[r]} \subset \mathbf{\Delta}^*$, though, a uniformly best rule exists. Of course, the restrictions should be such that we do not feel that important rules are cancelled out, rather that we feel that any proper rule should fulfill this requirement anyway. A very general restriction follows the so-called *invariance principle*. Informally, the invariance principle is based on the following argument (cp. Berger, 1995):

Principle 2.2.3 (Principle of Rational Invariance).

The action taken in a statistical decision problem should not depend on the unit of measurement used, or other such arbitrarily chosen incidentals.

To present the formal invariance principle, we need to define "arbitrarily chosen incidentals". Let two statistical decision problems be defined with corresponding formal structures, $I = 1, 2$, by:

states	$s_I \in \mathbf{S}_I$,
actions	$a_I \in \mathbf{A}_I$,
losses	$L_I : \mathbf{S}_I \times \mathbf{A}_I \to \mathbb{R}$,
Informations	$\vec{x}_I \in \mathbf{X}_I$,
representations	$\mathbf{\Lambda}_I : \left(\mathbf{X}_I, \mathcal{A}_I, \mathcal{P}_{\vec{X}_I \mid \mathbf{S}_I}\right)$.

These decision problems are said to have the same formal structure, if there exist some one-to-one transformations $g : \mathbf{S}_1 \to \mathbf{S}_2$ and $h : \mathbf{A}_1 \to \mathbf{A}_2$ with corresponding inverse transformations g^{-1} and h^{-1}, such that the following statements hold true for all $s_1 \in \mathbf{S}_1$, and $a_1 \in \mathbf{A}_1$:

$$\begin{aligned} L_1(s_1, a_1) &= L_2\left(g(s_1), h(a_1)\right), \\ P_{\vec{X}_1 \mid s_1} &= P_{\vec{X}_2 \mid g(s_1)}, \end{aligned}$$

And also for all $s_2 \in \mathbf{S}_2$, and $a_2 \in \mathbf{A}_2$ it is true that:

$$\begin{aligned} L_1\left(g^{-1}(s_2), h^{-1}(a_2)\right) &= L_2(s_2, a_2), \\ P_{\vec{X}_1 \mid g^{-1}(s_2)} &= P_{\vec{X}_2 \mid s_2}. \end{aligned}$$

Based on this, the invariance principle says:

Principle 2.2.4 (Invariance Principle).

If two statistical decision problems have the same formal structure in terms of state, action, loss, and representation without prior distribution, then the same decision rule should be used in each decision problem.

The invariance principle is defined without requiring the prior to be invariant, thus from a Bayesian perspective it is not really an implementation

or the principle of rationale invariance. It can be shown that the frequentist invariance principle is fulfilled when applying the conditional Bayesian principle to statistical decision problems with certain reference (respectively non-informative) priors. For details and examples, see Berger (1995, Section 6.6).

In case of point estimation problems with a convex loss function, e.g. the quadratic loss function, we only need to consider non-randomized decision rules, as for any randomized decision rule one can find a non-randomized decision rule which is uniformly not worse. To find some uniformly best rule the set of non-randomized rules $\mathbf{\Delta}$ is further restricted to the set of the so-called *unbiased* decision rules:

Definition 2.2.4 (Unbiased Decision Rule). A non-randomized decision rule $\delta : \mathbf{X} \rightarrow \mathbb{R}$ for an estimation problem with estimand $\theta \in \Theta$ is unbiased, if

$$\mathbf{E}\left(\delta(\vec{X}) \mid \theta\right) = \theta \text{ for all } \theta \in \Theta.$$

From a frequentist perspective, this restriction makes sense, because it ensures that, for infinite replications of the experiment $\delta(\vec{X}_1)$, $\delta(\vec{X}_2)$,...,$\delta(\vec{X}_N)$,... the amounts by which the decision rule over- and underestimates θ will balance. Often, among unbiased estimators a uniformly best can be found (Lehmann, 1983).

If on the set of rules that we want to consider, no uniformly best rule exists, we have to follow other strategies to define what is considered the overall best

rule to choose. Two approaches map the two-dimensional risk function in a one-dimensional space on which optimization can be performed: The minimax principle maps via worst-case, and the Bayes risk principle via average case considerations.

Definition 2.2.5 (Minimax Decision Rule).

Let the worst risk be defined by $\sup_{s\in\mathbf{S}} R(s,\delta)$. Then, a rule $\delta_1^{*[M]}\in\Delta^*$ is a minimax decision rule, if it minimizes the worst risk on the set of all randomized rules $\mathbf{\Delta}^*$.

Principle 2.2.5 (Minimax Principle).

In any statistical decision problem always choose an action $a\in\mathbf{A}$ according to a minimax decision rule that minimizes the worst risk.

Minimax decision rules exist for many decision problems (see e.g. Ferguson, 1967), though the minimax principle may be *devilish hard to implement* (Berger, 1995, p. 309). In general, minimax rules for classification problems with zero-one loss exist, and their derivation involves solving equations with a difficulty depending on the class of distributions $\mathcal{P}_{\vec{X}|\mathbf{C}}$. For an example Berger (see 1995, chapter 5).

Most commonly, classification rules are so-called *Bayes rules*:

Definition 2.2.6 (Bayes Risk and Bayes Rule). The Bayes risk of a decision rule $\delta\in\Delta$ for a given prior distribution P_S is the average risk a statistician would experience, if she would repeatedly use this decision rule for any

potential realization of $\vec{X}$:

$$\begin{aligned} r(P_S, \delta) &= \mathbf{E}\left(R(S, \delta)\right) \\ &= \mathbf{E}_S\left(\mathbf{E}_{\vec{X}|S}\left(L(S, \delta(\vec{X}))\right)\right) \end{aligned}$$

A Bayes rule is any decision rule that minimizes this risk:

$$\delta^{[B]} = \arg\min_{\delta \in \Delta} \{r(P_S, \delta)\}$$

Principle 2.2.6 (Bayes Risk Minimization).

In any statistical decision problem with specified prior distribution P_S, always choose an action $a \in \mathbf{A}$ according to a Bayes rule minimizing the Bayes risk.

Choosing a Bayes rule only on the set of non-randomized decision rules, is no restriction, because if a randomized Bayes rule $\delta^{*[B]} \in \mathbf{\Delta}^*$ with respect to a prior distribution exists, there exists also a non-randomized Bayes rule $\delta^{[B]} \in \mathbf{\Delta}$ for this prior (cp. Ferguson, 1967, Section 1.8).

It might seem contradictious that Bayes rules are introduced as a frequentist solution to the classification problem and not as a Bayesian solution. The reason is that the characterizing difference between the Bayesian approach and the frequentist approach is whether one wants to find best decision *locally* or *globally*. The conditional Bayes principle favors local solutions – a best decision given a quantification of the current uncertainty based on all available information. And from the frequentist perspective one wants best global decisions

– a best rule prescribing the decision maker given any potential information from the statistical investigation which decision to make.

2.2.3 Optimal Solutions

The basic idea about how actions should be chosen, differ a lot between the conditional Bayesian principle and the Bayes risk principle. According to the former, we only want to decide for the best action conditional on the observed data, and according to the latter, we want to follow a decision rule that gives minimum risk for indefinitely repetitions of the statistical investigation and appendant decisions. Yet, trying to do the best locally also leads (only requiring very little assumptions) to the best global rule. The assumptions have to be such that we can apply the Fubini theorem for interchanging orders of integration for the calculation of expectations. Explicitly, the loss function has to be a non-negative measurable function with respect to $(\mathbf{S}\times\mathbf{X}, \mathcal{A}(\mathbf{S})\times\mathcal{A}(\mathbf{X}))$. If that is fulfilled, we get

$$\begin{aligned} r(P_S, \delta) &= \mathbf{E}_S\left(\mathbf{E}_{\vec{X}|S}\left(L(S, \delta(\vec{X}))\right)\right) \\ &= \mathbf{E}_{\vec{X}}\left(\mathbf{E}_{S|\vec{X}}\left(L(S, \delta(\vec{X}))\right)\right) \\ &= \mathbf{E}_{\vec{X}}\left(\rho\left(\delta(\vec{X}), P_{S|\vec{X}}\right)\right), \end{aligned} \tag{2.2.17}$$

where ρ is the Bayesian expected loss given in equation (2.2.1). If for each $\vec{x}\in\mathbf{X}$ that can be realized, we always decide for the action according to the conditional Bayesian principle 2.2.1, we minimize the argument of $\mathbf{E}_{\vec{X}}(\circ)$ in (2.2.17) for all $\vec{x}$ with positive density $p(\vec{x})$, and thus we minimize $r(P_S, \circ)$ for $\delta\in\Delta$ (see Berger, 1995, Section 4.4).

In consequence, Bayes rules are optimal solutions to a statistical classification problem with informative distribution $P_{\vec{X}|C}$ and class prior distribution P_C, both from a frequentist point of view as well as from a Bayesian point of view.

Though the basic idea about how actions should be chosen, differ a lot between the Bayesian school and the frequentist school, an assignment to classes with Bayes rules is an optimal solution of a statistical classification problem in both schools. In the following, all the statistical classification rules considered are Bayes rules with respect to zero-one loss $L^{[01]}$.

Combining the notation of a frequentist Bayes rule (2.2.17) and the Bayes assignment (2.2.9), we will write Bayes rules with respect to a pecific statistical model $\mathbf{\Lambda}$ in the following way:

$$\hat{c}(\vec{x}|\mathtt{bay}) \quad = \quad \arg\max_{c\in\mathbf{C}} \left\{ p(c|\mathbf{\Lambda}) p(\vec{x}|c,\mathbf{\Lambda}) \right\}. \tag{2.2.18}$$

These rules have lowest expected error rate on $\mathbf{X}$, which corresponds with the lowest error probability with respect to some randomly chosen object $\omega\in\mathbf{X}$. And locally for any given measurement $\vec{x}(\omega)$ it assigns to the class with the highest posterior probability.

2.3 Learning Classification Rules

In the preceding sections we have shown that Bayes rules have desirable properties in solving statistical classification problems from the Bayesian perspective as well as from the frequentist perspective — under the assumption that we can specify the full Bayesian statistical model with prior distribution P_C and

a class of informative distributions $CP_{\vec{X}|\mathbf{C}}$.

Very often, such a specification is difficult to find. Firstly, a frequentist definition of probability does not allow to code uncertainty about the class by a distribution P_C. Secondly, even more difficult it may be deemed to specify for each c, $c \in \mathbf{C}$, a precise informative distribution $P_{\vec{X}|c}$. Only in very special applications, like randomized experiments where chance is under control, a statistician of any school will feel comfortable with determining these distributions from scratch and without additional information. In all other cases, one would prefer to define all distributions P_C and $P_{\vec{X}|c}, c \in \mathbf{C}$, to belong to classes of distributions with unknown parameters, say $\left\{P_{C|\vec{\pi}}, \vec{\pi} \in \mathbf{\Pi}\right\}$ and $\left\{P_{\vec{X}|\theta_c}, c \in \mathbf{C}, \theta_c \in \Theta\right\}$. Or in other words, one specifies the joint distribution $P_{C,\vec{X}|\vec{\pi},\theta}$ with unknown parameters $\vec{\pi} \in \mathbf{\Pi}$ and $\theta \in \Theta$.

In that way, when learning classification rules, not only $\vec{X}$ but also C is modelled to be some random variable $C : \mathbf{\Omega} \to \mathbf{C}$ with probability space $(\mathbf{\Omega}, \mathcal{A}, P)$.

When the statistician has to *learn classification rules from experience* additional information about these unknown or uncertain parameters is available.

Definition 2.3.1 (Training Set).

The additional information that can be used to learn classification rules from experience consists of a number of already observed objects with known class membership, the so-called *training set* $\mathbf{L}$:

$$\begin{aligned} \mathbf{L} &:= \{(c, \vec{x})(\omega_n), \omega_n \in \mathbf{\Omega}, n = 1, ..., N_{\mathbf{L}}\}, \\ &= \{(c_n, \vec{x}_n) n = 1, ..., N_{\mathbf{L}}\}, \\ &:= \{c_n, \vec{x}_n, n = 1, ..., N_{\mathbf{L}}\}. \end{aligned}$$

The third representation allows for statements like $c_n \in \mathbf{L}$ or $\vec{x}_n \in \mathbf{L}$, which are often convenient.

In the standard statistical approach, the training set $\mathbf{L}$ is treated to be a random sample from the joint distribution $P_{C,\vec{X}|\vec{\pi},\theta}$, and the new object to assign is deemed to be another independent random draw from that distribution. This is typically not fulfilled in data mining applications (see Section 1.1), but nonetheless the default way to proceed. Of course, if one knows about certain dependencies among the objects, more sophisticated statistical models can and should be deployed, see e.g. Chapter 7.

The class of distributions from which to choose P_C is not controversial, as it is a distribution over a finite number of potential outcomes $c \in \mathbf{C}$. All potential distributions are elements of the class of multinomial distributions with unknown class probabilities $\pi_c \geq 0, c \in \mathbf{C}$ such that $\sum_{c \in \mathbf{C}} \pi_c = 1$.

Definition 2.3.2 (Unit Simplex).

The space of vectors $\vec{u}$ with non-negative elements that sum up to one is a so-called *unit simplex*, and will be denoted by $\mathbf{U}^G \subset [0,1]^G$.

When we say, the class distribution P_C is a multinomial distribution with parameter $n = 1$ for the "sample size" and parameter vector $\vec{\pi} \in \mathbf{U}^G$ for the "class probabilities" this is slightly imprecise. It is actually the random unit vector $\vec{E}_{\mathbf{C}}(C) \in \mathbf{U}^G$ that is multinomially distributed, with elements $E_c(C), c \in \mathbf{C}$

defined as

$$E_c(C) := \mathbb{I}_c(C) \quad := \quad \begin{cases} 1, & \text{if } C = c,\ c \in \mathbf{C} \\ 0 & \text{otherwise.} \end{cases} \tag{2.3.1}$$

In general, $\mathbb{I}_{\mathbf{A}} : \mathbb{R}^M \to \{0, 1\}$ denotes the *indicator function* of some set $\mathbf{A} \subseteq \mathbb{R}^M$. And we represent the multinomial distribution with parameters 1 and $\vec{\pi}$ by $P_{C|\vec{\pi}} = \mathcal{MN}(1, \vec{\pi})$.

Definition 2.3.3 (Class and Class Indicators).

Let C be the random variable $C(\circ) : \mathbf{\Omega} \to \mathbf{C}$ representing the class of some object $\omega \in \mathbf{\Omega}$ with probability space $(\mathbf{\Omega}, \mathcal{A}, P)$. The vector $\vec{E}_{\mathbf{C}}(C)$ given in (2.3.1), is a one-to-one mapping of this random variable. We call it the vector of *class indicators*.

Defining an appropriate class for the informative distributions $P_{\vec{X}|\theta_c}$ will be dependent on the application.

Definition 2.3.4 (Statistical Classification Problem with Learning).

In a statistical classification problem with learning we are given a statistical classification problem, but the representation of the information content of the measurement on the classes is incomplete: the informative distribution $P_{\vec{X}|c}$ is specified in dependence on some parameter $\theta_c \in \Theta$ with different but unknown values for the classes $P_{\vec{X}|\theta_c}$, $c \in \mathbf{C}$. Whether Bayesian or non-Bayesian, the prior class distribution P_C is specified to be a multinomial distribution with unknown class probabilities $\vec{\pi}$: $P_{C|\vec{\pi}} = \mathcal{MN}(1, \vec{\pi})$. This results in defining a

joint distribution $P_{C,\vec{X}|\vec{\pi},\theta^*}$ decomposed into $P_{C,\vec{X}|\vec{\pi},\theta^*} = \mathcal{M}\mathcal{N}(1,\vec{\pi}) \otimes P_{\vec{X}|\theta_C}$, with $\theta^* \in \Theta^G$, $\theta_C \in \Theta$. In the Bayesian approach, we quantify the uncertainty about the parameters in the distribution $P_{\vec{\pi},\theta_1,...,\theta_G}$.

The task is to assign a new object $\omega \in \mathbf{\Omega}$ to one of the classes, given these incomplete distributional assumptions and some training data consisting of $N_{\mathbf{L}}$ objects with known classes and measurements. These are assumed to be a random sample from the joint distribution $P_{C,\vec{X}|\vec{\pi},\theta^*}$.

state	the true class $c(\omega_{N_{\mathbf{L}}+1}) \in \mathbf{C}$ for the new object $\omega_{N_{\mathbf{L}}+1} \in \mathbf{\Omega}$,
action	the assigned class $\hat{c}(\omega_{N_{\mathbf{L}}+1}) \in \mathbf{C}$,
loss	the zero-one loss function $L = L^{[01]}$,
information	some measurements $\vec{x}(\omega_{N_{\mathbf{L}}+1})$,
experience	$\mathbf{L} = \{(c_n, \vec{x}_n), n = 1, ..., N_{\mathbf{L}}\}$,
representation	$\mathbf{\Lambda}$: $\left(\mathbf{C} \times \mathbf{X}, \mathcal{A}, \left\{\mathcal{M}\mathcal{N}(1,\vec{\pi}) \otimes P_{\vec{X}|\theta_C}, \vec{\pi} \in \mathbf{U}^G, \theta_C \in \Theta\right\}\right)^{N_{\mathbf{L}}+1}$, and , optionally, some prior distribution $P_{\vec{\pi},\theta^*}$.

There are two basic strategies to solve a statistical classification problem with learning:

- Decompose the task in two decision problems, one estimation problem for the parameters, and one statistical classification problem for the assignment and solve them one after the other. This results in so-called *plug-in Bayes rules*, abbreviated with `pib` . This strategy will be described in Section 2.3.1.

- Do not differentiate between "experience and information". Use both as "information", and solve the "problem with learning" as statistical

classification problem. This results in so-called *predictive Bayes rules*, abbreviated with `peb` , and presented in Section 2.3.3.

2.3.1 Plug-in Bayes Rules

In the first approach, typically one estimates θ and $\vec{\pi}$ separately according to some principle for point estimation, and then assigns to classes using the resulting so-called *plug-in* Bayes rule.

For estimating the parameters of $P_{C|\vec{\pi}}$, one counts the number of objects in the training set $\mathbf{L}$ in each class. This results in the vector of frequencies $\vec{f}_{\mathbf{L}} \in \mathbb{N}^G$, with elements defined by

$$f_{c|\mathbf{L}} \quad = \quad \sum_{n=1}^{N_{\mathbf{L}}} \mathbb{I}_c(c_n), c \in \mathbf{C}. \tag{2.3.2}$$

The $c_1, .., c_{N_{\mathbf{L}}}$ are assumed to be realizations of a random sample $C_1, ..., C_{N_{\mathbf{L}}}$ from $\mathcal{MN}(1, \vec{\pi})$, therefore the corresponding random vector of class frequencies $\vec{F} \in \mathbb{N}^G$ is distributed according to $\mathcal{MN}(N_{\mathbf{L}}, \vec{\pi})$.

1. Estimation of the parameters of the class distribution

state	the parameter $\vec{\pi} \in \mathbf{U}^G$,
action	the estimate $\vec{\pi}_{\mathbf{L}} \in \mathbf{U}^G$,
loss	some loss function $L(\circ, \circ) : \mathbf{U}^G \times \mathbf{U}^G \rightarrow \mathbb{R}$,
information	vector of observed frequencies $\vec{f}_{\mathbf{L}} \in \mathbb{N}^G$,
representation	$\mathbf{\Lambda}_1 \ : \ \left(\mathbb{N}^G, \mathcal{A}, \left\{\mathcal{MN}(N_{\mathbf{L}}, \vec{\pi}), \vec{\pi} \in \mathbf{U}^G\right\}\right)$ optionally some prior distribution $P_{\vec{\pi}}$.

2. Estimation of the parameters θ of the informative distributions:

state	the parameters $\theta_c \in \Theta \subseteq \mathbb{R}^M$, $c \in \mathbf{C}$
action	the estimates $\theta_{c|\mathbf{L}} \in \Theta$,
loss	some loss function $L(\circ, \circ) : \Theta \times \Theta \to \mathbb{R}$,
information	training set $\mathbf{L} := \{(c_n, \vec{x}_n), n = 1, ..., N_{\mathbf{L}}\}$
representation	$\mathbf{\Lambda}_2 : \left(\mathbf{X}^{N_{\mathbf{L}}}, \mathcal{A}^{N_{\mathbf{L}}}, \left\{\otimes_{n=1}^{N_{\mathbf{L}}} P_{\vec{X}|\theta_{c_n}}, \theta_{c_n} \in \Theta\right\}\right)$ optionally some prior distributions P_{θ_c}, $c \in \mathbf{C}$.

3. Assignment:

state	the true class $c(\omega_{N_{\mathbf{L}}+1}) \in \mathbf{C}$ for the given $\omega N_{\mathbf{L}}+1 \in \mathbf{\Omega}$,
action	the assigned class $\hat{c}(\omega_{N_{\mathbf{L}}+1}) \in \mathbf{C}$,
loss	the zero-one loss function $L = L^{[01]}$,
information	some measurements $\vec{x}(\omega_{N_{\mathbf{L}}+1})$,
representation	$\mathbf{\Lambda}_3 : \left(\mathbf{X}, \mathcal{A}, \left\{P_{\vec{X}|\theta_{c|\mathbf{L}}}, c \in \mathbf{C}\right\}\right)$ and $P_{C|\vec{\pi}_{\mathbf{L}}}$.

The plug-in Bayes rule for the representation $\mathbf{\Lambda} = \{\mathbf{\Lambda}_1, \mathbf{\Lambda}_2, \mathbf{\Lambda}_3\}$ is given by:

$$\hat{c}\left(\vec{x}_{N_{\mathbf{L}}+1}|\mathbf{L}, \mathbf{\Lambda}\right) = \arg\max_{c \in \mathbf{C}} \left\{p\left(c|\vec{x}_{N_{\mathbf{L}}+1}, \theta_{c|\mathbf{L}}\vec{\pi}_{\mathbf{L}}, \mathbf{\Lambda}_3\right)\right\} \tag{2.3.3}$$

$$= \arg\max_{c \in \mathbf{C}} \left\{p\left(c|\vec{\pi}_{\mathbf{L}}, \mathbf{\Lambda}_1\right) p\left(\vec{x}_{N_{\mathbf{L}}+1}|\theta_{c|\mathbf{L}}, \mathbf{\Lambda}_2\right)\right\}. \tag{2.3.4}$$

In the next section, the plug-in procedure will be exemplified by introducing two common classification methods in statistics: the linear and the quadratic discriminant analysis.

2.3.2 Example: Linear and Quadratic Discriminant Classifiers

The origin of discriminant analysis goes back to Fisher (1936), who developed a "discriminant" rule for two classes. His original justification rests upon an approach to classification as a special type of regression, with the predictors as regressor variables and the class as regressand. And the Fisher linear discriminant function maximizes the separation between classes in a least-squares sense on the set of all linear functions. Later, one revealed that the corresponding classification rule was also a plug-in Bayes rule under the assumption that the predictors are multivariate normally distributed with unequal means in the classes and equal covariance matrices. For the prove, we refer to McLachlan (1992).

The class distribution is both in linear and in quadratic discriminant analysis (LDA and QDA) modelled as being multinomial $P_{C|\vec{\pi}} = \mathcal{M}\mathcal{N}(1, \vec{\pi})$ with unknown parameters $\vec{\pi} \in \mathbf{U}^G$. The measurements $\vec{x}(\omega)$ on some object $\omega \in \mathbf{\Omega}$ are modelled as realizations of continuous random vectors — the predictors — from the space of real vectors, $\vec{X} \in \mathbf{X} = \mathbb{R}^K$. The sampling distributions $P_{\vec{X}|\theta_c}$ are multivariate normal distributions with unequal mean vectors $\vec{\mu}_c \in \mathbb{R}^K$ for objects in different classes, $c \in \mathbf{C}$. This is also called a *gaussian modelling*. LDA and QDA differ in their assumption about the covariance matrices $\mathbf{\Sigma}_c, c \in \mathbf{C}$, of the normal distributions: in QDA they may be different in the different classes, in LDA one assumes the same covariances for the distributions in each class, that is $\mathbf{\Sigma}_c \equiv \mathbf{\Sigma}$.

We implement LDA and QDA as plug-in Bayes rules. Thus, we have to do

point estimation for the parameters $\vec{\pi}$ and $\vec{\mu}_c, \mathbf{\Sigma}_c$ resp. $\mathbf{\Sigma}$, $c \in \mathbf{C}$.

1. Estimation of the parameters of the multinomial distribution

state	the parameter, $\vec{\pi} \in \mathbf{U}^G$,
action	the estimate $\vec{\pi}_{\mathbf{L}} \in \mathbf{U}^G$,
loss	quadratic loss function L^q,
information	vector of observed frequencies $\vec{f}_{\mathbf{L}} \in \mathbb{N}^G$,
representation	$\mathbf{\Lambda}_1 : \left(\mathbb{N}^G, \mathcal{A}, \left\{\mathcal{MN}(N, \vec{\pi}), \vec{\pi} \in \mathbf{U}^G\right\}\right)$ optionally some prior distribution $P_{\vec{\pi}}$,
strategy	direct computation of optimal solution with respect to quadratic loss respectively according to the maximum likelihood principle.

The parameters $\vec{\pi}$ are estimated by the observed relative frequencies on the training set $\mathbf{L}$:

$$\vec{\pi}_{\mathbf{L}} := \frac{\vec{f}_{\mathbf{L}}}{N_{\mathbf{L}}}. \tag{2.3.5}$$

The estimator $\vec{\pi}_{\mathbf{L}}$ is at the same time the maximum-likelihood estimator for $\vec{\pi}$ and the uniformly best unbiased estimator with respect to the quadratic loss (e.g. Lehmann, 1983).

2. Estimation of the parameters of the multivariate normal distributions:

state	the parameters, $\mu_c \in \mathbb{R}^K$, $\mathbf{\Sigma}_c \in \mathbb{Q}_{pd}^K$, $c \in \mathbf{C}$
action	the potential estimates $\mu_{c\|\mathbf{L}} \in \mathbb{R}^K$, $\mathbf{\Sigma}_{c\|\mathbf{L}} \in \mathbb{Q}_{pd}^K$, $c \in \mathbf{C}$,
loss	quadratic loss function L^q defined for a vector of all parameters,
information	training set $\mathbf{L} := \{(c_n, \vec{x}_n), n = 1, ..., N\}$
QDA-repr.	$\mathbf{\Lambda}_2 : \left((\mathbb{R}^K)^N, \mathcal{A}^N, \left\{\otimes_{n=1}^N \mathcal{N}(\vec{\mu}_{c_n}, \mathbf{\Sigma}_{c_n}), \vec{\mu}_{c_n} \in \mathbb{R}^K, \mathbf{\Sigma}_{c_n} \in \mathbb{Q}_{pd}^K\right\}\right)$.
LDA-repr.	$\mathbf{\Lambda}_2 : \left((\mathbb{R}^K)^N, \mathcal{A}^N, \left\{\otimes_{n=1}^N \mathcal{N}(\vec{\mu}_{c_n}, \mathbf{\Sigma}), \vec{\mu}_{c_n} \in \mathbb{R}^K, \mathbf{\Sigma} \in \mathbb{Q}_{pd}^K\right\}\right)$,
strategy	direct computation of optimal unbiased solutions according to the maximum likelihood principle.

We estimate the vector of means $\vec{\mu}_c$ by the vector of the empirical means of all objects belonging to class c on the learning set. The elements of $\vec{\mu}_{c|\mathbf{L}}$ are

given by:

$$\mu_{k,c|\mathbf{L}} := \frac{1}{f_{c|\mathbf{L}}} \sum_{n=1}^{N_{\mathbf{L}}} \left(\mathbb{I}_c(c_n) x_{k,n}\right), c \in \mathbf{C}. \tag{2.3.6}$$

The resulting estimator $\left(\vec{\mu}_{1|\mathbf{L}}, ...\vec{\mu}_{G|\mathbf{L}}\right)$ is at the same time the maximum-likelihood estimator and the uniformly minimum risk estimator of $(\vec{\mu}_1, ...\vec{\mu}_G)$.

The maximum-likelihood estimator of the covariance matrices $\mathbf{\Sigma}_c$ of the QDA-model is the empirical covariance matrix:

$$\mathbf{\Sigma}_{c|\mathbf{L}}^{[ML]} := \frac{1}{f_{c|\mathbf{L}}} \sum_{n=1}^{N_{\mathbf{L}}} \left(\mathbb{I}_c(c_n) \left(\vec{x}_n - \vec{\mu}_{c|\mathbf{L}}\right)' \left(\vec{x}_n - \vec{\mu}_{c|\mathbf{L}}\right)\right), c \in \mathbf{C}.$$

We follow the usual practice of estimating $\mathbf{\Sigma}_c$ with the unbiased estimator, the so-called *sample variance*

$$\mathbf{S}_{c|\mathbf{L}} := \frac{f_{c|\mathbf{L}}}{f_{c|\mathbf{L}} - 1} \mathbf{\Sigma}_{c|\mathbf{L}}^{[ML]} \tag{2.3.7}$$

The resulting estimator $\left(\vec{\mu}_{1|\mathbf{L}}, ...\vec{\mu}_{G|\mathbf{L}}, \mathbf{S}_{1|\mathbf{L}}, ..., \mathbf{S}_{G|\mathbf{L}}\right)$ is the uniformly minimum risk estimator of $(\vec{\mu}_1, ...\vec{\mu}_G, \mathbf{\Sigma}_1, ..., \mathbf{\Sigma}_G)$ with respect to quadratic loss (McLachlan, 1992).

The maximum-likelihood estimator of the common covariance matrix $\mathbf{\Sigma}$ of the LDA-model is the so-called *pooled within-group empirical covariance matrix*:

$$\mathbf{\Sigma}_{\mathbf{L}}^{[ML]} := \frac{1}{N_{\mathbf{L}}} \sum_{c=1}^{G} \left(f_{c|\mathbf{L}} \mathbf{\Sigma}_{c|\mathbf{L}}^{[ML]}\right).$$

We also use the unbiased variant for estimating the common covariance of the LDA-model:

$$\mathbf{S}_{\mathbf{L}} := \frac{N_{\mathbf{L}}}{N_{\mathbf{L}} - G} \mathbf{\Sigma}_{\mathbf{L}}^{[ML]}.$$

The final plug-in Bayes rule for the QDA is thus given by:

$$\hat{c}\left(\vec{x}|\mathbf{L},\texttt{qda}\right) \;=\; \arg\max_{c\in\mathbf{C}}\left\{\frac{f_{c,\mathbf{L}}}{N_{\mathbf{L}}}p\left(\vec{x}|\vec{\mu}_{c|\mathbf{L}},\mathbf{S}_{c|\mathbf{L}}\right)\right\}, \tag{2.3.8}$$

where $p\left(\circ|\vec{\mu}_{c|\mathbf{L}},\mathbf{S}_{c|\mathbf{L}}\right)$ is the density of the multivariate normal distribution:

$$p\left(\vec{x}|\vec{\mu}_{c|\mathbf{L}},\mathbf{S}_{c|\mathbf{L}}\right) = \\ (2\pi)^{-\frac{K}{2}}\det\left(\mathbf{S}_{c|\mathbf{L}}\right)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\left(\vec{x}-\vec{\mu}_{c|\mathbf{L}}\right)\mathbf{S}_{c|\mathbf{L}}^{-1}\left(\vec{x}-\vec{\mu}_{c|\mathbf{L}}\right)'\right) \tag{2.3.9}$$

The joint faces $f_{c,g}$ of the partitions between two classes for these continuous densities are given implicitly by the equations

$$p\left(c|\vec{x},\vec{\pi}_{\mathbf{L}},\vec{\mu}_{c|\mathbf{L}},\mathbf{S}_{c|\mathbf{L}}\right) \;=\; p\left(g|\vec{x},\vec{\pi}_{\mathbf{L}},\vec{\mu}_{g|\mathbf{L}},\mathbf{S}_{g|\mathbf{L}}\right), c\neq g, c, g\in\mathbf{C}.$$

The form of these borders is the origin of the names of the methods: After taking the logarithm and some other minor transformations, one can see that these implicit functions are in case of QDA quadratic functions of $\vec{x}$.

The argmax rule of LDA is the same as the argmax rule of QDA, only replacing the local sample variances by the pooled sample variances. This simplifies the implicit functions such that they can be shown to be linear functions of $\vec{x}$ (see McLachlan, 1992). For the sake of completeness, we also display the argmax rule for the LDA:

$$\hat{c}\left(\vec{x}|\mathbf{L},\texttt{lda}\right) \;=\; \arg\max_{c\in\mathbf{C}}\left\{\frac{f_{c,\mathbf{L}}}{N_{\mathbf{L}}}p\left(\vec{x}|\vec{\mu}_{c,\mathbf{L}},\mathbf{S}_{\mathbf{L}}\right)\right\}. \tag{2.3.10}$$

with

$$p\left(\vec{x}|\vec{\mu}_{c|\mathbf{L}},\mathbf{S}_{\mathbf{L}}\right) \\ =\; (2\pi)^{-\frac{K}{2}}\det\left(\mathbf{S}_{\mathbf{L}}\right)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\left(\vec{x}-\vec{\mu}_{c|\mathbf{L}}\right)\mathbf{S}_{\mathbf{L}}^{-1}\left(\vec{x}-\vec{\mu}_{c|\mathbf{L}}\right)'\right). \tag{2.3.11}$$

As McLachlan (1992, p.56) puts it, there are *no compelling reasons* for using the unbiased variants of the covariance estimators. If unbiasedness was considered important, one should use unbiased estimators of the ratio of the densities, because these determine the Bayes-rule. In his book, he presents these estimators (McLachlan, 1992).

When learning classification rules, one is not so much interested in unbiased point estimation, rather in best classifications. These would be achieved, if the distributional assumptions were fulfilled, and we would use a predictive Bayes-rule, not being based on point estimation at all.

In this work, though, we focus on typical data mining situations, where any distributional assumptions may be considerably violated, therefore theoretical justifications of the type above are all of minor importance. And standard estimators are used for convention and convenience.

2.3.3 Predictive Bayes Rules

We announced to present two statistical solutions to solve a statistical classification problem with learning. The first follows frequentist strategies to solve point estimation problems for the learning and then solves the classification problem according to the principle of Bayes risk minimization. The second strategy, the predictive Bayes rules, is an implementation of the conditional Bayes principle. It aims at the best local class assignment given the current knowledge about the situation, consisting of a specific prior uncertainty about classes and their relation to measurements on objects quantified in $P_{\vec{\pi},\theta}$, the

training data $\mathbf{L}$ that gives us additional information about that relation, and some measurement $\vec{x}(\omega)\in\mathbf{X}$ on the object $\omega\in\mathbf{\Omega}$ we have to classify.

Decision Making Given the Prior Assume for now, we had solved the problem to specify a prior distribution $P_{\vec{\pi},\theta}$. Then decision making according to the conditional Bayesian principle is straightforward. Experience and information represent the current information on which to base the decision on the classification problem:

state	the true class $c(\omega_{N_{\mathbf{L}}+1})\in\mathbf{C}$ for the given $\omega_{N_{\mathbf{L}}+1}\in\mathbf{\Omega}$,
action	the assigned class $\hat{c}(\omega_{N_{\mathbf{L}}+1})\in\mathbf{C}$,
loss	the zero-one loss function $L = L^{[01]}$,
information	learning set and measurement: $(c_n, \vec{x}_n),\ n = 1, ..., N_{\mathbf{L}}$, $\vec{x}_{N_{\mathbf{L}}+1}$,
representation	$\mathbf{\Lambda} : \left(\mathbf{C}\times\mathbf{X}, \mathcal{A}, \left\{\mathcal{M}N(1,\vec{\pi})\otimes P_{\vec{X}\mid\theta_C}, \vec{\pi}\in\mathbf{U}^G, \theta_C\in\Theta\right\}\right)^{N_{\mathbf{L}}+1}$, and some prior distribution $P_{\vec{\pi},\theta_1,...,\theta_G}$.

Applying the conditional Bayesian principle with respect to the updated model $\{\mathbf{L}, \mathbf{\Lambda}\}$, results in the following predictive Bayes rule:

$$\begin{aligned} c^{[B]}(\vec{x}|\mathbf{L},\mathbf{\Lambda}) &= \arg\max_{c\in\mathbf{C}} \{p(c|\vec{x},\mathbf{L},\mathbf{\Lambda})\} \\ &= \arg\max_{c\in\mathbf{C}} \{p(c,\vec{x}|\mathbf{L},\mathbf{\Lambda})\} \\ &= \arg\max_{c\in\mathbf{C}} \{p(c|\mathbf{L},\mathbf{\Lambda})\, p(\vec{x}|c,\mathbf{L},\mathbf{\Lambda})\} \\ &= \arg\max_{c\in\mathbf{C}} \left\{\int_{\mathbf{U}\times\Theta} p(c|\vec{\pi},\mathbf{\Lambda})\, p(\vec{x}|\theta_c,\mathbf{\Lambda})\, p(\vec{\pi},\theta_c|\mathbf{L},\mathbf{\Lambda})\, d\mu(\vec{\pi},\theta_c)\right\}. \end{aligned}$$

Comparing the last line with the plug-in Bayes rule exposes the main difference between the two approaches. With the plug-in Bayes rule, one assigns to

the class with the highest probability computed on the basis of individual (best) estimates of the unknown parameters. Whereas the predictive Bayes rule assigns to the class with the highest expected probability on the basis of the a-posteriori distribution of the parameters. Asymptotically (that is for $N \to \infty$) "good" point estimators converge to the true parameters, and in most cases, the posterior distribution converges to a normal distribution with all mass arbitrarily close to the true parameter. Therefore, for large N, the assignments with both approaches will not be too different.

Specifying the Prior

The open problem is, how to specify the prior distributions. In the following we will, as is commonly done, assume that prior to the investigation the uncertainty about $\vec{\pi}$ and the θ_c, $c \in \mathbf{C}$ was independent. In addition, the prior uncertainty of the parameter θ_c is assumed to be the same for all $c \in \mathbf{C}$. Then the prior distribution can be factorized $P_{\vec{\pi},\theta_1,\ldots,\Theta_G} = P_{\vec{\pi}} \otimes P_{\theta}^{G}$. In consequence, the updating of the distribution of the parameters can also be decomposed:

$$\begin{aligned} p\left(\vec{\pi},\theta|\mathbf{L},\mathbf{\Lambda}\right) &= \frac{p\left(\mathbf{L}|\vec{\pi},\theta\right)p\left(\vec{\pi},\theta|\mathbf{\Lambda}\right)}{k(\mathbf{L})} \\ &= \frac{1}{k(\mathbf{L})}\prod_{n=1}^{N} p\left(c_n,x_n|\vec{\pi},\theta\right)p\left(\vec{\pi}|\mathbf{\Lambda}\right)p\left(\theta|\mathbf{\Lambda}\right) \\ &= \frac{1}{k(\mathbf{L})}\left(\prod_{n=1}^{N} p\left(c_n|\vec{\pi}\right)p\left(\vec{\pi}|\mathbf{\Lambda}\right)\right)\left(\prod_{n=1}^{N} p\left(x_n|\theta_{c_n}\right)p\left(\theta_{c_n}|\mathbf{\Lambda}\right)\right). \end{aligned}$$

This is often very helpful for the computation. It is not necessary to calculate the constant term $k(\mathbf{L})$, as it has no influence on the relative size of $p\left(c|\vec{x},\mathbf{L},\mathbf{\Lambda}\right)$ among the classes, and thus no influence on the final assignment into one of

the classes.

For defining the form of the prior distributions, we will use standard *natural conjugate* prior modelling.

Definition 2.3.5 (Family of natural conjugate priors).

Let $\mathcal{F} := \left\{p(\circ|\theta) : \mathbf{X} \to \mathbb{R}_0^+, \theta \in \Theta\right\}$ define the densities of the class of informative distributions, and $\mathcal{P} := \left\{p(\circ) : \theta \to \mathbb{R}_0^+\right\}$ a class of prior densities for $\theta \in \Theta$. Then the class $\mathcal{P}$ is called conjugate for $\mathcal{F}$ if

$$p(\theta|\vec{x}) \in \mathcal{P} \text{ for all } p(\circ|\theta) \in \mathcal{F} \text{ and } p(\circ) \in \mathcal{P}.$$

This definition is vague so far, as e.g. the class of all distributions is conjugate to any class of sampling distributions. If in addition, the densities belonging to $\mathcal{P}$ are of the same functional form as the class of likelihood functions arising from $\mathcal{F}$, then $\mathcal{P}$ is called the *natural conjugate prior family* to $\mathcal{F}$.

With natural conjugate priors, not only the tractability of Bayesian updating is highly improved, but also the effects of prior and sample information can be easily understood: we start with a certain functional form for the prior, and we end up with a posterior of the same functional form, but with parameters updated by the sample information.

Assume the class of prior distributions $\mathcal{P}_{\theta|\Psi} := \left\{p(\circ|\psi) : \Theta \to \mathbb{R}_0^+, \psi \in \Psi\right\}$ to be uniquely parameterized by parameters $\psi \in \Psi$. If $\mathcal{P}_\psi$ is the natural conjugate prior for the class of informative distributions

$$\mathcal{P}_{\vec{X}|\Theta} := \left\{p(\circ|\theta, \mathbf{\Lambda}) : \mathbf{X} \to \mathbb{R}_0^+, \theta \in \Theta\right\}$$

according to some model $\mathbf{\Lambda}$ then for any $\vec{x}\in\mathbf{X}$ and $\psi\in\Psi$ there exists some $\psi_{\vec{x}}\in\Psi$ such that the posterior distribution is a member of $\mathcal{P}_{\theta|\Psi}$. In terms of the update mechanism given in equation (2.2.2) this reads:

$$p(\theta|\vec{x},\mathbf{\Lambda}) = \frac{p(\vec{x}|\theta,\mathbf{\Lambda})p(\theta|\psi,\mathbf{\Lambda})}{p(\vec{x}|\mathbf{\Lambda})} \tag{2.3.12}$$

$$= p(\theta|\psi_{\vec{x}},\mathbf{\Lambda}) \tag{2.3.13}$$

This is not only useful for tracing the influence of the prior, but also the prior can be translated into an *equivalent sample* that carries the same information. That helps determining an appropriate prior. You will see an example in the following subsection. A general outline involves the class of regular exponential distributions, canonical parameters and the relationship with sufficient statistics. We refer the interested reader to Bernardo and Smith (1994, Section 5.2.2).

Specifying the Prior for $\vec{\pi}$

Bayesian modelling of the informative distribution and the corresponding prior distribution varies for predictive Bayes rules – suitable solutions are highly dependent on the given task. Common is the problem to model the class distribution and the corresponding prior. We will present in the following a minimal informative conjugate modelling thereof.

Remember, $P_{C|\vec{\pi}}$ is a multinomial distribution with class probabilities $\pi_c \geq 0, c\in\mathbf{C}$ such that $\sum_{c\in\mathbf{C}}\pi_c = 1$.

The natural conjugate prior family for the multinomial model is the class of Dirichlet priors $\mathcal{D}(\vec{\alpha})$ with $\vec{\alpha}\in(\mathbb{R}^+)^G$. This *multinomial-Dirichlet* model will

play an important role later in standardized partition spaces (Chapter 5, thus it will be described below quite detailed.

By comparing the density of the multinomial distribution $\mathcal{MN}(N, \vec{\pi})$, and the Dirichlet distribution $\mathcal{D}(\vec{\alpha})$, you see what is meant by requiring a natural conjugate prior to have the same functional form as the likelihood function:

$$p(\vec{n}|\vec{\pi}) = \frac{\Gamma(N_{\mathbf{L}}+1)}{\prod_{c=1}^{G}\Gamma(n_c+1)}\prod_{c=1}^{G}\pi_c^{n_c}, \vec{\pi}\in\mathbf{U}^G, \tag{2.3.14}$$

$$p(\vec{\pi}|\vec{\alpha}) = \frac{\Gamma(\alpha_0)}{\prod_{c=1}^{G}\Gamma(\alpha_c)}\prod_{c=1}^{G}\pi_c^{\alpha_c-1}, \tag{2.3.15}$$

with $\vec{n}\in\mathbb{N}^G$, $\sum_{c=1}^{G} n_c = N_{\mathbf{L}}$, $\vec{\pi}\in\mathbf{U}^G$ $\vec{\alpha}\in(\mathbb{R}^+)^G$ $\sum_{c=1}^{G}\alpha_c = \alpha_0$, and $\Gamma(\circ)$ denoting the Gamma function. It is obvious that the parameter vector $\vec{\alpha}$ plays the role of the frequency vector $\vec{n}$.

Updating the prior with the observed class frequencies $n_{1|\mathbf{L}}, ..., n_{G|\mathbf{L}}$ in the data $\mathbf{L}$, the elements of the parameter vector of the posterior Dirichlet distribution are:

$$\alpha_{c|\mathbf{L}} = \alpha_c + f_{c|\mathbf{L}},\ c\in\mathbf{C}.$$

In other terms, $\mathcal{D}(\vec{\alpha}|\mathbf{L})=\mathcal{D}\left(\vec{\alpha}+\vec{f_{\mathbf{L}}}\right)$.

The relationship between prior and posterior is easy to understand: the Dirichlet distribution gathers the information about $\vec{\pi}$ in the vector of so-far "imagined" ($\vec{\alpha}$) and observed frequencies ($\vec{f_{\mathbf{L}}}$) of the classes in the data, $\vec{\alpha}+\vec{f_{\mathbf{L}}}$. The parameters $\alpha_c, c\in\mathbf{C}$ embody the same information on $\vec{\pi}$ as an sample of α_0 outcomes of the experiment with observed frequencies $\alpha_c, c\in\mathbf{C}$ does. Therefore α_0 is the so-called *equivalent sample size* (ESS) of the prior uncertainty.

The mean vector and the covariances of the posterior Dirichlet distribution

are:

$$\begin{aligned}\mathbf{E}\left(\vec{\pi}|\vec{\alpha},\mathbf{L}\right) &= \frac{\vec{\alpha}+\vec{f_{\mathbf{L}}}}{\alpha_0+N_{\mathbf{L}}} := \vec{p}_{\mathbf{L}},\\ \mathbf{\Sigma}\left(\vec{\pi}|\vec{\alpha},\mathbf{L}\right) &:= \left[\sigma_{c,g|\mathbf{L}}\right]_{c,g\in\mathbf{C}},\end{aligned}$$

with

$$\begin{aligned}\sigma_{c,c|\vec{\alpha},\mathbf{L}} &= \frac{p_{c,\mathbf{L}}(1-p_{c,\mathbf{L}})}{1+N_{\mathbf{L}}+\alpha_0} \text{ and}\\ \sigma_{c,g|\vec{\alpha},\mathbf{L}} &= \frac{-p_{c,\mathbf{L}}p_{g,\mathbf{L}}}{1+N_{\mathbf{L}}+\alpha_0},\ g\neq c,\ c,g\in\mathbf{C}.\end{aligned}$$

Sometimes, one wants to know the marginal distribution of the sample entity, in this case C, given the prior with parameters $\vec{\alpha}$ and the training data $\mathbf{L}$:

$$p(c|\mathbf{L},\vec{\alpha},\mathbf{\Lambda}) = \int_{\mathbf{U}}\left(p(c|\vec{\pi},\mathbf{\Lambda})p(\vec{\pi}|\mathbf{L},\vec{\alpha},\mathbf{\Lambda})d\mu(\vec{\pi})\right).$$

This distribution is called the posterior predictive distribution. As in case of the multinomial distribution, the random entity for which the posterior distribution is described is not really C but rather the vector of class indicators $\vec{E}(C)\in\mathbf{U}^G$ (Definition 2.3.3).

In general, the marginal distribution of some frequency vector $\vec{F}$ given $\mathbf{L}$ and $\vec{\alpha}$ in this setting is a multinomial-Dirichlet distribution $\mathcal{M}d\left(F_\bullet,\vec{\alpha}+\vec{f_{\mathbf{L}}}\right)$ that differs from the multinomial distribution $\mathcal{M}N\left(F_\bullet,\frac{\vec{\alpha}+\vec{f_{\mathbf{L}}}}{\alpha_0+N_{\mathbf{L}}}\right)$, mainly by having larger (absolute) values for the variances and covariances.

Yet, for the special case of only one new trial, namely the distribution of

$\vec{E}(C)$, the multinomial-Dirichlet distribution $\mathcal{M}d\left(1, \vec{\alpha} + \vec{f}_{\mathbf{L}}\right)$ is also a multinomial distribution $\mathcal{M}N\left(1, \frac{\vec{\alpha}+\vec{f}_{\mathbf{L}}}{\alpha_o+N_{\mathbf{L}}}\right)$. That is, the posterior predictive distribution is defined by the probabilities:

$$p(c|\mathbf{L}, \mathbf{\Lambda}) \;=\; \frac{\alpha_c + f_{c|\mathbf{L}}}{\alpha_0 + N_{\mathbf{L}}},\ c \in \mathbf{C}. \tag{2.3.16}$$

This results in the following central moments (Bernardo and Smith, 1994, Appendix)):

$$\mathbf{E}\left(\vec{E}|\vec{\alpha}+\vec{f}_{\mathbf{L}}\right) \;=\; \frac{\vec{\alpha}+\vec{f}_{\mathbf{L}}}{\alpha_0+N_{\mathbf{L}}} := \vec{p}_{\mathbf{L}}, \tag{2.3.17}$$

$$\mathbf{\Sigma}\left(\vec{E}|\vec{\alpha}+\vec{f}_{\mathbf{L}}\right) \;:=\; \left[\sigma_{c,g|\mathbf{L}}\right]_{c,g\in\mathbf{C}},\ \text{with} \tag{2.3.18}$$

$$\sigma_{c,c} \;=\; p_{c|\mathbf{L}}(1 - p_{c|\mathbf{L}})\ \text{and} \tag{2.3.19}$$

$$\sigma_{c,g} \;=\; p_{c|\mathbf{L}} p_{g|\mathbf{L}},\ g \neq c,\ c, g \in \mathbf{C} \tag{2.3.20}$$

Now assume, our training data was appropriately modelled to the the realization of some some i.i.d. sample from that multinomial class distribution and some sampling distribution. For example, let $\omega \in \mathbf{\Omega}$ be the realization of some random draw from some population, where $\vec{\pi}_{\mathbf{\Omega}}$ denotes the vector of relative frequencies of objects in the classes in this population. Then, for $N_{\mathbf{L}} \to \infty$, the observed relative frequencies converge to the true relative frequencies, and so do the elements of the mean vector of the Dirichlet distribution, and the (co)-variances converge to zero:

$$\frac{f_{c,\mathbf{L}}}{N_{\mathbf{L}}} \to \pi_{c,\mathbf{\Omega}},\ \frac{\alpha_c + f_{c,\mathbf{L}}}{\alpha_o + N_{\mathbf{L}}} \to \pi_{c,\mathbf{\Omega}},\ \sigma_{c,g|\mathbf{L}} \to 0, c, g \in \mathbf{C}. \tag{2.3.21}$$

In other words, as it should be, with more and more independent examples, our uncertainty about the true parameter of the class distribution vanishes, and gets centered around the true parameter.

Form of the conjugate prior and update mechanism have been explained and specified above, so the final decision is on the priors' parameters. Standard minimal informative priors are achieved by setting all $\alpha_c, c \in \mathbf{C}$, either to be equal to one, $\vec{\alpha} := \vec{1}$, or equal to zero, $\vec{\alpha} := \vec{0}$. In the former the prior distribution assigns equal density to all $\vec{\pi} \in \mathbf{U}^G$, in the latter uniform *improper* density is assigned to the vector of logarithms of the elements of $\vec{\pi}$ (Gelman et al., 1995). The density is improper, as the integral over this density is infinity, which violates the probability assumptions. Many minimal informative priors are improper, which is not deemed a problem, as long as one takes care that the resulting posterior is proper.

For the two priors under consideration, the updated posterior is either $\mathcal{D}\left(\vec{f}_{\mathbf{L}}\right)$ or $\mathcal{D}\left(\vec{f}_{\mathbf{L}} + \vec{1}\right)$. The more objects are presented in the training data, the less important this difference is for any class assignment as this depends on relative frequencies. With small data sizes and with $\vec{\alpha} = \vec{0}$, we may run into technical difficulties: for the posterior of the prior of zeros to be proper, at least one object in each group has to be observed. That is the main reason, why we in general prefer $\vec{\alpha} = \vec{1}$. The critical question is, whether the defined classes are such that we are sure there are objects of any class in the population. If according to our prior knowledge, we can be sure of that, but we have no (justifiable) idea, that any class should have a higher frequency than another, then it seems appropriate to model this with an imaginative sample of one object in each class.

In the preceding paragraphs we have given an outline of general Bayesian conjugate modelling and a detailed description of the multinomial-Dirichlet

model. In the next section, we will present you a full description of a specific predictive Bayes rule that exemplifies the complete procedure of predictive Bayes rules.

2.3.4 Example: Continuous Naïve Bayes Classifier

The naïve Bayes classifier is based on distributional assumptions between the class and observed facts on some object, and thus a statistical classification rule. Yet, it is mainly used in information retrieval, a branch of computer science. The use of the naïve Bayes in information retrieval can be traced back to the early 60ties (Maron and Kuhns, 1960; Maron, 1961) for automatic text categorization, and it is still the benchmark classifier in this field. For an overview Lewis (see 1998). Typically, the observed facts — words in a text — are modelled with high-dimensional vectors of binary random variables, but in our applications we use a continuous — namely gaussian — modelling. Therefore we call the resulting classifier a continuous naïve Bayes classifier (CNB).

As usually, the class distribution is modelled as being multinomial $P_{C|\vec{\pi}} = \mathcal{MN}(1, \vec{\pi})$ with unknown parameters $\vec{\pi} \in \mathbf{U}^G$.

Given the class, predictor variables are assumed to be stochastically independent and normally distributed with unequal and unknown means and variances, $P_{\vec{X}|c} = \mathcal{N}(\nu_c, \mathbf{\Sigma}_c)$, $c \in \mathbf{C}$. So far, the representation is the same as for QDA. It is different, as we impose a restriction on the set of considered covariance matrices: all $\mathbf{\Sigma}_c$ have to be diagonal matrices, $\mathbf{\Sigma}_c \in \mathbb{Q}_{[D]}^K$, $c \in \mathbf{C}$.

By this, we assume for given class $c \in \mathbf{C}$ the elements $X_1, ..., X_K$ of the K-dimensional vector $\vec{X}$ to be independently distributed, each according to some univariate normal distribution $P_{X_k|c} = \mathcal{N}\left(\mu_{k|c}, \sigma^2_{k|c}\right)$, $k = 1, ..., K$, $c \in \mathbf{C}$. This is the independence assumption that characterizes all naïve Bayes classifiers, whatever the type (normal, exponential, Bernoulli, multinomial,...) of the univariate sampling distributions $P_{X_k|c}$, $k = 1, ..., K$, $c \in \mathbf{C}$ may be. The assignment according to the conditional Bayesian principle can thus be factorized in $K + 1$ factors:

$$\begin{aligned} \hat{c}(\vec{x}|\texttt{cnb}) &= \arg\max_{c \in \mathbf{C}} \{p(c|\texttt{cnb}) p(\vec{x}|c, \texttt{cnb})\} \\ &= \arg\max_{c \in \mathbf{C}} \left\{ p(c|\texttt{cnb}) \prod_{k=1}^{K} p(x_k|c, \texttt{cnb}) \right\}. \end{aligned}$$

Each of the factors can be quite easily learnt from the training data. The name can be understood in the light of this simplifying effect of the independence assumption, when one uses it although one knows, it is not really valid. Quite nicely, it often works despite that fact!

We will use the assumptions of the continuous naïve Bayes classifier to construct a predictive Bayes rule. Thus, in addition to the distributional assumptions stated above, we need to define the prior distribution for the parameters $\vec{\pi}$,$\mu_{k,c}, \sigma_{k|c}$, $k = 1, ..., K$, $c \in \mathbf{C}$. In a first step, we decompose this distribution in independent distributions for $\vec{\pi}$, and for each pair $(\mu_{k|c}, \sigma_{k|c})$, $k = 1, ..., K$, $c \in \mathbf{C}$.

We set the prior for the distribution $P_{\vec{\pi}}$ of the parameters of the multinomial distribution $\vec{\pi}$ to be a Dirichlet distribution $\mathcal{D}\left(\vec{1}\right)$. This results in a posterior

predictive distribution $P_{C|\mathbf{L},\vec{\alpha}}$ being also multinomial with probabilities

$$p(c|\mathbf{L},\mathbf{\Lambda}) \;\; = \;\; \frac{\alpha_c + f_{c|\mathbf{L}}}{\alpha_0 + N_{\mathbf{L}}},\; c \in \mathbf{C}.$$

This has been shown in the preceding subsection 2.3.3.

For each pair $(\mu_{k|c}, \sigma^2_{k|c})$, $k = 1, ..., K$, $c \in \mathbf{C}$ we set up the standard minimal informative conjugate prior for the univariate normal distribution as presented in Gelman et al. (1995). Dropping the class and variable indices for now, the joint distribution of (μ, σ^2) is factorized as $P_\sigma \otimes P_{\mu|\sigma}$. P_σ is the scaled inverse χ^2 distribution with parameter $\nu_0 \in \mathbb{R}$ for the degrees of freedom and $\sigma_0^2 \in \mathbb{R}^+$ for the scale, denoted by $\mathbf{inv}\chi^2(\nu, \sigma_0^2)$ and the conditional distribution $P_{\mu|\sigma^2}$ is some normal distribution with parameters $\mu_0 \in \mathbb{R}$ and $\frac{\sigma^2}{\kappa_0} \in \mathbb{R}^+$.

The minimal informative starting prior is an improper distribution uniform on $(\mu_{k|c}, \log(\sigma^2_{k|c}))$ or, equivalently,

$$p(\mu_{k|c}, \sigma^2_{k|c}) \;\; \propto \;\; (\sigma^2_{k|c})^{-1}.$$

The joint posterior distribution $p(\mu_{k|c}, \sigma^2_{k|c}|\mathbf{L})$ is proportional to the likelihood function multiplied by the factor σ^{-2}:

$$p\left(\mu_{k|c}, \sigma^2_{k|c}|\mathbf{L}\right) \;\; \propto \;\; \sigma_{k|c}^{-n-2} \exp\left(-\frac{(N_{\mathbf{L}}-1)\mathbf{S}^2_{k|c,\mathbf{L}} + N_{\mathbf{L}}(\mu_{k|c,\mathbf{L}} - \mu_{k|c})}{2\sigma^2_{k|c}}\right),$$

where $\mu_{k|c,\mathbf{L}}$ and $\mathbf{S}^2_{k|c,\mathbf{L}}$ are the univariate realizations of the empirical mean and the sample variance as defined in (2.3.6) and (2.3.7).

The resulting posterior predictive distribution $p(x_k|c, \mathbf{L}, \texttt{cnb})$ is a Student-t-distribution $\mathcal{T}\left(\mu_{k|c,\mathbf{L}}, \sqrt{1+\frac{1}{N_{\mathbf{L}}}}\mathbf{S}_{k|c,\mathbf{L}}, f_{c|\mathbf{L}}-1\right)$ with $f_{c|\mathbf{L}}-1$ degrees of freedom, location $\mu_{k|c,\mathbf{L}}$, and scale $\sqrt{1+\frac{1}{f_{c|\mathbf{L}}}}\mathbf{S}_{k|c,\mathbf{L}}$, $k = 1, ..., K, c \in \mathbf{C}$.

The final predictive Bayes rule is given by

$$\hat{c}(\vec{x}|\mathbf{L}, \texttt{cnb}) \quad = \quad \arg\max_{c \in \mathbf{C}} \left\{ \frac{\alpha_c + f_{c|\mathbf{L}}}{\alpha_0 + N_{\mathbf{L}}} \prod_{k=1}^{K} p(x_k|c, \mathbf{L}, \texttt{cnb}) \right\}, \qquad (2.3.22)$$

with

$$p(x_k|c, \mathbf{L}, \texttt{cnb}) \quad = \quad h(f_{c|\mathbf{L}}, \mathbf{S}_{k|c,\mathbf{L}}) \left(1 + \frac{1}{f_{c|\mathbf{L}}} \left(\frac{x_k - \mu_{k|c,\mathbf{L}}}{\mathbf{S}_{k|c,\mathbf{L}}} \right) \right)^{\frac{f_{c|\mathbf{L}}+1}{2}},$$

where h is some appropriate function. Important to notice in the equation above is the dependency of the exponent only on the observed frequencies of objects in the classes $f_{c|\mathbf{L}}$. In consequence, the functions representing the joint faces of the partitions of $\mathbf{X}$ can in general not be simplified to be quadratic forms, only if $\mathbf{L}$ is a balanced sample in each class. Nevertheless, pictures of the resulting joint faces resemble much those of the plug-in QDA, where the joint faces are quadratic forms.

Chapter 3

Machine Learning

We embed the learning of classification rules in the corresponding machine learning framework of concept learning. A comparable representation of methods for concept learning and classification rule learning is developed. We present various aspects of performance that are important for concept learning, and different theoretical frameworks that define good learning in these terms.

3.1 Basic Definitions

As learning is the main concern of machine learning, defining what it means is a crucial point. Typically one defines learning in a broad sense such that it covers most intuitive ideas people have about learning. According to Langley (1995) this reads as follows:

Definition 3.1.1 (Learning).

Learning is the improvement of performance in some environment through the acquisition of knowledge resulting from experience in that environment.

Defining learning in dependence of the improvement of performance is not mandatory and is questioned e.g. by Michalski (1986). Such a definition is exclusive for learning incidences, where learning happens without a predefined goal – and such a predefinition is sometimes difficult to state in data mining in general. Yet, for classification rule learning, the difficulty is not to define some performance criterion, but rather that the most common definition of performance – correctness – might not cover important aspects of performance one might want to improve. In Chapter 5 we will introduce performance aspects that go beyond the correctness. That is, we do not question the importance of performance for classification rule learning, we broaden the notion of performance, therefore, Definition 3.1.1 is a good choice for defining learning for our purposes. The initial point for machine learning solutions to classification tasks is concept learning. Mitchell (1997) defines concept learning as follows:

Definition 3.1.2 (Concept learning).

Concept learning is the problem of automatically inferring the general definition of some concept, given examples labelled as members or nonmembers of the concept. In other words, concept learning is inferring a boolean-valued function from training examples of its input and output.

Based on this definition, we can now compare the general set up of classification rule learning with concept learning: In the Definition 2.1.5 of the statistical classification problem, we distinguish between the object ω and the measurements on the object $\vec{x}(\omega)$, the class $c(\omega)$ being deemed some other

characteristic of the object. This is common in statistics and uncommon in machine learning. Indirectly, this already says the object ω is more than its description $\vec{x}(\omega)$, so that the idea of a deterministic relationship between the object's class $c(\omega)$ and its observations $\vec{x}(\omega)$ is immediately questionable, as we may lack valuable information about the object.

Quite differently, the starting point in machine learning is the inference of some concept. Other than the term class, the term concept relates directly to an *idea of something by combining all its characteristics or particulars; a construct* (Webster's Dictionary, 1994). (Just as a philosophical note in the margin: obviously one can doubt, whether "class" can ever be a characteristic of some object in the world or whether it always will be some concept humans have about objects in the world.) If the examples in $\mathbf{L} = \{(c_n, \vec{x}_n), n = 1, ..., N_{\mathbf{L}}\}$ (see Definition 2.3.1) are optimally chosen, each input $\vec{x}_n$ should provide all information about the combined "characteristics and particulars" of the associated concept with number $c_n, n = 1, ..., N_{\mathbf{L}}$. And in consequence, the assumption of a probabilistic relationship appears strange.

Relating concepts and their definitions to a statistical investigation: what would happen, if all relevant information about the class membership was present in $\vec{x}$? Then, for the classification task, there is no need to distinguish the object ω and the information $\vec{x}$. Therefore, we will equate the "optimal description" with the "object". In data mining applications, the assumption of optimal descriptions is most commonly relaxed to allow for noise corrupting the input and also — potentially — the output. (Throughout this work we will not assume the latter). To allow for noise we equate the non-optimal,

actually available descriptions with the observed measurements on the object $\vec{x}\in\mathbf{X}$. We distinguish "ideal concepts" from "realizable concepts":

Definition 3.1.3 (Ideal and Realizable Concepts). We call concepts defined in terms of some optimal descriptions $\omega\in\mathbf{\Omega}$ *ideal concepts*, and we will represent them by their boolean functions $e_c(\circ|\mathbf{\Omega}), c\in\mathbf{C}$.

Concepts in terms of potentially incomplete or corrupted, yet available, descriptions $\vec{x}\in\mathbf{X}$, are called *realizable concepts*. The corresponding boolean functions are denoted by $e_c(\circ|\mathbf{X}), c\in\mathbf{C}$.

In the notation of certain values of these functions, we drop the identifying domain, if the instance symbol is not ambiguous, as e.g. in $e_c(\vec{x})$ or $e_c(\omega)$, $c\in\mathbf{C}$.

In Section 2.3 we presented Bayes rules as the goal of statistical classification rule learning. We argued at the end of Section 2.2.2 that randomized Bayes rules can not be better than deterministic Bayes rules, and restricted the optimization to be performed only on the set of non-randomized rules. That is, though statisticians model a non-deterministic relationship between the observation $\vec{x}(\omega)$ and the class $c(\omega)$, the learnt classification rule $\hat{c}(\circ|\mathbf{L}, \mathtt{bay}) : \mathbf{X}\rightarrow\mathbf{C}$ is deterministic. In that sense, statistical classification rule learning is an instance of concept learning as stated in Definition 3.1.2.

[**Notation**] As it is common in machine learning, in Definition 3.1.2 concepts are formally defined to be boolean-valued functions, and their domain is equated with the "input space" $\mathbf{X}$. For the reasons given above, we define the domain of ideal concepts to be the "optimal input

space" $\mathbf{\Omega}$. In the statistical context we already saw that for the learning of classification rules, classes are modelled to be random variables (Section 2.3). That is, they are also functions with domain $\mathbf{\Omega}$, the only difference being that these functions correspond to a probability space $(\mathbf{\Omega}, \mathcal{A}, P)$. We assimilated the statistical and the machine learning's representation regarding the domain.

In this work we will mainly consider multi-class problems, where each object belongs to one and only one class of a set of classes. In Langley (1995) this is called the *competitive framework for concepts*, where an optimal description can be an instance of one and only one out of a finite number of competing concepts. We defined the counterdomain of the class to be $\mathbf{C} = \{1, ..., G\}$, or alternatively, using the dual representation with the vector of class indicators $\vec{e}_{\mathbf{C}}$ the counterdomain is $\mathbf{U}^G$ (cp. 2.3.3). In the same way, we define the set of competing concepts to be numbered by $c \in \mathbf{C} = \{1, ..., G\}$, and we denote by $\vec{e}_{\mathbf{C}} \in \mathbf{U}^G$ the vector of the logical values of the corresponding boolean functions.

This assimilates statistical and machine learning's representation with respect to the counterdomain. In Table 3.1 you find the synopsis of the assimilated notation for concept and classification rule learning.

The Definition 3.1.1 of learning rests upon four rather vague terms, namely performance, environment, knowledge, and experience. This vagueness is intended for the definition to be broad. In the following sections we will clarify the notion of these terms in the context of this work:

In Section 3.2 we acquire the environmental factors of concept learning as they emerge from our definition of a classification problem and from the characteristics of data mining applications.

In Section 3.3 we devise a structure for comparable representations of methods for concept learning and classification rule learning.

In Section 3.4 we present various aspects of performance that are important for concept learning, and different theoretical frameworks that define good

Symbol	Statistics	Machine Learning
$\mathbf{\Omega}$	population	optimal input space
$\omega \in \mathbf{\Omega}$	object	optimal description
$\mathbf{C}$	set of classes	set of concept numbers
$c(\omega)$	class of object ω	number of the ideal concept of which ω is an instance
$e_c(\circ\|\mathbf{\Omega})$	class indicator function for class c	ideal concept number c
$\vec{e}_{\mathbf{C}}(\circ\|\mathbf{\Omega})$	vector of class indicator functions	vector of ideal competing concepts
$\mathbf{X}$	predictor space	input space
$\vec{x}$	observed measurement of predictors	non-optimal description
$e_c(\circ\|\mathbf{X})$	class indicator function for assignments to class c	realizable competing concept number c
$\vec{e}_{\mathbf{C}}(\circ\|\mathbf{X})$	vector of class indicator functions	vector of realizable competing concepts
$c(\circ\|\mathbf{X})$	classification rule	matching rule for numbers of realizable competing concepts

Table 3.1: Assimilated notation

learning in these terms.

And finally, in Section 3.5 we present ways how to do a good job in learning good concepts.

3.2 Environment

Among other things, the environment determines the task, aspects of the performance criterion, the amount of supervision, whether the learning system has to learn online or offline, and the strength of the regularities on which the learning system can base its learning (cp. Langley, 1995).

Some aspects of this environment are fixed by the main topic of this work:

The classification problem as posed in Definition 1.3.2 determines the task and partly the related performance criterion: the zero-one loss with respect to the correctness of the classification of some presented object. As we have seen in the development of the statistical approaches, though, there are different ways to extent this concrete, individual loss to an "expected" loss with respect to the correctness of the classification of some object with unknown class.

The experience is presented in terms of a set of objects with known classes (see Definition 2.3.1). This constitutes the manner of learning to be completely supervised and offline.

In data mining applications, the strength of the regularities is one of the big questions. Learning concepts is one tool utilized in the larger context of the knowledge discovery process, where one tries to find out about regularities and their strengths in the database. Particularly tricky about the regularity question is that it is only partly predetermined by the domain and the data.

The perceived strength of regularity also depends on the language used to describe the regularities.

An important characteristic of regularities is whether they are probabilistic or deterministic. In their assumptions about this, approaches in statistics and machine learning differ the strongest. Statisticians are trained to think in terms of (genuinely) probabilistic relationships. One way to explain probabilistic relationships is to assume a random sample of objects from some population. If the entities among which we want to reveal relationships are characteristics of these objects, this random draw induces a probabilistic relationship, including deterministic relationships as special cases.

This approach stands in contrast to the cultural habit in the machine learning community. As one can see in Definition 3.1.2, starting point is typically some deterministic relationship, which is — if necessary — softened to allow for *noise*. As the potential causes of noise, one regards e.g. corrupted supervised feedback or input description (cp. Langley (1995)).

3.3 Representation

The designer of a learning system has to decide upon the representation of the experience and the acquired knowledge. With these choices, one determines the capacity of the learning system, and of course this has an enormous influence on the potential performance of the system.

Determining the Experimental Unit In the classification problem, the first choice is about the unit that defines what an individual example is. In

Definition 1.3.2 this unit is represented by the "object" ω. Chapter 7 is devoted to an application, where this unit is defined in two different ways, and where the change of the unit leads to considerably better predictions.

Representation of Individual Examples Once it is clear what a single example is, to represent its characteristics we can employ

a) binary features, denoting the absence or presence of a certain feature,
b) nominal attributes that are similar to binary features, only that they allow for more than two mutually exclusive features,
c) numeric attributes that take on real, integral, or ordinal values, or
d) relational literals that subsume featural, nominal, and numeric schemes.

The last representation is used for tasks that are inherently relational in nature and require more sophisticated formalisms to be processed — like first-order logic instead of propositional logic. The standard statistical framework is based on propositional logic only, so that in this work relational literals play no role. The consequences of the restriction to propositional logic in statistics can be best understood in the motivation of current research developing systems to overcome the deficiencies of it (Jaeger, 1997).

The class $c \in \{1, ..., G\}$ is obviously represented by a nominal attribute. In general, the characteristics used to determine the class in a classification problem can be represented by binary or nominal features, as well as by numeric attributes. In the examples and applications in this work, though, these characteristics are always represented by numeric attributes. This reflects the statistical background of the author. Numeric representations — and applications where this is appropriate — are most common in statistics.

[**Notation**] For the assimilated notation, we will always assume the input space $\mathbf{X}$ to be numeric, that is

$$\mathbf{X} \subseteq \mathbb{R}^K, K \in \mathbb{N}. \tag{3.3.1}$$

Also, if optimal descriptions $\omega \in \mathbf{\Omega}$ are available, these are also represented numerically

$$\mathbf{\Omega} \subseteq \mathbb{R}^K, K \in \mathbb{N}.$$

We will sometimes substitute ω by a vector $\vec{w} = \omega \in \mathbf{\Omega} \subseteq \mathbb{R}^K$ when optimal descriptions are available.

Using only numeric attributes is less restrictive as it seems, because boolean or nominal attributes can always be translated into numeric vectors in which the values are limited to 0 and 1.

Representation of Individual Concepts On the basis of descriptions of examples one can start to describe individual concepts. Langley (1995) differentiates between extensional and intensional descriptions of concepts.

Definition 3.3.1 (Extensional definitions and descriptions of a learning system).

An *extensional definition* of some realizable concept $e_c(\circ|\mathbf{X})$ consists of an enumeration of all descriptions that are positive examples of this concept

$$\mathbf{X}^{[c]} := \{\vec{x} \in \mathbf{X} : e_c(\vec{x}) = 1\}, c \in \mathbf{C}.$$

In this context, $\vec{x} \in \mathbf{X}$ are called *extensional descriptions*. Elements of the extensional descriptions $\vec{x} \in \mathbf{X}^{[c]}$ are also called *instances* or *members* of concept number c, $c \in \mathbf{C}$.

We can only enumerate realizable concepts with finite potency $\left|\mathbf{X}^{[c]}\right| \in \mathbb{N}$ to formulate extensional definitions. Given some extensional description $\vec{x}$, we find out about its membership in concept number c by a simple match with the list $\mathbf{X}^{[c]}$. For some infinite list $\mathbf{X}^{[c]}$ this is clearly impossible in finite time. To overcome this problem, *intensional definitions* are designed to be more compact compared to such lists. In the following, we will develop a representation of intensional definitions to relate them to statistical models.

Intensional definitions describe realizable concepts e.g. by logical combinations of feature or attribute values. Certain (restricted) spaces of logical combinations are very common *concept description languages* in machine learning. For example, the intensional definition of some realizable concept $e_c(\circ|\mathbf{X})$ can consist of conjunctions of constraints on individual attributes. Mitchell (1997) denotes constraints on 6 nominal attributes for a certain concept by sixtupels of the form

$$<?, a_2, a_3, ?, ?, ?> .$$

To be an instance of this concept, the ? say, these attributes can have any value, the second and third attribute have to be equal to certain values, $x_2 \stackrel{!}{=} a_2$ and $x_3 \stackrel{!}{=} a_3$. This is only one of many ways to represent conjunctions of constraints.

Our choice for representing constraints on individual attributes is a set of boolean functions on individual features or attributes $x_k \in \mathbf{X}_k, k = 1, ..., K$, the elements of the description $\vec{x} \in \mathbf{X} = \times_{k=1}^{K} \mathbf{X}_k$. The individual constraints in the

example above read as follows:

$$\begin{aligned} e_{c,k}(x_k) &= 1,\ k = 1, 4, 5, 6, \\ e_{c,2}(x_2) &= \mathbb{I}_{a_2}(x_2), \\ e_{c,3}(x_3) &= \mathbb{I}_{a_3}(x_3), \end{aligned}$$

and the corresponding intensional description consists of the logical values $e_{c,k}, k = 1, ..., 6$. In general, a set of individual constraints is noted as follows:

$$\mathcal{I}^{[c]} = \{e_{c,k}(\circ) : \mathbf{X}_k \rightarrow \{0, 1\}, k = 1, ..., K\}. \tag{3.3.2}$$

Connecting Extensional and Intensional Description The information in $\mathcal{I}^{[c]}$ is not sufficient to define the corresponding concept $e_c(\circ|\mathbf{X})$ so far. One needs in addition some instructions how to combine the elements of the intensional description $\mathcal{I}^{[c]}$. Langley (1995) calls these instructions the *interpreter*. The interpreter allows to extend the intensional definition such that we can determine for each $\vec{x} \in \mathbf{X}$ whether it is a member or not a member of concept number c. Implicitly, the intensional definition $\mathcal{I}^{[c]}$ and the interpreter determine the partition of the input space into

$$\mathbf{X} = \mathbf{X}^{[c]} \cup \mathbf{X} \backslash \mathbf{X}^{[c]},\ c \in \mathbf{C} \tag{3.3.3}$$

In machine learning these partitions are typically called *decision regions*, and the joint faces between them *decision boundaries*.

A description like the one in (3.3.2) can be interpreted by the

1) the logical,
2) the threshold, or
3) the competitive approach.

In the *logical* approach the interpreter carries out an 'all or none' matching process. If and only if the extensional description $\vec{x}$ fulfills all conditions of the elements of the intensional definition $\mathcal{I}^{[c]}$, it is seen to be an instance of the corresponding concept number $c \in \mathbf{C}$:

$$e_c(\vec{x}) \quad = \quad \prod_{k=1}^{K} (e_{c,k}(x_k)).$$

Interpreted in this way, logical combinations result in a partition of the instance space with decision boundaries that are parallel to the axes.

In the *threshold* approach the interpreter carries out a partial matching process. Only some of the elements of the intensional definition must hold, or more generally, a weighted sum of the individual functions must lie above a certain threshold $T_c \in \mathbb{R}$ for a single $c \in \mathbf{C}$:

$$e_c(\vec{x}) \quad = \quad \mathbb{I}_{(T_c,\infty)} \left(\sum_{k=1}^{K} w_{c,k} e_{c,k}(x_k) \right) \tag{3.3.4}$$

Here, extensional definitions are unions of extensional definitions of the type that can be generated by the logical approach. This results in decision boundaries that are piecewise parallel to the axes.

So far, the interpreter matched an extensional description $\vec{x} \in \mathbf{X}$ with a single intensional definition $\mathcal{I}^{[c]}$. In this work, we consider a set of competing concepts $e_c(\circ|\mathbf{X}), c \in \mathbf{C}$. Here, we have to ensure, that for each extensional description $\vec{x}$ one and only one $e_c(\vec{x}), c \in \mathbf{C}$ is non-zero. We denote this restriction

by requiring the vector $\vec{e}_{\mathbf{C}}$ of the competing concepts $e_c, c{\in}\mathbf{C}$, to be a member of the unit simplex, $\vec{e}_{\mathbf{C}}{\in}\mathbf{U}^G$.

The appropriate interpreter follows the so-called *competitive* approach. For each $c{\in}\mathbf{C}$ and $\vec{x}{\in}\mathbf{X}$ the interpreter performs a partial matching, and computes the *competitive degree of match*

$$m(\circ,\circ) \quad : \quad \mathbf{C}\times\mathbf{X}\rightarrow\mathbb{R} \tag{3.3.5}$$

of the extensional description $\vec{x}$ with the intensional definitions $\mathcal{I}^{[c]},\ c{\in}\mathbf{C}$. It then selects the best competitor to be the best-matching concept. Thus, the set of competing intensional concept definitions $\mathcal{I}^{[c]}, c{\in}\mathbf{C}$ and an instruction for the evaluation of the competitive degree of match determines for each $\vec{x}{\in}\mathbf{X}$ the number of the concept $c{\in}\mathbf{C}$ it is matched with.

Different from the logical or the threshold approach, in the competitive approach decision regions are context-specific. That is with the same intensional definition $\mathcal{I}^{[c]}$, for example the set $\mathbf{X}^{[c]}$ corresponding to the binary partition with threshold approach $\mathbf{X}=\mathbf{X}^{[c]}\cup\mathbf{X}\backslash\mathbf{X}^{[c]},\ c{\in}\mathbf{C}$ may be different to the set $\mathbf{X}^{[c]}$ in the competitive approach:

$$\mathbf{X} \;=\; \bigcup_{c\in\mathbf{C}}\mathbf{X}^{[c]}, \tag{3.3.6}$$

$$\mathbf{X}^{[c]}\cap\mathbf{X}^{[\tilde{c}]}=\emptyset, c{\neq}\tilde{c}, c, \tilde{c}{\in}\mathbf{C}. \tag{3.3.7}$$

For example, let $\mathcal{I}^{[c]}$ be defined by a set of one-dimensional boolean functions as in (3.3.2). The competitive degree of match $m(c,\vec{x}|\texttt{one})$ can be measured by the same weighted sum of the elements as in (3.3.4) for the threshold approach. Only the competitive interpreter compares the individual match $m(c,\vec{x}|\texttt{one})$ not with a fixed threshold T_c, but with the matches $m(\tilde{c},\vec{x}|\texttt{one})$

of this description of the other concepts $\tilde{c} \neq c,\ c, \tilde{c} \in \mathbf{C}$. The description $\vec{x}$ is interpreted to be an instance of the concept with highest degree of match. The number of the best-matching concept is determined by:

$$\begin{aligned} \hat{c}(\vec{x}|\texttt{one}) &= \arg\max_{c \in \mathbf{C}} m(c, \vec{x}|\texttt{one}) \\ &= \arg\max_{c \in \mathbf{C}} \sum_{k=1}^{K} w_{c,k} e_{c,k}(x_k). \end{aligned}$$

The threshold approach and the competitive approach can of course also be used to interpret more complicated intensional definitions than the exemplifying individual boolean valued functions in (3.3.2). We want to derive a general framework to represent intensional concept definitions such that we can relate them to statistical models. We have to introduce some more notation for that purpose.

[**Notation**] In the machine learning literature, intensional definitions are often expressed by logic syntax. In the assimilated representation of statistical and machine learning approaches, we will represent them as function spaces with domain $\mathbf{X}$ and counterdomain $\mathbb{R}$, $\mathcal{I}^{[c]} \subset \mathcal{F}(\mathbf{X} \to \mathbb{R})$

Based on the numeric representation of the examples (3.3.1) such that $\mathbf{X} \subseteq \mathbb{R}^K$, the elements of intensional concept definitions relevant in this work can be represented by real-valued functions on subsets of the predictor variables:

$$f(\circ|\upsilon) : \mathbf{X}^{[\mathbf{M}]} \to \mathbb{R},\ \mathbf{M} \subseteq \{1, ..., K\}, \upsilon \in \Upsilon,$$

with $\mathbf{X}^{[\mathbf{M}]} := \bigtimes_{k \in \mathbf{M}} \mathbf{X}_k$.

For each function on some subset of variables exists an extension on the superset of all variables. Define $\vec{x}^{[\mathbf{M}]} \in \mathbf{X}^{[\mathbf{M}]}$ in dependence of $\vec{x} \in \mathbf{X}$ such that for all $k \in \mathbf{M}$ the elements are equal, $x_k^{[\mathbf{M}]} = x_k$. Then the extension $f(\circ|\upsilon, \mathbf{X})$ of $f(\circ|\upsilon, \mathbf{X}^{[\mathbf{M}]})$ is achieved by requiring:

$$f(\vec{x}|\upsilon, \mathbf{X}) \equiv f(\vec{x}^{[\mathbf{M}]}|\upsilon, \mathbf{X}^{[\mathbf{M}]}) \text{ for all } \vec{x} \in \mathbf{X}.$$

That way, we can define the functions of the intensional definitions to be members from some class $\mathcal{F}_\Upsilon(\mathbf{X}\to\mathbb{R})$ with some parameterization Υ which we always force to be unique. The parameter $\upsilon\in\Upsilon$ contains the definition of the actual domain $\mathbf{X}^{[\mathbf{M}]}$.

Definition 3.3.2 (Intensional concept definitions). We will represent intensional concept definitions by a set of functions $\mathcal{I}_{\Upsilon_c}$ from some class of functions $\mathcal{F}_\Upsilon(\mathbf{X}\to\mathbb{R})$ with unique parameters $\upsilon\in\Upsilon$:

$$\mathcal{I}_{\Upsilon_c} := \{f(\circ|\upsilon_c),\ \upsilon_c\in\Upsilon_c\}\subset\mathcal{F}_\Upsilon.$$

The set $\mathcal{I}_{\Upsilon_c}$ of functions may be finite or infinite, depending on the potency $|\Upsilon_c|$, $c\in\mathbf{C}$.

The instructions, how to combine the functions in $\mathcal{I}_{\Upsilon_c}$ to some real-valued function $m_c\in\mathcal{F}\,(\mathbf{X}\to\mathbb{R})$ that measures the individual degree of match of any description with one of the competing concepts $c\in\mathbf{C}$ are subsumed in an operator $\sqcup$:

$$\sqcup : \mathcal{I}_{\Upsilon_c}\to\mathcal{F}\,(\mathbf{X}\to\mathbb{R})\,.$$

The final competitive degree of match is given by the weighted individual match.

This determines for any $\vec{x}\in\mathbf{X}$ the matched concept number $c\in\mathbf{C}$ according to the competitive interpreter:

Definition 3.3.3 (Best-matching competing concepts).

The number of the best-matching competitive concept for any description $\vec{x}\in\mathbf{X}$ is determined by the following *matching rule*:

$$\hat{c}\left(\vec{x}\mid\sqcup,\Upsilon_{\mathbf{C}},\vec{w}\right) \;:=\; \arg\max_{c\in\mathbf{C}} w_c\left(\sqcup\mathcal{I}_{\Upsilon_c}\right)(\vec{x}), \tag{3.3.8}$$

with some operator $\sqcup : \mathcal{I}_{\Upsilon_c} \rightarrow \mathcal{F}\left(\mathbf{X}\rightarrow\mathbb{R}\right)$, $c\in\mathbf{C}$, $\Upsilon_{\mathbf{C}} := \{\Upsilon_1,...,\Upsilon_G\}$, and $\hat{\vec{\pi}} := (\hat{\pi}_1,...,\hat{\pi}_G)$.

The corresponding *best-matching competing concepts* $e_c\left(\circ|\sqcup,\Upsilon_{\mathbf{C}},\hat{\vec{\pi}}\right)$, $\mathbf{X}\rightarrow\{0,1\}$ with

$$e_c(\vec{x}|\sqcup,\Upsilon_{\mathbf{C}},\hat{\vec{\pi}}) \;:=\; \mathbb{I}_c\left(\hat{c}\left(\vec{x}\mid\sqcup,\Upsilon_{\mathbf{C}},\vec{w}\right)\right),\vec{x}\in\mathbf{X},c\in\mathbf{C},$$

are uniquely defined by two joint components relevant for all concepts, and two concept-specific components.

The joint components that define competing concepts are:

J1) the functional class $\mathcal{F}_{\Upsilon}\left(\mathbf{X}\rightarrow\mathbb{R}\right)$ for the intensional definitions $\mathcal{I}_{\Upsilon_c}\subset\mathcal{F}_{\Upsilon}$, and

J2) the operator $\sqcup : \mathcal{I}_{\Upsilon_c} \rightarrow \mathcal{F}\left(\mathbf{X}\rightarrow\mathbb{R}\right)$ for the instructions how to determine the individual match $m(c,\vec{x})$ of descriptions $\vec{x}\in\mathbf{X}$ with the intensional concept definition $\mathcal{I}_{\Upsilon_c}$, $c\in\mathbf{C}$.

These joint components determine the class of concepts that can be learnt by a learning system running with these joint components. This class is called the *hypotheses space* $\mathcal{H}$:

$$\mathcal{H} \;:=\; \left\{ h(\circ) \,\middle|\, \begin{array}{l} h(\vec{x}) := \mathbb{I}_c\left(\hat{c}\left(\vec{x}\mid\sqcup,\Upsilon_{\mathbf{C}},\vec{w}\right)\right), \\ \sqcup,\ \Upsilon_{\mathbf{C}}\subset\Upsilon^G,\ \hat{\vec{\pi}}\in\mathbb{R}^G,\ c\in\mathbf{C} \end{array} \right\} \tag{3.3.9}$$

The two concept specific components are typically the parameters the system has to learn from the examples:

S1) the parameters $\Upsilon_c \subset \Upsilon$ of the intensional definitions, and
S2) the weights $\hat{\pi}_c \in \mathbb{R}$,

for each $c \in \mathbf{C}$.

For ease of notation and whenever possible without loss of important information the representational choices will be subsumed in either the name of the method that is based on these choices (like e.g. `lda` , `qda` , and `cnb` in the preceding chapter), or with `met` as placeholder.

The intensional concept definitions $\mathcal{I}^{[c]}$, $c \in \mathbf{C}$, are the machine learning counterpart of the statistical model $\mathbf{\Lambda}$. And the statistical 'interpreters' are the evaluation instructions for deriving optimal class assignments according to some principle, e.g. conditional Bayesian, Minimax, or Bayes Risk introduced in Chapter 2.

In this section, we have structured the representation of best-matching competing concepts to be able to present methods from machine learning and methods from statistics in a comparable manner. In the following sections we will do so for two methods from machine learning, namely decision trees (Section 3.3.1) and support vector machines (Section 3.3.3, and the statistical methods introduced in the preceding Chapter (Section 3.3.2)

3.3.1 Example: Univariate Binary Decision Tree

The first example for our way to represent intensional definitions will introduce a standard method in the machine learning community : a decision tree. *Decision tree learning is one of the most widely used and practical methods for inductive inference* (Mitchell, 1997, pg. 52).

We will develop the intensional definition for a univariate binary decision tree (TREE) on the basis of K numeric attributes. A basic assumption for this definition is that for each attribute the order of its values has a meaning. In other words, values $x_k \in \mathbb{R}, k = 1, ..., K$, of attributes are measured on an ordinal or metric scale. A univariate binary decision tree matches descriptions $\vec{x} \in \mathbf{X} \subseteq \mathbb{R}^K$ with concepts by sorting them down the tree along nodes with numbers $q \in \mathbf{N} \subset \mathbb{N}$ from the *root node* $q := 1$ to some *leaf node*, denoted by $q^{[l]} \in \mathbf{N}^{[l]} \subseteq \mathbf{N}$. The leaf nodes provide the information for the final matching rule.

Each non-leaf node $q \in \mathbf{N} \backslash \mathbf{N}^{[l]}$ specifies a split. A split in a univariate binary tree depends on the value $x_{k(q)}$ the description $\vec{x} \in \mathbf{X}$ yields on a specific individual attribute $k(q) \in \{1, ..., K\}$. The description $\vec{x}$ is processed down the left branch, if the value is below or equal to a certain threshold value $t(q) \in \mathbf{X}_{k(q)}$, and down the right branch otherwise. Thus each description $\vec{x}$ goes along some path starting in the root node $q = 1$ and ending in the leaf node $q^{[l]}(\vec{x}) \in \mathbf{N}^{[l]}$. The path consists of a series of node numbers stored as an ordered subset of $\mathbf{N}$:

$$\mathbf{pt}(\vec{x}) \quad := \quad \left\{ 1, q_2^{[p]}(\vec{x}), q_3^{[p]}(\vec{x}), ..., q^{[l]}(\vec{x}) \right\} \subseteq \mathbf{N}. \tag{3.3.10}$$

We define univariate binary decision trees, such that all nodes $q \in \mathbf{N}$ carry information about the degree of match between each concept and any description $\vec{x}$ that is processed through this node, $q \in \mathbf{pt}(\vec{x})$. This information is provided by the weights $w(c, q)$. Relevant for the final decision are only the weights $w(c, q^{[l]})$ in leaf nodes $q^{[l]} \in \mathbf{N}^{[l]}$: they provide the competitive degree of match $m(c, \vec{x}) := w(c, q^{[l]}(\vec{x}))$ of the description $\vec{x}$ with concept numbers $c \in \mathbf{C}$. Often only weights in leaf nodes are given in a definition of a decision tree. We include weights in split nodes in our definition, because these play a role in learning decision trees.

Let $D \in \mathbb{N}$ denote the *depth* of a tree. The depth is the maximum number of nodes along any path from the root node to a leaf node, not counting the leaf node. Thus, D counts the binary splits along the longest path. Obviously, we can have a maximum of 2^D leaf nodes and a maximum of $2^{D+1} - 1$ nodes all together. Typically, not all potential nodes are elements of a decision tree, but we assume a certain "ghost numbering" of nodes as if all nodes were present. In Figure 3.1 you see a scheme of that numbering for a tree of depth $D = 3$.

As a result of that numbering, paths are given by

$$\begin{aligned} q_1^{[p]} &= 1 \\ q_2^{[p]} &\in \{2, 3\} \\ &\dots \\ q_{d+1}^{[p]} &\in \left\{2q_d^{[p]}, 2q_d^{[p]}+1\right\} \\ &\dots \\ q^{[l]} := q_{d_{\max}}^{[p]} &\in \left\{2q_{d_{\max}-1}^{[p]}, 2q_{d_{\max}-1}^{[p]} + 1\right\} \end{aligned}$$

With these preliminaries we can define univariate binary decision trees:

Definition 3.3.4 (Univariate Binary Decision Tree).

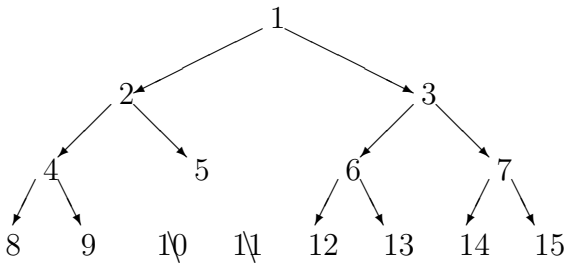

Figure 3.1: **Pattern for numbering the nodes in a decision tree.** Any potential node is numbered from top to bottom and from left to right. The actual nodes of a certain tree define the set $\mathbf{N}$. If some potential node is not a member of the tree — like 10 and 11 in the tree above — their numbers are not assigned. In the example $\mathbf{N} := \{1, 2, 3, 4, 5, 6, 7, 8, 9, 12, 13, 14, 15\}$.

Let $\mathbf{N}^{[a]}(D)$ be the set of the numbers of all potential nodes of trees with depth $D \in \mathbb{N}$, $\mathbf{N}^{[a]}(D) := \left\{1, 2, ..., 2^{D+1}-1\right\}$. We specify a univariate binary decision tree for extensional descriptions $\vec{x} \in \mathbf{X} \subseteq \mathbb{R}^K$ by:

1. $D \in \mathbb{N}$: the depth of the tree
2. $\mathbf{N} \subseteq \mathbf{N}^{[a]}(D)$: the set of numbers of the realized nodes in the tree,
3. $\vec{k} \in \{0, 1, ..., K\}^{2^{D+1}-1}$: the respective numbers of inspected attributes of a split, including the dummy "0" if no attribute is inspected:

$$k(q) \quad : \quad \begin{cases} k(q) = 0, & \text{if } q \in \mathbf{N}^{[a]}(D) \backslash \mathbf{N} \text{ or } q \in \mathbf{N}^{[l]} \\ k(q) \in \{1, ..., K\} & \text{else.} \end{cases}$$

4. $\vec{t} \in \mathbb{R}^{2^{D+1}}$: the respective threshold values, with

$$t(q) \quad : \quad \begin{cases} t(q) \in \mathbf{X}_0 := \{0\} & \text{if } k(q) = 0, \\ t(q) \in \mathbf{X}_{k(q)} & \text{else.} \end{cases}$$

5. $\mathbf{W} \in \mathbb{R}^{G \times \left(2^{D+1}-1\right)}$: a matrix of weights $\mathbf{W}[c, q] := w(c, q)$ for the competitive degree of match between concepts and descriptions passing through that node and $w(c, q) = 0$ for all ghost nodes $q \in \mathbf{N}^{[a]}(D) \backslash \mathbf{N}$ Columns of the matrix $\mathbf{W}$: $\vec{w}_q := (w(1, q), .., w(G, q))$ represent the weights of concepts per potential node in $\mathbf{N}^{[a]}(D)$. Rows provide

the weights of potential nodes $\vec{w}_c := \big(w(c,1),...,w(c,2^{D+1}-1)\big)$ per concept number $c \in \mathbf{C}$.

The functional class $\mathcal{F}_\Upsilon\,(\mathbf{X} \to \mathbb{R})$ for intensional definitions representable by univariate binary decision trees can now be expressed as:

$$\begin{aligned} J1 : \mathcal{F}_\Upsilon &= \mathcal{F}_{D,\mathbf{N},\vec{k},\vec{t},\vec{w}} \\ &= \left\{ \begin{array}{l} \mathbb{I}_{(-\infty,t(q)]}\left(\circ|\mathbf{X}_{k(q)}\right), \\ t(q) \in \mathbf{X}_{k(q)}, \\ \vec{k} \in \{0,1,...,K\}^{2^{D+1}-1}, \vec{w}_c \in \mathbb{R}^{2^{D+1}-1} \\ q \in \mathbf{N}^{[a]}(D), D \in \mathbb{N}. \end{array} \right\} \end{aligned}$$

Except for the weight vector $\vec{w}_c$, all parameters of the concept definitions are shared by all competing concepts. The shared parameters $D, \mathbf{N}, \vec{k}, \vec{t}$ determine the $\mathbf{pt}(\vec{x})$ of a description $\vec{x} \in \mathbf{X}$ from the root to its leaf node $q^{[l]}(\vec{x})$, and the weights $w(1, q^{[l]}(\vec{x}), ..., w(G, q^{[l]}(\vec{x}))$ determine the competitive degree of match.

The interpreter of these concept definitions is based on the *tree operator*:

$$J2 : \left\lfloor \overline{T} \right\rfloor \;:\; \mathcal{I}_{D,\mathbf{N},\vec{k},\vec{t},\vec{w}_c} \to \mathcal{F}(\mathbf{X} \to \mathbb{R})$$

that links each tree with the routine that determines for each $\vec{x} \in \mathbf{X}$ the concept's weight in the leaf node: $w(c, q^{[l]}(\vec{x}))$. We describe this operator $\left\lfloor \overline{T} \right\rfloor$ by the algorithm that determines for any (correctly specified) univariate binary decision tree and any corresponding $\vec{x} \in \mathbf{X}$ the weights of all concepts in the leaf node.

In Algorithm 3.3.1 we give the pseudo-(matlab)-code of this algorithm. The variable `tree` is a structure with fields for $D, \vec{k}, \vec{t}, \mathbf{W}$. The numbering according

to the scheme 3.1 allows to work along actual paths of the tree along nodes $q \in \mathbf{N}$ without considering $\mathbf{N}$ explicitly.

Algorithm 3.3.1

```
function [nleaf,w]=find_leafnode(x,tree)
q=1;
D=tree.D;
while q ≤ 2^(D+1)-1
     k=tree.k(q);
     t=tree.t(q);
     w=tree.W(:,q);
     if k ≠ 0
          if x(k) ≤ t
               q=2*q;
          else
               q=2*q+1;
          end
     else
          nleaf=q;
          q=2^(D+1);
     end
end
```

The concept specific parameter S1 in Definition 3.3.3 of intensional tree definitions are the weight vectors $\vec{w}_c \in \mathbb{R}^{2^{D+1}-1}$, $c \in \mathbf{C}$. Typically, no additional concept specific weight vectors (S2: $\hat{\pi}_c \in \mathbb{R}, c \in \mathbf{C}$) are used.

The final match of extensional descriptions $\vec{x} \in \mathbf{X}$ with the competing concepts with numbers $c \in \mathbf{C}$ is given by:

$$\begin{aligned} \hat{c}(\vec{x}|\texttt{tree}) &= \arg\max_{c \in \mathbf{C}} \left\{ \left\lfloor {}_T \right\rfloor \mathcal{I}_{D,\mathbf{N},\vec{k},\vec{t},\vec{w}}(\vec{x}) \right\} \\ &= \arg\max_{c \in \mathbf{C}} \left\{ w(c, q^{[l]}(\vec{x})) \right\} \end{aligned}$$

3.3.2 Example: Plug-in and Predictive Bayes Rules

In this section, we will embed the two variants of Bayes rules — plug-in and predictive — from statistics in the concept learning framework. In statistical terms, the score of the informative density $p(\vec{x}|\theta_c, \mathbf{\Lambda})$ for a certain $\vec{x}\in\mathbf{X}$ quantifies the probability to observe $\vec{x}(\omega)$ on some randomly generated object $\omega\in\mathbf{\Omega}$ that is a member of class c. In machine learning terms, $p(\vec{x}|\theta_c, \mathbf{\Lambda})$ quantifies the degree of match of some non-optimal description $\vec{x}$ with concept number c.

Plug-in Bayes rules For a plug-in Bayes rule the common component J1 from Definition 3.3.3 is the class of the densities of the informative distributions

$$J1: \quad \mathcal{F}_{\Upsilon} \quad := \mathcal{F}_{\Theta} = \quad \{p(\circ|\theta, \mathbf{\Lambda}), \theta\in\Theta\}.$$

For LDA and QDA that is the class of all densities of multivariate normal distributions in $\mathbb{R}^K$: (cp. with equations (2.3.9) and (2.3.11))

$$\begin{aligned}\mathcal{F}_{\Theta} &:= \mathcal{F}_{\mathbb{R}^K\times\mathbb{Q}^K_{[pd]}} \\ &= \left\{(2\pi)^{-\frac{K}{2}} \det(\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\vec{x}-\vec{\mu})\,\Sigma^{-1}(\vec{x}-\vec{\mu})'\right),\ \vec{\mu}\in\mathbb{R}^K,\ \Sigma\in\mathbb{Q}^K_{[pd]}\right\}\end{aligned}$$

Each concept is defined by the estimated density of its informative distribution:

$$\mathcal{I}_{\theta_{c|\mathbf{L}}} = \left\{p\left(\circ|\theta_{c|\mathbf{L}}, \mathbf{\Lambda}\right)\right\},\quad \theta_{c|\mathbf{L}}\in\Theta,\ c\in\mathbf{C}.$$

In LDA and QDA these estimated densities have different mean vectors, in LDA equal variances, in QDA unequal variances:

$$\begin{aligned}\mathcal{I}_{\vec{\mu}_{c|\mathbf{L}},\mathbf{S}_{\mathbf{L}}} &= \left\{p\left(\circ|\vec{\mu}_{c|\mathbf{L}}, \mathbf{S}_{\mathbf{L}}, \texttt{lda},\right)\right\}, \qquad \vec{\mu}_{c|\mathbf{L}}\in\mathbb{R}^K,\ \mathbf{S}_{\mathbf{L}}\in\mathbb{Q}^K_{[pd]},\ c\in\mathbf{C}, \\ \mathcal{I}_{\vec{\mu}_{c|\mathbf{L}},\mathbf{S}_{c|\mathbf{L}}} &= \left\{p\left(\circ|\vec{\mu}_{c|\mathbf{L}}, \mathbf{S}_{c|\mathbf{L}}, \texttt{qda},\right)\right\}, \quad \vec{\mu}_{c|\mathbf{L}}\in\mathbb{R}^K,\ \mathbf{S}_{c|\mathbf{L}}\in\mathbb{Q}^K_{[pd]},\ c\in\mathbf{C}.\end{aligned}$$

Represented in that way, only one function defines an individual concept, and we do not really need to define J2 the operator to combine "it". (For the sake of completeness we could say we use the identity mapping $\sqcup\mathcal{I}_{\Upsilon_c} = \mathcal{I}_{\Upsilon_c}$ as operator).

The concept specific parameters are the estimated parameters $\vec{\mu}_{c|\mathbf{L}}$ and — in case of QDA — $\mathbf{S}_{c|\mathbf{L}}$ of the multivariate normal distributions and the weights $\hat{\pi}_{c|\mathbf{L}} \in \mathbb{R}$, estimated by the relative class frequencies (see 2.3.2):

$$\begin{array}{lllll} S1(\texttt{lda}): & \vec{\mu}_{c|\mathbf{L}} & \in & \mathbb{R}^K, & c \in \mathbf{C}, \\ S1(\texttt{qda}): & \vec{\mu}_{c|\mathbf{L}} & \in & \mathbb{R}^K,\ \mathbf{S}_{c|\mathbf{L}} \in \mathbb{Q}^K_{[pd]}, & c \in \mathbf{C}, \\ S2 & \hat{\pi}_{c|\mathbf{L}} & = & \frac{f_{c|\mathbf{L}}}{N_{\mathbf{L}}}, & c \in \mathbf{C}. \end{array}$$

This results in the plug-in Bayes rules as given in (2.3.10) and (2.3.8).

The **predictive Bayes rule** of CNB is defined with the conjugate multinomial Dirichlet model for the class distribution $P_{C|\vec{\pi}}$ independently of the conjugate priors $P_{\theta_c|\psi} \equiv P_{\theta|\psi}$ for the informative distributions for $c \in \mathbf{C}$.

In general, for predictive Bayes rules the class of functions $\mathcal{F}_\Upsilon$ consists not of the class of informative densities belonging for $\mathcal{P}_{\vec{X}|\Theta}$, but of its product with the corresponding prior densitiy for $\mathcal{P}_{\theta|\Psi}$:

$$J1: \quad \mathcal{F}_\Upsilon = \quad \mathcal{F}_{\Theta,\Psi} = \quad \{p(\circ|\theta, \mathbf{\Lambda})\, p(\theta|\psi),\ \theta \in \Theta,\ \psi \in \Psi\}$$

The particular independence assumption in the naïve Bayes classifier results in intensional concept definitions based on functions for individual attributes, along with the corresponding prior. In the CNB classifier as introduced in Section 2.3.4 each of them is a univariate normal distribution $\mathcal{N}(\mu_k, \sigma_k^2)$ with standard minimal informative conjugate joint prior distribution for mean and variance (μ_k, σ_k^2), $k = 1, ..., K$.

The conjugate prior for each pair (μ_k, σ_k^2), $k = 1, ..., K$, is the product of the scaled inverse χ^2 distribution $\mathbf{inv}\chi^2(\nu_0, \sigma_0^2)$ for the variances $\sigma_{k|c}$ and the normal distribution $\mathcal{N}\left(\mu_0, \frac{\sigma^2}{\kappa_0}\right)$ for the conditional distributions $P_{\mu_{k|c}|\sigma^2_{k|c}}$.

$$\begin{aligned}\mathcal{F}_\Upsilon &= \mathcal{F}_{\vec{\nu}_0, \vec{\sigma}_0^2, \vec{\mu}_0, \vec{\kappa}_0} \\ &= \left\{ \begin{array}{l} p\left(\circ|\mu_k, \sigma_k^2\right) p\left(\mu_k, \sigma_k^2|\nu_{k|0}, \sigma^2_{k|0}, \mu_{k|0}, \kappa_{k|0}\right) \\ \mu_k \in \mathbb{R},\ \sigma_k^2 \in \mathbb{R}^+,\ \nu_{k|0} \in \mathbb{N}^0, \mu_{k|0} \in \mathbb{R}\ \kappa_{k|0}, \sigma^2_{k|0} \in \mathbb{R}^+, \\ k = 1, ..., K \end{array} \right\}\end{aligned}$$

The intensional concept definitions for each concept number $c \in \mathbf{C}$ is given by the (same!) class of informative distributions together with the class-specific posterior density with updated parameters $\nu_{k,c|\mathbf{L}}, \sigma^2_{k,c|\mathbf{L}}, \mu_{k,c|\mathbf{L}}, \kappa_{k,c|\mathbf{L}}$ (cp. (2.3.12)):

$$\begin{aligned}&\mathcal{I}_{\vec{\nu}_{c|\mathbf{L}}, \vec{\sigma^2}_{c|\mathbf{L}}, \vec{\mu}_{c|\mathbf{L}}, \vec{\kappa}_{c|\mathbf{L}}} \\ &\quad = \quad \left\{p\left(\circ|\mu_{k,c}, \sigma^2_{k,c}\right) p\left(\mu_{k,c}, \sigma^2_{k,c}|\nu_{k,c|\mathbf{L}}, \sigma^2_{k,c|\mathbf{L}}, \mu_{k,c|\mathbf{L}}, \kappa_{k,c|\mathbf{L}}, k = 1, ..., K\right)\right\} \\ &\quad \text{with} \quad \nu_{k,c|0} \in \mathbb{N}^0, \mu_{k,c|0} \in \mathbb{R},\ \kappa_{k,c|0}, \sigma^2_{k,c|0} \in \mathbb{R}^+,\ k = 1, ..., K\end{aligned}$$

The concept specific component S1 of these intensional definitions are not the uncertain parameters $\left(\vec{\mu}_c, \vec{\sigma^2}_c\right)$, but the parameters of the conjugate distribution that describes the posterior uncertainty about $\left(\vec{\mu}_c, \vec{\sigma^2}_c\right)$, $c \in \mathbf{C}$.

$$S1: \quad \nu_{k,c|\mathbf{L}} \in \mathbb{N}^0, \mu_{k,c|\mathbf{L}} \in \mathbb{R},\ \kappa_{k,c|\mathbf{L}}, \sigma^2_{k,c|\mathbf{L}} \in \mathbb{R}^+,\ k = 1, ..., K,\ c \in \mathbf{C}$$

With the multinomial-Dirichlet model for the class distribution $P_{C|\vec{\pi}}$ the concept specific weights are determined by the posterior predictive probabilities (see 2.3.16):

$$S2: \quad \hat{\pi}_{c|\mathbf{L}} = \frac{\alpha_c + f_{c|\mathbf{L}}}{\alpha_0 + N_{\mathbf{L}}}, \quad c \in \mathbf{C}, \quad \alpha_0 := \sum_{c=1}^G \alpha_c.$$

To remind, $f_{c|\mathbf{L}}, c \in \mathbf{C}$ is the observed frequency of examples in the classes.

The operator for predictive Bayes rules is the integral with respect to θ that evaluates the mean posterior degree of match:

$$\begin{array}{rrcl} J2: & \left(\sqcup\mathcal{I}^{[c]}\right)(\vec{x}) & := & \int_{\Theta} p\left(\vec{x}|\theta,\mathbf{\Lambda}\right) p(\theta|\psi_{c|\mathbf{L}},\mathbf{\Lambda}) d\mu(\theta) \\ & & = & p\left(\vec{x}|\psi_{c,\mathbf{L}},\mathbf{\Lambda}\right). \end{array}$$

In case of the modelling of CNB with minimal informative priors this results in (see Section 2.3.4):

$$\left(\sqcup\mathcal{I}^{[c]}\right)(\vec{x}) \;=\; \prod_{k=1}^{K} p\left(x_k \middle| \mu_{k|c,\mathbf{L}}, \sqrt{1+\tfrac{1}{f_{c|\mathbf{L}}}}\mathbf{S}_{k|c,\mathbf{L}}, f_{c|\mathbf{L}}-1\right).$$

with the densities from a Student-t-distribution with $f_{c|\mathbf{L}}-1$ degrees of freedom, location $\mu_{k|c,\mathbf{L}}$, and scale $\sqrt{1+\frac{1}{f_{c|\mathbf{L}}}}\mathbf{S}_{k|c,\mathbf{L}}$, $k = 1, ..., K, c{\in}\mathbf{C}$. To remind, $\mathbf{S}_{k|c,\mathbf{L}}$, $k = 1, ..., K$ is the sample variance of variable $X_{k|c}$ in the sub-sample of all examples in $\mathbf{L}$ in class c, $c{\in}\mathbf{C}$.

The competitive degree of match is measured by the product of class weight and the mean posterior degree of match and results in the predictive Bayes rules as given in (2.3.22):

$$\hat{c}(\vec{x}|\mathbf{L},\texttt{cnb}) \;=\; \arg\max_{c\in\mathbf{C}} \left\{ \frac{\alpha_c+f_{c|\mathbf{L}}}{\alpha_0+N_{\mathbf{L}}} \prod_{k=1}^{K} p\left(x_k \middle| \mu_{k|c,\mathbf{L}}, \sqrt{1+\tfrac{1}{f_{c|\mathbf{L}}}}\mathbf{S}_{k|c,\mathbf{L}}, f_{c|\mathbf{L}}-1\right) \right\}$$

3.3.3 Example: Support Vector Machines

Support vector machines (SVM, Vapnik, 1995) separate descriptions belonging to two competing concepts $c{\in}\mathbf{C} = \{1, 2\}$ by hyperplanes with a large margin between the descriptions of the competing concepts. Support vector machines are very popular in the machine learning community, which would certainly not be the case if they were only about linear separation with hyperplanes.

With the "kernel-trick", though, one can map descriptions into a possibly infinite dimensional Euclidean space and then do a linear separation in that space. Actually, in current research support vector machines are approached as one instantiation of the larger class of *kernel machines*, that include the fields of gaussian process prediction, mathematical programming with kernels, regularization networks, reproducing kernel Hilbert spaces, and related methods (Schölkopf and Smola, 2000-2002).

The following derivation of the components J1, J2, S1, and S2 of the best-matching competing concepts for support vector machines relies on the presentations in Schölkopf et al. (1995) and Burges (1998), though notation is different.

Consider a hyperplane defined on the basis of the dot product $(\circ \cdot \circ) : \mathbf{F} \times \mathbf{F} \rightarrow \mathbb{R}$ in some (possibly infinite dimensional) Euclidian space $\mathbf{F}$:

$$\mathbf{hp}(f, \theta) \quad = \quad , \{g \in \mathbf{F} : \ f \cdot g + \theta = 0\} \tag{3.3.11}$$

for some $f \in \mathbf{F}$ and $\theta \in \mathbb{R}$. The pair of parameters (f, θ) defining the hyperplane so far are not unique. Given a finite set of *reference points* $\mathbf{G}_N := \{g_1, ..., g_N\} \subset \mathbf{F}$, though, we can use these to pose a restriction that makes the parameters unique:

$$\min_{n=1}^{N} |f \cdot g_n + \theta| \quad \stackrel{!}{=} \quad 1. \tag{3.3.12}$$

This leads to the canonical form of the hyperplane with respect to the reference points $\mathbf{G}_N$:

$$\mathbf{hp}(\mathbf{G}_N, f, \theta) \quad = \quad \left\{ g \in \mathbf{F} : \ f \cdot g + \theta = 0, \ \min_{n=1}^{N} |f \cdot g_n + \theta| = 1 \right\}. \tag{3.3.13}$$

This canonical form induces two other parallel hyperplanes with equations

$$\mathbf{hp}^{+}(\mathbf{G}_N, f, \theta) \quad : \quad f \cdot g + \theta = +1 \tag{3.3.14}$$

$$\mathbf{hp}^{-}(\mathbf{G}_N, f, \theta) \quad : \quad f \cdot g + \theta = -1 \tag{3.3.15}$$

respectively. Per definition, no reference point lies between these hyperplanes. This subspace plays an important role in the derivation of performance criteria for the selection of best separating hyperplanes as you will see in Section 3.4.2. Note here, that the distance between these hyperplanes – the so-called *margin* – results in (see e.g. Burges, 1998):

$$\mathbf{mr}(\mathbf{G}_N, f, \theta) \quad = \quad \frac{2}{\|f\|}. \tag{3.3.16}$$

The space $\mathbf{F}$ can be the input space $\mathbf{X} \in \mathbb{R}^K$ itself, or the counterdomain of some mapping $\Phi(\circ) : \mathbf{X} \rightarrow \mathbf{F}$.

Neither presentation of results nor calculation typically takes place in this Euclidean space, only the linear separation of the mapped descriptions. We do not need to perform calculations in this space, because one considers mappings $\Phi(\circ)$ to which there exist some kernel function, which can be used to calculate the dot product in that space:

$$\mathbf{K}(\vec{x}, \vec{y}) \quad = \quad \Phi(\vec{x}) \cdot \Phi(\vec{y}), \text{ for all } \vec{x}, \vec{y} \in \mathbf{X}$$

Kernel functions that represent the dot product in some Euclidean space can be found by checking the so-called *Mercer's condition* (Burges, 1998, e.g.).

Standard kernels are e.g.polynomial kernels of fixed degree $p \in \mathbb{N}$:

$$\mathbf{K}\left(\vec{x}, \vec{y} | p\right) \quad = \quad \left(\sum_{k=1}^{K} (x_k y_k) + 1 \right)^{p},$$

or Gaussian radial basis functions with width parameter σ^2:

$$\mathbf{K}\left(\vec{x},\vec{y}|\sigma^2\right) \;=\; \exp\left(\frac{-\sum_{k=1}^{K}(x_k-y_k)^2}{2\sigma^2}\right). \tag{3.3.17}$$

Throughout this work, we use the support vector machine with Gaussian RBF kernels $\mathbf{K}\left(\circ,\circ|\sigma^2\right):,\ \mathbf{X}\times\mathbf{X}\rightarrow\mathbb{R}_0^+$.

Each hyperplane corresponds to a specific concept $e_c(\circ \mid \mathbf{X}), c{\in}\mathbf{C}$ in the following way: first find the smallest ball in the feature space that contains all feature reference points of $\mathbf{X}_N$. Let

$$\mathbf{B}_{\mu,R} := \left\{\vec{x}{\in}\mathbf{X} : \|\Phi(\vec{x})-\mu\| < R\right\}, \mu{\in}\mathbf{F}, R{\in}\mathbb{R}$$

be any ball in the feature space with center μ and radius R. Then the smallest ball containing all feature reference points of $\mathbf{X}_N$ is given by:

$$\mathbf{B}(\mathbf{X}_N) \;:=\; \arg\min\left\{\mathbf{B}_{\mu,R} : \Phi(\vec{x}_n){\in}\mathbf{B}_{\mu,R},\ \text{for all } \vec{x}_n{\in}\mathbf{X}_N, \mu{\in}\mathbf{F}, R{\in}\mathbb{R}\right\}$$

We call $\mathbf{B}(\mathbf{X}_N)$ the *feature reference ball*.

The domain of the decision function of support vector machine classifiers is restricted to all descriptions within the feature reference ball. Descriptions within the feature reference ball and above the hyperplane are defined to be members of the concept, those below are non-members. That is, we define

$$e_c\left(\vec{x} \mid \sigma^2, \mathbf{X}_N, \vec{w}, \theta)\right) \;:=$$
$$\begin{cases} \mathbb{I}_{\mathbb{R}^+}\left(K(\vec{w},\vec{x})+\theta \mid \sigma^2\right) & \text{if } \vec{x}{\in}\mathbf{B}(\mathbf{X}_N), \\ \text{undefined} & \text{otherwise.} \end{cases} \tag{3.3.18}$$

This is the standard threshold approach (3.3.4) with threshold $T=0$ to the interpretation of intensional definitions.

The class of all such canonical forms of hyperplanes within the corresponding feature reference ball is the joint component J1 of the support vector machine classifier:

$$J1: \ \mathcal{F}_{\Upsilon} \ = \ \mathcal{F}_{\sigma^2,\mathbf{X}_N,\vec{w},\theta} \tag{3.3.19}$$

$$= \left\{ \begin{array}{l} \left\{ \begin{array}{l} \vec{x}\in\mathbf{B}(\mathbf{X}_N) : K(\vec{w},\vec{x}|\sigma^2)+\theta=0, \\ \min_{n=1}^{N}\{|K(\vec{w},\vec{x}_n|\sigma^2)+\theta|\}=1, \end{array} \right\} \\ \sigma^2\in\mathbb{R}^2, \ \mathbf{X}_N\subseteq\mathbf{X}, \ \vec{w}\in\mathbb{R}^K, \ \theta\in\mathbb{R} \end{array} \right\} \tag{3.3.20}$$

To make individual concepts comparable, they should be defined on the same domain. Therefore, the reference points should be shared. We also want them to be based on the same kernel function with equal widths, that is σ^2 is also modelled by a joint parameter.

For a corresponding individual concept definition $\mathcal{I}^{[c]}=\mathcal{I}_{\sigma^2,\mathbf{X}_N,\vec{w}_c,\theta_c}$ we have to fix the concept specific parameters S1:

$$S1: \quad \vec{w}_c\in\mathbb{R}^K, \quad \theta_c\in\mathbb{R}.$$

To develop a competitive degree of match we interpret the value that is compared to the threshold $T=0$ in (3.3.18) as measurement for the individual degree of match. Thus the operator $\sqcup : \mathcal{I}_{\sigma^2,\mathbf{X}_N,\vec{w}_c,\theta_c} \rightarrow \mathcal{F}\left(\mathbf{X}\rightarrow\mathbb{R}\right)$ is defined by

$$J2 \quad : \quad \left(\sqcup\mathcal{I}_{\sigma^2,\mathbf{X}_N,\vec{w}_c,\theta_c}\right)(\vec{x})$$

$$= \begin{cases} K(\vec{w}_c,\vec{x}|\sigma^2)+\theta_c & \text{if } \vec{x}\in\mathbf{B}(\mathbf{X}_N), \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Note that the absolute individual degree of match of valid descriptions according to J2 divided by $\|\Phi(\vec{w}_c)\|=\sqrt{K(\vec{w}_c,\vec{w}_c)}$ measures the perpendicular distance to the concept's hyperplane:

$$d(\vec{x},\mathcal{I}^{[c]}) \ = \ \frac{|K(\vec{w}_c,\vec{x}|\sigma^2)+\theta_c|}{\sqrt{K(\vec{w}_c,\vec{w}_c)}}.$$

There exist several suggestions how to combine binary SVM to more competing concepts (see e.g. Platt et al., 2000; Vogtländer and Weihs, 2000). We will use the so-called *one-against-rest-strategy*, where each concept is defined as binary competing concept against the comprised concept "all others", namely the "rest".

We take the view that the distances to the different hyperplanes can be compared in size and represent the competitive degree of match:

$$\hat{c}(\vec{x}|\texttt{svm}) \quad = \quad \begin{cases} \arg\max_{c\in\mathbf{C}} \left\{ \frac{K(\vec{w}_c,\vec{x}|\sigma^2)+\theta_c}{\sqrt{K(\vec{w}_c,\vec{w}_c)}} \right\} & \text{if } \vec{x}\in\mathbf{B}(\mathbf{X}_N), \\ \text{undefined} & \text{otherwise.} \end{cases} \tag{3.3.21}$$

This makes $(K(\vec{w}_c,\vec{w}_c))^{-\frac{1}{2}}$ to be the concept weight factor S2, $c\in\mathbf{C}$.

This can be justified, because the hyperplanes as we defined them lie in the same space, namely the feature ball. One may think of other combinations that take into account that smaller values of $K(\vec{w}_c,\vec{w}_c)$ can be interpreted to signal a higher "quality" of the hyperplane as separator as you will see later. One can think of other weightings with this factor.

Bottom Line about Representation

We saw that for competing concepts in machine learning typically a competitive interpreter is used to assess the competitive degree of match of descriptions with the concepts' individual intensional concept definitions $\mathcal{I}_{\Upsilon_c}$. We derived a representation of best-matching competing concepts based on two common components namely the class of functions from which the intensional definitions can be taken (J1), the instructions to assess individual degree of match

(J2) in general, and the concept specific parameters of the intensional definitions (S1) and an additional weighting factor for each concept (S2). The final matching rule is an argmax rule.

To some extent, the set of functions defined to be superset of the functions in $\mathcal{I}_{\Upsilon_c}$ is arbitrary: one could always use $\sqcup\mathcal{I}_{\Upsilon_c}$. Which functions are defined to be the basic functions is important for two reasons:

1) it can be useful for the interpretation of concepts e.g. by revealing the form of the decision boundaries,
2) it can organize the learning of concepts as we will see in the next section.

3.4 Performance and Learning

The task a learning system has to perform when learning competing concepts is to infer for each concept the corresponding "boolean-valued function from examples of its input and output" (Definition 3.1.2) — given that any input is an instance of exactly one of the concepts.

According to our general representation for best-matching competing concepts in Definition 3.3.3 the first step of learning is done by the developer of the learning system by specifying the joint components $J1 : \mathcal{F}_\Upsilon$ and $J2 : \sqcup$. The next step in learning consists in determining the concept specific parameters $S1 : \Upsilon_c \subset \Upsilon$ and $S2 : w_c \in \mathbb{R}$ for $c \in \mathbf{C}$ on the basis of the examples in $\mathbf{L}$.

If one wants to learn 'optimal' parameters one has to define the performance measure to rank different choices for the parameters according to their performance.

In this respect, given any specification of these parameters, the performance on matching some input $\vec{x}^* \in \mathbf{X}$ with known number of the "true" concept c^* is binary: either the learning systems output according to the learnt matching rule is correct or not:

$$\hat{c}\left(\vec{x}^* \mid \mathbf{L}, \texttt{met}\right) \stackrel{?}{=} c^*.$$

In general, one can define different losses for mistakes in dependence of the concepts that were mismatched, but in this work we will only consider the zero-one loss. This is the common choice for the individual loss in classification problems and is also the commonly chosen basic individual performance measure in concept learning. Given the example $(c^*, \vec{x}^*)$:

$$L^{[01]}(c^*, \hat{c}) := \begin{cases} 0 & \text{if } c^* = \hat{c}\left(\vec{x} \mid \mathbf{L}, \texttt{met}\right), \\ 1 & \text{otherwise.} \end{cases}$$

As stated earlier, the divisive question is how to extent this concrete, individual loss to some "expected" loss. As in the frequentist approach, in concept learning the expected loss is defined with respect to the matching of any hypothetical future description $\vec{x} \in \mathbf{X}$, and not conditional on one specific description at hand as in the conditional Bayesian principle.

The expected loss in concept learning is normally called the *expected error rate* or the *error probability*, respectively, as these two coincide for zero-one loss as is shown e.g. in (2.2.7) and below in (3.4.2). This is the starting point to make assumptions about the generating process of descriptions and their relation to concepts, because these assumptions establish the basis to "expect" something. The first assumption is about the relation of concepts and descriptions: whether optimal descriptions are available or not. In machine learning learning systems are typically first analyzed on the basis of ideal

concepts (Section 3.4.1), and then the derived statements are extended to realizable concepts (Section 3.4.3).

3.4.1 Ideal Concepts

Let us assume ideal concepts $e_c(\circ|\mathbf{\Omega})$, $c\in\mathbf{C}$, based on available optimal descriptions $\omega\in\mathbf{\Omega}\subseteq\mathbb{R}^K$. The performance of a learnt best-matching rule $\hat{c}\,(\circ \mid \mathbf{L}, \texttt{met}) := \hat{c}\left(\circ \mid \sqcup, \Upsilon_{\mathbf{C}|\mathbf{L}}, \vec{w}_{\mathbf{L}}\right)$ can be understood in terms of its *mean loss on the input space*:

$$\overline{L^{[01]}_{\mathbf{\Omega}}}\,(\hat{c}) \;:=\; \frac{1}{\mu(\mathbf{\Omega})}\int_{\mathbf{\Omega}} L^{[01]}\left(c(\omega), \hat{c}\,(\omega \mid \mathbf{L}, \texttt{met})\right) d\mu(\omega) \qquad (3.4.1)$$

because $\frac{1}{\mu(\mathbf{\Omega})}$ is finite assuming μ to be either the counting measure or the Lebesgue measure (cp. (2.1.1)).

According to this performance criterion, all potential descriptions are equally treated, as the loss quantifying the error $c(\omega) \neq \hat{c}^{[m]}(\omega|\mathbf{L})$ in matching some description ω gets the same weight $\frac{1}{\mu(\mathbf{\Omega})}$ for all $\omega\in\mathbf{\Omega}$. This is an appropriate choice to define some "expected error rate", if

1. the generation of future descriptions is completely under control and we know they will be generated by a complete enumeration of the input space without repetition, or
2. the random process that governs the generation of future descriptions follows some uniform distribution on the input space such that each description has the same probability to be generated.

Extending these assumptions, we can model future descriptions $\omega\in\mathbf{\Omega}$ to be sampled from some population of descriptions according to some general probability model $(\mathbf{\Omega}, \mathcal{A}, P)$. Here, the distribution P can be any distribution on $(\mathbf{\Omega}, \mathcal{A})$, such that there may be some descriptions $\omega\in\mathbf{\Omega}$ that have a

higher probability to occur than others. Errors on these descriptions should be penalized more than errors on descriptions that have a low probability to be generated.

We denote with $\vec{W}(\omega) := \omega$, $\omega \in \mathbb{R}^K$, the random variable on $(\mathbf{\Omega}, \mathcal{A}, P)$ to be able to distinguish the random variable $\vec{W}$ from its realization $\omega \in \mathbf{\Omega}$ in the notation.

The expected error rate of a learnt matching rule $\hat{c}\,(\circ \mid \mathbf{L}, \texttt{met})$ with respect to the distribution $P_{\vec{W}}$ and zero-one loss results in the probability that an error occurs when the learnt matching rule is used in future:

$$\begin{aligned}
\mathbf{EP}(\hat{c}, P_{\vec{W}}) &:= \mathbf{E}_{\vec{W}}\left(L^{[01]}\left(c(\vec{W}), \hat{c}\left(\vec{W} \mid \mathbf{L}, \texttt{met}\right)\right)\right) \\
&= \int_{\mathbf{\Omega}} L^{[01]}\left(c(\omega), \hat{c}\,(\omega \mid \mathbf{L}, \texttt{met})\right) p(\omega) d\mu(\omega) \\
&= \int_{\mathbf{\Omega}} \left(1 - \mathbb{I}_{c(\omega)}(\hat{c}\,(\omega \mid \mathbf{L}, \texttt{met})\right) p(\omega) d\mu(\omega) \\
&= P_{\vec{W}}\left(\{\omega \in \mathbf{\Omega} :\ c(\omega) \neq \hat{c}\,(\omega \mid \mathbf{L}, \texttt{met})\}\right). \qquad (3.4.2)
\end{aligned}$$

Therefore, in the following we will use the terms expected error rate and *error probability* interchangeably, referring to $\mathbf{EP}(\hat{c}, P_{\vec{W}})$ of a rule $\hat{c}\,(\circ \mid \mathbf{L}, \texttt{met}) : \mathbf{\Omega} \rightarrow \mathbf{C}$ when sampling objects $\omega \in \mathbf{\Omega}$ from $P_{\vec{W}}$.

Note also, that the expected loss corresponds to the frequentist expected loss, the so-called risk function in Definition 2.2.3:

$$\mathbf{E}_{\vec{W}}\left(L^{[01]}(c(\vec{W}), \hat{c}^{[m]}(\vec{W}|\mathbf{L})\right) = R(c, \hat{c}^{[m]}).$$

To minimize the risk function, in a standard frequentist approach, one would make assumptions about the distribution $P_{\vec{W}}$, e.g. to be a member of a specific parametric class of distributions with parameters $\theta \in \mathbb{R}^M$, and learn these parameters according to one of the frequentist principles (Minimax, Invariance,

Bayes,...). In machine learning, though, typically one wants to learn rules without such a restriction on the distribution $P_{\vec{W}}$.

Yet, as long as we do not know neither $P_{\vec{W}}$ nor the true ideal competing concepts $\vec{e}_{\mathbf{C}}(\circ|\mathbf{\Omega}), c{\in}\mathbf{C}$, we can not evaluate the performance of any learning system with respect to the mean loss on the input space nor with respect to the error probability. We have to find ways to relate these theoretical *performance criteria* to something we can observe: *performance measures*. If optimal descriptions are available, in the experience as well as in future, the inductive learning hypothesis is a useful guide. Informally it says (Mitchell, 1997):

Principle 3.4.1 (Inductive Learning Hypothesis).

Any concept found to approximate the ideal concept well over a sufficiently large set of training examples will also approximate the ideal concept well over other unobserved examples.

Following this hypothesis, the performance measure for best-matching competitive concepts is the *empirical risk* respectively the *training error rate*:

Definition 3.4.1 (Training Error Rate). The empirical risk of a set of best-matching competitive concepts is the expected loss with respect to the empirical distribution $P_{\vec{W}|\mathbf{L}}$ according to the examples $\mathbf{L}$. For zero-one loss this results in the *training error rate*:

$$\begin{aligned}\mathbf{EP}(\hat{c}, P_{\vec{W}\mathbf{L}}) &:= \mathbf{E}_{\vec{W}|\mathbf{L}}\left(L^{[01]}(c(\vec{W}), \hat{c}(\vec{W}|\mathbf{L}, \mathtt{met})\right) \\ &= \frac{1}{N_{\mathbf{L}}}\sum_{n=1}^{N_{\mathbf{L}}} L^{[01]}\left(c_n, \hat{c}(\omega_n|\mathbf{L}, \mathtt{met}\right).\end{aligned}$$

Minimizing the training error rate $\mathbf{EP}(\hat{c}, P_{C,\vec{X}|\mathbf{L}}$ can be shown to be sound given certain assumptions are fulfilled:

1. The descriptions $\omega_{\mathbf{L}} := \{\omega_1, ..., \omega_{N_{\mathbf{L}}}\}$ in the training set can be interpreted to be realizations of some random sample from some population with probability space $(\mathbf{\Omega}, \mathcal{A}, P_{\vec{W}})$. (**i.i.d. assumption**)
2. The concept numbers $\mathbf{c_L} := \{c_1, ..., c_{N_{\mathbf{L}}}\}$ of the training examples are assigned according to the true competing concepts $\vec{e}_{\mathbf{C}} \in \mathbf{U}$ (**supervised learning**).
3. Future descriptions will be additional independent random draws from the same population. Their unknown "true" concept numbers will correspond to the same "true" competing concepts $\vec{e}_{\mathbf{C}} \in \mathbf{U}^G$ (**structural stability**).
4. Either $\mathbf{\Omega}$ is a countable set, or the true concepts only have countable discontinuities on $\mathbf{\Omega}$.

Assume, we do know nothing about the probability $P_{\vec{W}}$ and optimal descriptions are available. Let us split the error probability **EP** in two parts:

$$\begin{aligned} & P_{\vec{W}}\left(\left\{\omega \in \mathbf{\Omega} :\ c(\omega) \neq \hat{c}^{[m]}(\omega|\mathbf{L})\right\}\right) \\ &= P_{\vec{W}}\left(\left\{\omega \in \mathbf{L} :\ c(\omega) \neq \hat{c}^{[m]}(\omega|\mathbf{L})\right\}\right) + P_{\vec{W}}\left(\left\{\omega \in \mathbf{\Omega} \backslash \mathbf{L} :\ c(\omega) \neq \hat{c}^{[m]}(\omega|\mathbf{L})\right\}\right) \end{aligned}$$

For countable spaces $\mathbf{\Omega}$ and corresponding discrete distributions $P_{\vec{W}}$ and with more and more examples sampled from the distribution $P_{\vec{W}}$ (that is $N_{\mathbf{L}} \to \infty$) the second addend goes to zero, and obviously one should minimize the first addend. But even for small training sets, preferring competing concepts $\vec{e}_{\mathbf{C}|\mathbf{L}}$ that make more errors on the training set than others can not be justified without additional assumptions on $P_{\vec{W}}$ or the true concepts $\vec{e}_{\mathbf{C}}$. If we preferred some matching rule with more errors over one with fewer errors, the

first addend would obviously increase. And there is no information about the behavior of the second addend to justify some believe that the second addend becomes smaller with such a choice. We can follow the same reasoning for continuous distributions $P_{\vec{W}}$ on uncountable spaces $\mathbf{\Omega}$ assuming the corresponding concepts only have countable discontinuities.

As we will see, this is different if only non-optimal descriptions are available. For non-optimal descriptions minimizing the training error rate is not a good strategy to achieve low error probability.

In broad hypotheses spaces typically there will be more than one hypothesis that gives no training error. Within the subspace of hypotheses that make no training error, so far we have no mean to distinguish its members with respect to their error probability. And also, to compare learning systems, many learning systems are able to output some hypothesis or a set of hypotheses that make no training errors. Thus, minimum training error is no decisive performance measure, neither to find the best hypothesis within some hypotheses space, nor to compare the performance of different learning systems. We have to combine the training error with other performance criteria and their measures to be able to rank further given we want to do so.

The so-called *probably approximately correct* (PAC) learning framework offers a way to assess the quality of different learning systems in terms of the computational resources required for high quality learning, which relates to the number of training examples required for high quality learning. In addition, results based on the PAC learning model provide useful insights regarding the relative complexity of different learning problems in terms of the true concepts

to learn and regarding the rate at which generalization accuracy improves with additional training examples (Mitchell, 1997).

The guarantee of the quality these learning systems have to issue is that a learnt hypothesis for two competing concepts will probably be approximately correct for any true concept from a certain class of concepts. In other words, the error probability of the learnt hypothesis is bound to be below a certain level, and the learning system has to learn such a hypothesis with a probability above a certain level.

Definition 3.4.2 (PAC learnability).

Consider a class of ideal concepts $\mathcal{C}\left(\mathbf{\Omega} \rightarrow \{0,1\}\right) \subseteq \mathcal{F}\left(\mathbf{\Omega} \rightarrow \{0,1\}\right)$ with $\mathbf{\Omega} \subseteq \mathbb{R}^K$, and some learner with hypotheses space $\mathcal{H}\left(\mathbf{\Omega} \rightarrow \{0,1\}\right)$. Then $\mathcal{C}$ is PAC learnable by the learner using $\mathcal{H}$, if for

— all concepts $e_c \in \mathcal{C}$,
— any distribution $P_{\vec{W}}$ over $\mathbf{\Omega}$,
— any $0 < \varepsilon < \frac{1}{2}$
— any $0 < \delta < \frac{1}{2}$

the learning system finds with probability $\mathbf{P} > (1-\delta)$ some hypothesis $h \in \mathcal{H}$ with error probability $\mathbf{EP}(h, P_{\vec{W}}) < \varepsilon$.

As it is common, this definition is in terms of the learning of only two competing concepts. We have not found an extension to more competing concepts under the assumption of ideal concepts, though we would expect that it exists. Intuitively, it seems possible to extent it to the multi-class case

by some defined path for learning several binary matching rules and/or some defined path for the final best matching from binary matching rules. We do not go deeper into this, because the PAC learning model itself will play no further role in this work, and is mainly introduced as a basis for the understanding of the structural risk minimization framework we will now start to derive. We present the structural risk minimization principle as it is given in Shawe-Taylor et al. (1996) and Shawe-Taylor and Bartlett (1998): this framework differs slightly from the original framework of Vapnik (1995) in that it makes implicit priors on hypotheses space in the latter framework explicit.

In Definition 3.4.2 of PAC learnability we have to argue on the basis of some fixed but unknown class of true concepts $\mathcal{C}$. It is possible to bound the error probability also without such a restriction.

General bounds can be derived on the basis of an analysis of the capacity of hypotheses spaces to partition some finite set of descriptions into two groups:

Definition 3.4.3 (Growth function and VC-dimension).

Let $\mathcal{H}$ be a hypotheses space $\mathcal{H}(\mathbf{\Omega} \rightarrow \{0,1\})$. Let $\mathbf{\Omega}_N := \{\omega_1, ..., \omega_N\} \subset \mathbf{\Omega} \subseteq \mathbb{R}^K$ be a set of descriptions in the input space. Let

$$\mathcal{H}_{|\mathbf{\Omega}_N} := \{(h(\omega_1), ..., h(\omega_N)), h \in \mathcal{H}\} \subseteq \{0,1\}^N$$

denote the set of all dichotomies on this set that can be induced by $\mathcal{H}$.

The growth function $G_{\mathcal{H}}(\circ) : \mathbb{N} \rightarrow \mathbb{N}$ relates the size N of potential sets with the maximum number of dichotomies that the hypotheses space can induce over it:

$$G_{\mathcal{H}}(N) \;\; := \;\; \max\left\{\left|\mathcal{H}_{|\mathbf{\Omega}_N}\right|, \; \mathbf{\Omega}_N \in \mathbf{\Omega}^N\right\}.$$

If $G_{\mathcal{H}}(N) = 2^N$ for some set $\mathbf{\Omega}_N$ all dichotomies on $\mathbf{\Omega}_N$ can be fitted by the hypotheses space, we say $\mathcal{H}$ *shatters* $\mathbf{\Omega}_N$. The Vapnik-Chernovenkis dimension **VC**-dimension finally determines the maximum size N up to which the hypotheses space can shatter some set:

$$\mathbf{VC}(\mathcal{H}) \;:=\; \max\left\{N : G_{\mathcal{H}}(N) = 2^N\right\}.$$

The Vapnik-Chernovenkis dimension measures the flexibility of the hypotheses space and in that it is a specific measure of its complexity.

The following bound on the error probability in dependence on the **VC**-dimension is based on Shawe-Taylor et al. (1996):

Theorem 3.4.1 (Bound for the Error Probability in fixed Hypotheses Space). *Let $\mathcal{H}$ be some hypotheses space with $\mathbf{VC}(\mathcal{H}) = d$. Let the descriptions $\{\omega_1, ..., \omega_{N_\mathbf{L}}\}$ in $\mathbf{L}$ be the realizations of some random sample from $(\mathbf{\Omega}, \mathcal{A}, P_{\vec{W}})$. If the learning system finds some hypothesis $h \in \mathcal{H}$ with no training error, $\mathbf{EP}(h, P_{\vec{W}|\mathbf{L}}) = 0$, then with probability $\mathbf{P} \geq 1-\delta$ the error probability $\mathbf{EP}(h, P_{\vec{W}})$ will be smaller than the following bound:*

$$b_{\mathbf{EP}}(N_\mathbf{L}, d, \delta) \;=\; \frac{4d}{N_\mathbf{L}} \ln\left(\frac{2eN_\mathbf{L}}{d}\right) + \frac{4}{N_\mathbf{L}}\left(\ln\left(\frac{4}{\delta}\right)\right).$$

Some explanation Note that d must be much smaller than $N_\mathbf{L}$ for the bound to be meaningful in the sense that it is below the natural bound of one.

Let $x := \frac{d}{N_{\mathbf{L}}}$, $0 < x \leq 1$:

$$\begin{aligned} b_\varepsilon(N_{\mathbf{L}}, x, \delta) &< 1 \\ \Leftrightarrow 4x \ln\left(\frac{2e}{x}\right) + \frac{4}{N_{\mathbf{L}}}\left(\ln\left(\frac{4}{\delta}\right)\right) &< 1 \\ \Rightarrow 4x \ln\left(\frac{2e}{x}\right) &< 1 \\ \Rightarrow x &< 0.0542 \end{aligned}$$

up to the forth decimal point. And if the **VC**-dimension d is much smaller than $N_{\mathbf{L}}$, this means that only some of the dichotomies in the training sample could have been fitted without training error by any hypothesis in the hypotheses space $\mathcal{H}$. This statement is based on a well-known result in computational learning theory, called *Sauer's Lemma* (Sauer, 1972). Given the true concept is much different from any fitting hypothesis — $(\mathbf{EP}(h, P_{\vec{W}}) > b_\varepsilon(N_{\mathbf{L}}, x, \delta))$ — then the probability $\mathbf{P}$ that such a training sample was generated is bounded, i.e. $\mathbf{P} < (1 - \delta)$. For more details see Shawe-Taylor et al. (1996).

Based on the dependency of the bounds of the error probability on the **VC**-dimension, a reasonable strategy to select an 'optimal' hypothesis with no training error is based on the **VC**-dimension of the hypotheses space in which this hypothesis can be found.

For that purpose, one structures the overall hypotheses space in a series of nested subspaces with increasing **VC**-dimension. Also, one imposes a prior distribution on this hierarchy that quantifies the uncertainty one has about which is the first subspace in which we will find the "true" concept. The bound for the error probability of a hypothesis with no training errors then

will depend on the **VC**-dimension of the first subspace in the hierarchy the hypothesis belongs to (see Shawe-Taylor and Bartlett, 1998):

Theorem 3.4.2 (Bound for the Error Probability in Structured Hypotheses Space). *Consider some hierarchy of nested hypotheses spaces*

$$\mathcal{H}_1 \subseteq \mathcal{H}_2 \subseteq \ldots \subseteq \mathcal{H}_d \subseteq \ldots \tag{3.4.3}$$

where $\mathbf{VC}(\mathcal{H}_d) = d,\ d \in \mathbb{N}$. *Let* $\left\{p_d \in \mathbb{R}_0^+,\ d \in \mathbb{N},\ \sum_{d=1}^{\infty} p_d = 1\right\}$ *be some series of non-negative numbers representing our prior uncertainty about which subspace to use. We call this the* prior on the hypotheses structure. *Let the descriptions* $\{\omega_1, ..., \omega_{N_\mathbf{L}}\}$ *in* $\mathbf{L}$ *be the realizations of some random sample from* $(\mathbf{\Omega}, \mathcal{A}, P_{\vec{W}})$. *Let* d *be the minimum* **VC***-dimension of some hypotheses space* $\mathcal{H}_d$ *in which the learning system finds some hypothesis* h_d *with no training error,* $\mathbf{EP}(h_d, P_{\vec{W}|\mathbf{L}}) = 0$.

Then with probability $\mathbf{P} \geq 1-\delta$ *the error probability* $\mathbf{EP}(h_d, P_{\vec{W}})$ *will not be larger than the following bound:*

$$b_{\mathbf{EP}}(N_\mathbf{L}, d, \delta) \quad = \quad \begin{cases} \frac{4d}{N_\mathbf{L}} \ln\left(\frac{2eN_\mathbf{L}}{d}\right)) + \frac{4}{N_\mathbf{L}} \left(\ln\left(\frac{1}{p_d}\right) + \ln\left(\frac{4}{\delta}\right)\right) & \text{for } p_d > 0, \\ \text{undefined} & \text{otherwise,} \end{cases}$$

provided $d \leq m$.

Using the *principle of structural risk minimization*, the error bound is the performance measure to rank different hypotheses in a large hypotheses space among all those with no training errors:

Principle 3.4.2 (Structural Risk Minimization).

Given a hierarchy of nested hypotheses spaces and a prior on the structure as in Theorem 3.4.2, and some hypotheses with no training errors on $\mathbf{L}$*, choose that hypothesis* $h_d \in \mathcal{H}_d, d \in \mathbb{N}$ *that has the lowest bound* $b_{\mathbf{EP}}(N_{\mathbf{L}}, d, \delta)$ *for its error probability* $\mathbf{EP}(h_d, P_{\vec{W}})$*.*

The prior $p_d > 0, d \in \mathbb{N}$, on the hypotheses structure has naturally to decrease for increasing $d \to \infty$ at least beyond some point $D \in \mathbb{N}$ to satisfy the constraint $\sum_{d=1}^{\infty} p_d = 1$. In other words, we bias our search towards hypotheses spaces with low complexity in terms of their **VC**-dimension. In that way the structural risk minimization principle is one implementation of **the** principle of inductive inference:

Principle 3.4.3 (Ockham's Razor).

"Pluralitas non est ponenda sine neccesitate" or "plurality should not be posited without necessity." The words are those of the medieval English philosopher and Franciscan monk William of Ockham (ca. 1285-1349). [...] Ockham's razor is also called the principle of parsimony. These days it is usually interpreted to mean something like "the simpler the explanation, the better" or "don't multiply hypotheses unnecessarily." (Carroll, 1994-2002).

Though Ockham's razor is almost globally successfully used, so far no-one has succeeded to prove optimality of Ockham's razor from first principles, though attempts have been made. Wolpert (1992) demonstrates the counter-evidence. Mitchell (1997) states two main problems that arise when trying to prove Ockham's razor in general:

1. within hypotheses spaces typically there does not exist an unique definition of their length and one can define and justify different

orderings of the hypotheses on the basis of different definitions of length, and

2. the definition of some ordering is even more arbitrary if one wants to compare hypotheses from different hypotheses spaces.

For example for TREE, we can motivate to measure length in terms of the depth of the tree D or the number of leaf nodes $\mathbf{N}^{[l]} \subseteq \mathbf{N}$. These definitions are related, but induce different orderings on the set of trees. Among all trees with no training errors, according to Ockham's razor we could justify to select different trees as 'the best' whichever definition of length we use.

It might seem that the ordering according to the **VC**-dimension was unique and 'the best'. Yet, there are many different ways to define nested structures fulfilling (3.4.3) with increasing **VC**-dimension within the same hypotheses space $\cup_{d=1}^{\infty}\mathcal{H}_d \subseteq \mathcal{F}(\mathbf{\Omega} \rightarrow \{0,1\})$. And there are other complexity measures, also related to the spaces capability to fit dichotomies of the training set that may even improve the error bounds in the sense that they are tighter (Shawe-Taylor and Bartlett, 1998, "fat shattering dimension")

Mitchell (1997) speculates, why Ockham's razor is so successful. In a very broad interpretation, this says, Ockham's razor works, because human beings can only reason with short explanations, and thus evolution has made us to perceive the world such that our internal representations of the world lead to short descriptions.

3.4.2 Example: Support Vector Machine

As a short reminder, the matching rule of a SVM with RBF-kernel $\mathbf{K}(\circ,\circ|\sigma^2)$ and reference points $\mathbf{\Omega}_N \subset \mathbf{\Omega} \subseteq \mathbb{R}^K$ is given by (cp. (3.3.21)):

$$\hat{c}(\omega|\mathtt{svm}) \quad = \quad \begin{cases} \arg\max_{c\in\mathbf{C}} \left\{ \frac{K(\vec{w}_c,\omega|\sigma^2)+\theta_c}{\sqrt{K(\vec{w}_c,\vec{w}_c)}} \right\} & \text{if } \omega\in\mathbf{B}(\mathbf{\Omega}_N), \\ \text{undefined} & \text{otherwise.} \end{cases}$$

And the pair of parameters $(\vec{w}_c,\theta_c)\in\mathbb{R}^{K+1}$ has to meet the constraint:

$$\min_{n=1}^{N} \left\{ \left| K(\vec{w}_c,\omega_n|\sigma^2) + \theta_c \right| \right\} = 1, \ c\in\mathbf{C} \tag{3.4.4}$$

The last restriction induces two hyperplanes that are parallel to the separating hyperplane whose distance defines the margin

$$\mathbf{mr}(\mathbf{B}(\mathbf{\Omega}_N),\Phi(\vec{w}),\theta) = 2\left(\|\Phi(\vec{w})\|\right)^{-1}$$

of the separating hyperplane (cp. (3.3.16)).

For an individual concept, the larger the margin, the lower is the **VC**-dimension of the corresponding set of all matching rules with separating hyperplanes with margins that are at least as large. This statement is valid, if the matching rules are defined only on the feature reference ball, as we do it. That is the link of support vector machines to the structural risk minimization principle. The theorem is given by Vapnik (1995):

Theorem 3.4.3 (VC-Dimension and Margins). *Let* $\mathbf{hp}\left(\mathbf{B}(\mathbf{\Omega}_N),\Phi(\vec{w}),\theta\right)$ *be some hyperplane in feature space* $\mathbf{F}$ *in canonical form with respect to reference points* $\mathbf{\Omega}_N \subseteq \mathbf{\Omega} \subseteq \mathbb{R}^K$. *Let* $\dim(\mathbf{F})\in\mathbb{R}^\infty$ *denote the possibly infinite dimension of* $\mathbf{F}$. *Let* R *be the radius of the feature reference ball* $\mathbf{B}(\mathbf{\Omega}_N)$.

Let the corresponding margin $\mathbf{mr}(\mathbf{B}(\mathbf{\Omega}_N), \Phi(\vec{w}), \theta) = 2\left(\|\Phi(\vec{w})\|\right)^{-1}$ *(3.3.16) be bounded from below by the following constraint:*

$$\|\Phi(\vec{w})\| \quad \leq \quad A \in \mathbb{R}.$$

Then the hypotheses space $\mathcal{H}_{R(\mathbf{X}_N)}$ *of matching rules restricted to the feature reference ball (cp. (3.3.21) for arbitrary Kernel) has* **VC***-dimension bounded by*

$$\mathcal{H}_{R(\mathbf{X}_N)} \quad \leq \quad \min\left\{R^2(\mathbf{X}_N)A^2, \dim(\mathbf{F})\right\} + 1.$$

Now assume we use the descriptions $\{\omega_1, ..., \omega_{N_\mathbf{L}}\}$ in the training set $\mathbf{L}$ as reference points $\mathbf{\Omega}_{N_\mathbf{L}}$. And we define an ordering of the hypotheses space of all hyperplanes within the feature ball $\mathbf{\Omega}_{N_\mathbf{L}}$ by decreasing size of the margin. We have thus introduced a specific hierarchy on the hypotheses space of all such hyperplanes.

A separating hyperplane with respect to the training set leads to a matching rule with no training errors: all members of one concept have to lie above the hyperplane, all members of the counter concept below. So far we use natural numbers as concept numbers, for two competing concepts that is $\mathbf{C} = \{1, 2\}$. For separating hyperplanes it is more convenient to use another numbering, namely:

$$y(c) \quad := \quad \begin{cases} 1 & \text{if } c = 1, \\ -1 & \text{if } c = 2. \end{cases}$$

With this coding, we can rewrite the restriction in (3.4.4) such that all examples of the training set that are members of concept 1 lie above ($y_n = 1$),

and all examples of the competing concept 2 below the hyperplane ($y_n = -1$), $n = 1, ..., N_{\mathbf{L}}$:

$$\min_{n=1}^{N_{\mathbf{L}}} \left\{ y_n \left(K(\vec{w}, \omega_n | \sigma^2) + \theta \right) \right\} \quad = \quad 1. \tag{3.4.5}$$

Choosing among all hyperplanes that fulfill this restriction the one with largest margin now corresponds with choosing that hypothesis with no training errors out of that hypotheses subspace with lowest **VC**-dimension.

The performance measure to minimize that ranks all hypotheses $h \in \mathcal{H}_{R(\mathbf{L})}$ with no training errors is thus

$$\text{score}(h|\texttt{svm}) \quad = \quad \|\Phi(\vec{w})\| \,. \tag{3.4.6}$$

Unfortunately, though, this does not lead to the minimization of the structural error bound according to Theorem 3.4.2. A necessary condition in the corresponding proof is the fact that the hierarchy on the hypothesis is determined independent of the data. A lot of work on support vector machines deals with reasonings why large margins of support vector machines correspond to low error probability, at least in practise. Burges (1998, Section 7) presents some of these approaches. Shawe-Taylor and Bartlett (1998) derive a specific principle for structural risk minimization over data-dependent hierarchies, where the "data driven" error bound can be understood as a posterior bound, given a preference relation over the hypotheses in the overall hypotheses space, as such a preference relation can be interpreted as improper prior.

3.4.3 Realizable Concepts

Again we assume ideal concepts $e_c(\circ|\mathbf{\Omega})$, $c\in\mathbf{C}$, based on optimal descriptions $\omega\in\mathbf{\Omega} \subseteq \mathbb{R}^K$. Yet, these optimal descriptions are not available and we will have to match descriptions $\vec{x}\in\mathbf{X}$, with realizable concepts $e_c(\circ|\mathbf{X})$, $c\in\mathbf{C}$ that approximate the ideal concepts.

For non-optimal descriptions the mean loss on the input space (3.4.1) is normally not considered as performance criterion.

In machine learning for non-optimal descriptions $\vec{x}\in\mathbf{X}$ and numbers $c\in\mathbf{C}$ of ideal competing concepts $\vec{e}_{\mathbf{C}} : \mathbf{\Omega} \rightarrow \mathbf{U}^G$ one typically assumes some unknown joint distribution over $P_{\mathbf{C},\vec{X}}$. To understand how this can be related to the idea of "noise corrupted descriptions", we will give one possible explanation here:

We can derive a probabilistic relation between non-optimal descriptions and concept numbers by assuming that future optimal descriptions $\omega\in\mathbf{\Omega}$ are sampled from some population of optimal descriptions according to some general probability model $(\mathbf{\Omega}, \mathcal{A}(\mathbf{\Omega}), P_W)$.

This optimal description is the instance of the concept with number $c(\omega)$. This defines some random variable $C(\circ) : \mathbf{\Omega} \rightarrow \mathbf{C}$ on $(\mathbf{\Omega}, \mathcal{A}, P)$. The corrupted version of the optimal description $\vec{x}(\omega)$ can be seen to be the output of a function $\vec{x} = f(\omega, e)$, with $e \in \mathbf{E}$ denoting some other event representing "noise": a random draw from $(\mathbf{E}, \mathcal{A}(\mathbf{E}), P_E)$. In that respect $\vec{X}(\circ, \circ)$ is a random variable on $\mathbf{\Omega}\times\mathbf{E}$ with probability space $\Big(\mathbf{\Omega}\times\mathbf{E}, \mathcal{A}(\mathbf{\Omega})\times\mathcal{A}(\mathbf{E})), P_{\vec{W},E}\Big)$. Extending $c(\omega) := c(\omega, e)$ for all $e\in\mathbf{E}$ and $\omega \in \mathbf{\Omega}$, we can define C and $\vec{X}$ as random variables on $(\mathbf{\Omega}\times\mathbf{E})$ with joint probability distribution $P_{C,\vec{X}}$.

The performance of a learnt best-matching rule

$$\hat{c}\left(\vec{x} \mid \sqcup, \Upsilon_{\mathbf{C}|\mathbf{L}}, \vec{w}_{\mathbf{L}}\right) = \hat{c}\left(\vec{x} \mid \mathbf{L}, \mathtt{met}\right), \ \vec{x}{\in}\mathbf{X}$$

is most often defined analogously to (3.4.2) to be the expected error with respect to the joint distribution $P_{C,\vec{X}}$:

$$\begin{aligned} \mathbf{EP}(\hat{c}^{[m]}, P_{C,\vec{X}}) &:= \mathbf{E}_{C,\vec{X}}\left(L^{[01]}\left(C, \hat{c}^{[m]}(\vec{X}|\mathbf{L})\right)\right) \\ &:= \int_{\mathbf{C}\times\mathbf{X}} L^{[01]}\left(c, \hat{c}^{[m]}(\vec{x}|\mathbf{L})\right) p(c,\vec{x}) d\mu(c,\vec{x}) \\ &= \int_{\mathbf{X}}\int_{\mathbf{C}} L^{[01]}\left(c, \hat{c}^{[m]}(\vec{x}|\mathbf{L})\right) p(c|\vec{x}) d\mu(c) p(\vec{x}) d\mu(\vec{x}) \\ &= P_{W,E}\left(\left\{(\omega,e){\in}\mathbf{\Omega}\times\mathbf{E}: \ c(\omega){\neq}\hat{c}^{[m]}(f(\omega,e)|\mathbf{L})\right\}\right). \end{aligned} \tag{3.4.7}$$

We already know the best matching rule that minimizes this expected error: it is the Bayes rule $\hat{c}(\circ|\mathtt{bay})$ in (2.2.18) with respect to the true probability model τ specifying $P_{C,\vec{X}}$ (see end of Section 2.2.2).

According to the true joint probability, there may be examples in the training set $\mathbf{L}$ such that

$$c_n \neq \arg\max_{c{\in}\mathbf{C}} p(c|\vec{x_n}, \mathbf{\Lambda}), n{\in}\{1,...,N_{\mathbf{L}}\}.$$

If one tries to learn competing concepts that make no training errors these examples lead astray from the best competing concepts. Thus, a learning instruction that avoids to fit these "non-typical" examples can lead to a rule with lower error probability.

Misleading examples in the training set cause a problem known as the *overfitting problem*. Overfit can be defined as follows (Mitchell, 1997, cp.):

Definition 3.4.4 (Overfit).

A learnt matching rule $h := \hat{c}\left(\vec{x} \mid \sqcup, \Upsilon_{\mathbf{C}|\mathbf{L}}, \vec{w}_{\mathbf{L}}\right) \in \mathcal{H}$ from some hypotheses space $\mathcal{H}$ overfits the training data $\mathbf{L}$ if there exists another matching rule $\tilde{h} := \hat{c}\left(\vec{x} \mid \sqcup, \tilde{\Upsilon}_{\mathbf{C}|\mathbf{L}}, \vec{\tilde{\pi}}_{\mathbf{L}}\right) \in \mathcal{H}$ that makes more training errors, but has a lower error probability:

$$\begin{aligned} \mathbf{EP}(h, P_{C,\vec{X}|\mathbf{L}}) &< \mathbf{EP}(\tilde{h}, P_{C,\vec{X}|\mathbf{L}}) \\ \mathbf{EP}(h, P_{C,\vec{X}}) &> \mathbf{EP}(\tilde{h}, P_{C,\vec{X}}) \end{aligned}$$

Note that in this definition of "overfit" the performance of a matching rule is compared to the performance of all matching rules from the same common hypotheses space. Alternatively, one could define overfit in terms of the error probability of the Bayes rule with respect to the true joint probability $P_{C,\vec{X}}$, the best, yet unknown rule. The practical use of the latter, though, is void as we will never be able to calculate nor to estimate it without making representational assumptions.

There are two basic strategies to fight overfitting:

1) split the training set $\mathbf{L}^*$ in a learning set $\mathbf{L}$ and a validation set $\mathbf{V}$. Learn various hypotheses $\mathcal{H}_{\mathbf{L}}$ on the learning set according to some "inner" performance measure and estimate the error probability $\mathbf{EP}(h, P_{C,\vec{X}})$ of these hypotheses by their empirical error rate on the validation set, $\mathbf{EP}(h, P_{C,\vec{X}|\mathbf{V}})$. Use this as performance measure to rank the hypotheses $h \in \mathcal{H}_{\mathbf{L}}$

2) penalize more complex concepts by the use of a performance measure that combines the training error rate with some complexity measure

$$\text{score}(h) \quad = \quad \mathbf{EP}(h, P_{C,\vec{X}|\mathbf{L}}) + \text{complexity}(\mathcal{H}) \qquad (3.4.8)$$

The justification of the first approach is straightforward, given the i.i.d. assumption of the example generating process is true. Changing the perspective on the scenario slightly, we now regard the loss of a randomly chosen example as the random variable of main interest. This perspective plays a major role in the development of standardized partition spaces, therefore it gets its own name: the *Bernoulli loss experiment*.

Definition 3.4.5 (Bernoulli loss experiment).

Given a learnt matching rule $\hat{c}(\circ|\mathbf{L}, \texttt{met})$ its zero-one loss $Z: \ \mathbf{C}\times\mathbf{X} \to \{0,1\}$ on a random new example $(C, \vec{X})$ is a random variable

$$Z\left(C, \vec{X}|\mathbf{L}, \texttt{met}\right) \quad := \quad L^{[01]}\left(C, \hat{c}(\vec{X}|\mathbf{L}, \texttt{met}\right) \qquad (3.4.9)$$

Given the i.i.d. assumption holds, the losses of examples of the validation set

$$\{z_n : z\left(c_n, \vec{x}_n|\mathbf{L}, \texttt{met}\right), (c_n, \vec{x_n})\in\mathbf{V}\}$$

are realizations of a sample of Bernoulli random variables

$$Z_n \sim \mathcal{Be}(p)\,, \ n = 1, ..., N_{\mathbf{V}},$$

with $p = \mathbf{EP}\left(\hat{c}(\vec{X}|\mathbf{L}, \texttt{met}), P_{C,\vec{X}}\right)$. And the validation error rate

$$\mathbf{EP}(\hat{c}, P_{C,\vec{X}|\mathbf{V}}) \quad = \quad \frac{1}{N_{\mathbf{V}}} \sum_{n=1}^{N_{\mathbf{L}}} z_n.$$

is a "good" estimator of parameter p as it is the maximum-likelihood estimator as well as the unbiased minimum variance estimator thereof.

The Bernoulli loss perspective is also useful to reason against the use of the training error as some estimate of the error probability: after the examples of the training set were used to learn the matching rule, it is inappropriate to model them as outcomes of a random experiment at all.

And before learning, the training set is a random quantity, and the rule to learn $\hat{c}^{[m]}(\circ|\mathbf{L})$ and the losses $Z_1, ..., Z_{N_\mathbf{L}}$ are both dependent on it:

$$\begin{aligned} Z_n &= Z\left(C_n, \vec{X_n}\middle|\left\{(C_1, \vec{X}_1), ..., (C_{N_\mathbf{L}}, \vec{X}_{N_\mathbf{L}})\right\}, \mathbf{L}, \texttt{met}\right), n = 1, ..., N_\mathbf{L}. \\ &= L^{[01]}\left(C_n, \hat{c}(\vec{X_n}|\mathbf{L}, \texttt{met})\middle|\mathbf{L}\right), n = 1, ..., N_\mathbf{L}. \end{aligned}$$

The rule to learn is dependent on the learning set, and the losses depend on the rule to learn and one of the examples in the learning set. These dependencies induce a relationship among the examples and thus they do not form an i.i.d. sample.

Justifications of the second approach all rest upon different formalizations of Ockham's razor (Principle 3.4.3). Most statistical model selection strategies are based on Ockham's razor, as Hansen and Yu (2001) put it into words it *is the soul of model selection.* The minimum description length principle, the minimum message length principle, model selection based on the Akaike information criterion, or the Bayesian information criterion are all formalizations of this idea. For an overview, see Hansen and Yu (2001).

The adaption of the structural risk minimization principle to non-optimal descriptions also fits in that approach. The version of Theorem 3.4.2 that

allows for training errors provides the corresponding bound for the error probability. Shawe-Taylor and Bartlett (1998) introduce more priors there: in Theorem 3.4.2 one specifies the prior $p_d, d{\in}\mathbb{N}$, on the structure, quantifying the uncertainty which subspace in the hierarchy to use, now additionally within each subspace $\mathcal{H}_d$ a prior with non-negative values $q_{d,k} : \sum_{n=1}^{N_{\mathbf{L}}} q_{d,k} = 1$, for all $d{\in}\mathbb{N}$ specifies the uncertainty on the number of errors a best hypothesis within this subspace will make on some training data of size $N_{\mathbf{L}}$.

Theorem 3.4.4 (Bound for the Error Probability in Structured Hypotheses Space with Noise).

Again consider some hierarchy of nested hypotheses spaces

$$\mathcal{H}_1 \subseteq \mathcal{H}_2 \subseteq \ldots \subseteq \mathcal{H}_d \subseteq \ldots$$

with $\mathbf{VC}(\mathcal{H}_d) = d,\ d{\in}\mathbb{N}$. *Let the non-negative numbers* $p_d,\ d{\in}\mathbb{N}$, *with* $\sum_{d=1}^{\infty} p_d = 1$ *define a prior on the hypotheses structure, and for all* $d{\in}\mathbb{N}$ *let* $q_{d,k} : \sum_{n=1}^{N_{\mathbf{L}}} q_{d,k} = 1$ *define a prior on the number of minimum training errors on a training set of size* $N_{\mathbf{L}}$ *of some hypothesis* $h_d{\in}\mathcal{H}_d,\ d{\in}\mathbb{N}$. *Let the descriptions in* $\mathbf{L}$ *be the realizations of a random sample according to the distribution* $P_{C,\vec{X}}$.

If the learning system finds some hypothesis $h_d{\in}\mathcal{H}_d$ *with* k *training errors, then with probability* $\mathbf{P} \geq 1{-}\delta$ *its error probability* $\mathbf{EP}(h_d, P_{\mathbf{C},\vec{X}})$ *will be smaller than the following bound:*

$$b_{\mathbf{EP}}(N_{\mathbf{L}}, d, k, \delta) \quad = \quad \begin{cases} 2\left(\frac{k}{N_{\mathbf{L}}} + \frac{2d}{N_{\mathbf{L}}} \ln\left(\frac{2eN_{\mathbf{L}}}{d}\right) + \frac{2}{N_{\mathbf{L}}} \ln\left(\frac{4}{p_d q_{d,k} \delta}\right)\right) & \text{for } q_{d,k} p_d > 0, \\ undefined & \text{otherwise,} \end{cases}$$

provided $d \leq N_{\mathbf{L}}$.

These error bounds of structural risk are all very loose, and can not really be understood in terms of some "guarantee of quality" as they are typically at least larger than 0.5 — which is the error probability we get if we match descriptions with two competing concepts by tossing a fair coin.

Thus, we prefer the error bound to be interpreted as a performance measure that combines training error with a founded penalty for the complexity to find the point where Ockham's razor shaves because further "plurality would be posited without necessity".

3.4.4 Example: Support Vector Machine

To allow for noise in SVM classifiers, we only have to modify the restriction (3.4.5)

$$\min_{n=1}^{N_{\mathbf{L}}} \left\{ y_n \left(K(\vec{w}, \omega_n | \sigma^2) + \theta \right) \right\} \quad = \quad 1.$$

that was introduced to force the examples of the two competing concepts to lie either all above or below the hyperplane.

One introduces so-called *slack variables* $\xi_n \geq 0,\ n = 1, ..., N_{\mathbf{L}}$ such that description $\vec{x}_n$ can lie on the wrong side if $\xi_n > 0,\ n = 1, ..., N_{\mathbf{L}}$. It is now easier to write the description as individual constraints on each example:

$$y_n \left| K(\vec{w}, \vec{x}_n | \sigma^2) + \theta \right| \quad \geq \quad 1 - \xi_n,\ n = 1, ..., N_{\mathbf{L}}. \tag{3.4.10}$$

Note that in this representation one does not require any more the bound of "1" to be reached. This is redundant, because by maximizing the margin we will reach the bound anyway.

If an error occurs, the corresponding slack variable ξ_n exceeds unity. Thus $\sum_{n=1}^{N_{\mathbf{L}}} \xi_n$ is an upper bound on the number of training errors.

And the performance measure is no longer based on the size of the margin alone, but combined with a penalty for training errors. We get another performance measure according to the second basic strategy (3.4.8) to fight overfitting. The self-evidence one entitles to this strategy is articulated by Burges (1998), when he introduces it. The quotation is adapted to nomenclature and notation in this work by the words and formula in squared brackets: *Hence, a natural way to assign an extra cost for errors is to change the* [performance measure] *to be minimized from* $[\|\Phi(\vec{w})\|]$ *to* $\left[\|\Phi(\vec{w})\| + C\left(\sum_{n=1}^{N_{\mathbf{L}}} \xi_n\right)\right]$, *where* $C[\in\mathbb{R}]$ *is a parameter to be chosen by the user, a larger* C *corresponding to assigning a higher penalty to errors.*

3.4.5 Conclusion

In machine learning the basic assumptions about the generating process of data differ from those in statistics. In consequence, the principles of "good" learning differ, as well as measures to assess the performance of matching rules, and hypotheses spaces in which to search for best matching rules.

Common to the approaches to classification rule learning and concept learning is the dependence of the performance measures on the i.i.d. assumption of examples. They differ in the assumptions about the probability space on which sampling takes place. In statistics the performance measures based on expected loss depend on an assumed and often quite restrictive class of distributions. In machine learning, one tries to avoid this type of assumptions,

and uses performance measures based on bounds of expected loss derived for (almost) any underlying probability distribution.

3.5 Optimization and Search

Assume we have specified the joint components $J1 : \mathcal{F}_{\Upsilon}$ and $J2 : \sqcup$ of some best-matching competing concepts (Definition 3.3.3). And we have also chosen some performance measure to rank the matching rules with differently chosen concept specific parameters $S1 : \Upsilon_c \subset \Upsilon$ and $S2 : \pi_c \in \mathbb{R}$ for $c \in \mathbf{C}$ on the basis of the examples in $\mathbf{L}$. The last step in learning is to find best parameters according to the performance measure.

In statistics, restrictions on the class of distributions assumed to govern the data generating process for past and future data, often enable heavy down-sizing of the hypotheses space $\mathcal{H}$ of potential 'optimal' classification rules. Without these restrictions, the hypotheses spaces $\mathcal{H}$ in machine learning are typically very broad. Finding 'optimal' solutions in broad spaces can be almost arbitrarily difficult. *Optimization* and *search* are two terms to describe ways to derive optimal solutions. These terms have different connotations, though. The following explanations are given in Bridle (1995):

An optimal search method is one that always finds the best solution (or a best solution, if there is more than one). For our purposes best is in terms of the value of the performance measure. Optimization can be defined as the process of finding a best solution, but it is also used, more loosely, meaning to find a sequence of better and better solutions.

Although there are optimization techniques which are not normally thought

of as search, and search methods which are not optimal or not defined as optimizing anything, most often we are dealing with search methods which seek to optimize a well-defined criterion, and usually we understand the search method in terms of the way it approximates to an optimal search.

Bridle (1995) distinguishes combinatorial optimization problems, where a solution is a sequence or more complicated data structure, from (non-linear) continuous optimization problems, where a solution is a vector of real numbers. We will reserve the term search for combinatorial problems and optimization for continuous problems.

Implicitly, optimization is often assumed to be the easier task compared to search in terms of computational resources needed, though this is of course not true in general. In the following, we will sketch important features of optimization and search relevant to this work while describing the implemented optimization and search strategies of the examples LDA, QDA, CNB, SVM and TREE.

3.5.1 Examples: Plug-in and Predictive Bayes Rules

The plug-in Bayes rules of LDA and QDA and predictive Bayes rules of CNB are all solutions of optimization problems in the sense above. They possess two important and quite desirable features:

1. they are optimal search methods, because they always find the or a best solution, if such exist, and
2. they can be computed directly — analytically or numerically — with standard algorithms.

The statistical models in which these strategies can be successfully pursued are very restrictive. In particular, statisticians are confronted with combinatorial problems if they are uncertain about structural components of the model distributions like e.g. which multivariate dependencies and/or which predictors to consider. In general, whenever one has to learn intensional definitions with basic function $f_\upsilon \in \mathcal{F}_\Upsilon(\mathbf{X} \to \mathbb{R})$, where the parameter $\upsilon \in \Upsilon$ determines the actual domain of $f_\upsilon(\circ) : \mathbf{X}^{[\mathbf{M}]} \to \mathbb{R}$, one has to solve a combinatorial problem.

For example, we performed variable selection with LDA and QDA in (Sondhauss and Weihs, 2001b) for the problem of business cycle prediction that is analyzed in Chapter 7. Given thirteen potential predictor variables ($K = 13$) we selected the best two by *exhaustive search* on the space of all potential combinations of two predictors $\binom{13}{2} = 78$. If we had intended to find the best subset of predictors among all subset of predictors, exhaustive search would have become infeasible on the $2^13 = 8192$ combinations.

For plug-in Bayes rules not only that these combinatorial problems can be computationally demanding, but also we run into the difficulties of potential overfit. The performance measures for best point estimation are only valid for ranking within the hypotheses space corresponding to the statistical model based on a fixed set of predictors, but not to rank hypotheses corresponding to different statistical models. In (Sondhauss and Weihs, 2001b) we solved this problem by a method related to the first strategy to fight overfitting: cross validation. For each statistical model a best plug-in rule was learnt on a sub-sample of the training data. Its performance was assessed by the empirical error on the rest of the training data. This was repeated for different

partitions of the training data into learning set and validation set. The overall performance measure used in cross validation is the average error rate on the validation sets according to the various repetitions of the validation.

For predictive Bayes rules the appropriate strategy to rank hypotheses from different statistical models consists of quantifying a prior over the statistical models, and then performing standard Bayesian updating of these model priors to select the best one. If these priors are motivated by some quantification of the complexity of the models, Bayesian model selection and structural risk minimization methods get close.

3.5.2 Example: Support Vector Machine

To find the best separating hyperplane to determine a SVM classifier, one first has to specify the Kernel type (we use the Gaussian RBF-Kernel in (3.3.17)), then the Kernel specific parameter (the width σ^2 of the RBF-Kernel) and the parameter C that controls the trade-off between margin maximization and training error minimization (Section 3.4.4).

Given the Kernel and these parameters, the hypotheses space for SVM matching rules is still very broad. One of the characteristics, the SVMs are famous for, is the fact that the best concept specific parameters can nevertheless be found with guarantee by an optimization procedure in polynomial time (Burges, 1998).

To minimize the performance measure

$$\|\Phi(\vec{w})\| + C\left(\sum_{n=1}^{N_{\mathbf{L}}} \xi_n\right) \tag{3.5.1}$$

derived in Section 3.4.4 under the constraints given in equation (3.4.10)

$$y_n \left| \mathbf{K}(\vec{w}, \vec{x}_n | \sigma^2) + \theta \right| \quad \geq \quad 1 - \xi_n, \; n = 1, ..., N_{\mathbf{L}},$$

one can switch to a Lagrangian formulation of the problem. For a detailed description see e.g. Burges (1998). As a consequence of the restrictions, the feature parameter $\phi(\vec{w})$ with minimum norm,

$$\phi(\vec{w})^{[opt]} := \arg \min_{\Phi(\vec{w}) \in \mathbf{F}} \left\{ \|\Phi(\vec{w})\| + C \left(\sum_{n=1}^{N_{\mathbf{L}}} \xi_n \right) \right\}$$

can be written as a weighted sum of the features of the descriptions in the training set:

$$\phi(\vec{w})^{[opt]} \quad := \quad \sum_{n=1}^{N_{\mathbf{L}}} \alpha_n y_n \Phi(\vec{x}_n), \tag{3.5.2}$$

with non-negative weights $\alpha_n, n = 1, ..., N_{\mathbf{L}}$. Only some descriptions $\vec{x}_n \in \mathbf{L}$ have a non-zero weight, $C > \alpha_n > 0$, namely those whose features lie on or within the margins of the hyperplane and those that represent training errors. The corresponding descriptions $\{\vec{x}_n \in \mathbf{L} : \alpha_n \neq 0, n = 1, ..., N_{\mathbf{L}}\}$ are the so-called *support vectors*.

The learnt matching rule that combines hyperplanes for concepts $c \in \mathbf{C}$ against all others thus can be written as:

$$\begin{aligned} &\hat{c}(\vec{x} | \mathbf{L}, \texttt{svm}) \\ &= \begin{cases} \arg\max_{c \in \mathbf{C}} \left\{ \frac{\sum_{n=1}^{N_{\mathbf{L}}} \alpha_{c,n|\mathbf{L}} \mathbf{K}(\vec{x}, \vec{x}_n | \sigma^2) + \theta_{c|\mathbf{L}}}{\sqrt{\mathbf{K}(\vec{w}_{c|\mathbf{L}}, \vec{w}_{c|\mathbf{L}})}} \right\} & \text{if } \vec{x} \in \mathbf{B}(\mathbf{X}_{N_{\mathbf{L}}}), \\ \text{undefined} & \text{otherwise.} \end{cases} \end{aligned} \tag{3.5.3}$$

Comparing SVM with RBF Kernel to other RBF classifiers, one sees that support vectors play the role of RBF centers. A clear figure of merit of the

SVM is the principled way of choosing the number and the location of these RBF centers (Schölkopf et al., 1997).

Yet not all parameters that define SVM are automatically learnt. We have to choose the parameters (C, σ^2). In the application in Chapter 7 we did no search or optimization of these parameters. We set them to certain values that performed well in a preliminary study. The original data set in the preliminary study is also part of the analysis in Chapter 7. Using the "optimized" values from preliminary studies on the same data later, certainly is not the most proper way to do in a comparative study, because it gives the respective classifier some advantage. But the other classifiers in this study were also to even larger extends already optimized in this preliminary study and the main question in Chapter 7 is about other properties, thus we did it nevertheless. For the simulated data in Chapter 7, though, this is not only a very efficient and but also a justified approach to use *background knowledge* to fix these values.

To adjust (C, σ^2) otherwise, one can use the Bernoulli loss experiment, and any optimization method that finds extremal points of multidimensional functions to minimize the validation error in dependence of (C, σ^2). One example is the gradient decent algorithm used by Chapelle et al. (2002) for (heavier) SVM model selection. As many of these methods, the gradient decent algorithm might get stuck in local minima, yet, it does not if the error surface is convex. Nevertheless, given certain restrictions on the functions' surface, these algorithms are *optimal* optimization strategies. otherwise, they are *heuristic* optimization strategies (Luenberger, 1973).

Based on model assumptions on the error surface, one can apply methods from statistical experimental design to optimize parameters. We minimized the validation set error with respect to the parameters (C, σ^2) of a specific RBF SVM in Garczarek et al. (2002) using experimental design with a quadratic loss function: We assumed that the error $\mathbf{EP}\left(h(C,\sigma^2), P_{C,\vec{X}|\mathbf{V}}\right)$ can be approximated by quadratic function of $(\log_{10}(C), -\log_e(\sigma^2)/K)$ restricted to the cube $[-1,2]^2$. That way we restricted the search for the best parameters (C, σ^2) on the rectangle $[\frac{1}{10}, 100] \times [\frac{K}{e^2}, Ke]$. Defining 5 optimal experimental points according to a central-composit plan for two variables (see e.g. Weihs and Jessenberger, 1999) the quadratic function was fitted and optimal parameters determined by the minimum of the fitted quadratic function on the rectangle.

3.5.3 Example: Univariate Binary Decision Tree

As we stated earlier, we are confronted with a combinatorial problem if the parameters $\upsilon \in \Upsilon$ determine the actual domain $\mathbf{X}^{[\mathbf{M}]}$ of the basic functions $f_\upsilon \in \mathcal{F}_\Upsilon$ of the intensional definitions.

In case of TREE the class of functions for the intensional definitions was derived in Section 3.3.1 to be:

$$\begin{aligned} J1 : \mathcal{F}_\Upsilon &= \mathcal{F}_{D,\mathbf{N},\vec{k},\vec{t},\vec{w}} \\ &= \left\{ \begin{array}{l} \mathbb{I}_{(-\infty,t(q)]}\left(\circ|\mathbf{X}_{k(q)}\right), \\ t(q) \in \mathbf{X}_{k(q)}, \\ \vec{k} \in \{0,1,...,K\}^{2^{D+1}-1}, \vec{w}_c \in \mathbb{R}^{2^{D+1}-1} \\ q \in \mathbf{N}^{[a]}(D), D \in \mathbb{N}. \end{array} \right\} \end{aligned}$$

Thus, the parameter for the depth of the tree D, the parameters $\mathbf{N} \subseteq \left\{1, 2, ..., 2^{D-1} - 1\right\}$ representing the numbers of the actual nodes in the tree,

and $\vec{k}$ are all involved in determining the domains of the basic functions:

$$\begin{aligned} f(x_{k(q)}) &= \mathbb{I}_{(-\infty,t(q)]}(x_{k(q)}),\ x_{k(q)}\in\mathbf{X}_k \\ & \qquad q\in\mathbf{N}\subseteq\left\{1,2,...,2^{D-1}-1\right\}. \end{aligned}$$

The parameters $D, \mathbf{N}, \vec{k}$ constitute the *tree structure*. And the problem to learn these and to estimate the parameters for the corresponding threshold values $\vec{t}$ and the weight vectors $\vec{w}_c, c\in\mathbf{C}$ will be performed by an organized search through the space of all potential tree structures.

In general, the search through some space is described by a set of operators to transform states and some procedure for applying those operators to search the space. Search is facilitated, if the search space is organized. A natural partial ordering that exists on each hypotheses space is based on the generality of the hypotheses:

Definition 3.5.1 (General to specific ordering).

If any description $\vec{x}\in\mathbf{X}$ that is an instance of hypothesis $h\in\mathcal{H}$ also is an instance of hypothesis $\tilde{h}\in\mathcal{H}$, then $\tilde{h}$ is more general or equal to h:

$$\begin{aligned} \tilde{h}\in\mathcal{H}(\mathbf{X}\rightarrow\{0,1\}) &\geq_g h\in\mathcal{H}(\mathbf{X}\rightarrow\{0,1\}) \\ :\Leftrightarrow \left\{\vec{x}\in\mathbf{X}:\ \tilde{h}(\vec{x})=1\right\} &\supseteq \{\vec{x}\in\mathbf{X}:\ h(\vec{x})=1\} \end{aligned}$$

This partial ordering plays an important role in most methods to learn concepts from training data. "States" in this respect are the partial rankings of hypotheses with respect to other hypotheses and operators define how to generalize or to specialize hypotheses.

On the basis of the general-to-specific ordering one can use strategies of three basic types to find some hypothesis $h \in \mathcal{H}$ with no training errors:

1) *Bottom-up strategies* start with the most specific possible hypothesis in $\mathcal{H}$, minimally generalize this hypothesis if it fails to cover one of the instances of the concept to learn in the training set, and stop if it covers all examples that are instances of the concept to learn. This hypothesis is the output of the search algorithm.
2) *Top-down strategies* start with the most general hypothesis in the hypotheses space, minimally specialize this hypothesis as long as it still covers some examples in the training data that do not belong to the concept to learn, and stop if no non-members are covered any more. This hypothesis is the output of the search algorithm.
3) *Version space* strategies combine both strategies, and a description of all hypotheses in the hypotheses space that have no training error is the output of these strategies.

Concept learning formulated as search problems and the development and analysis of strategies form a central and basic part of machine learning research. We refer to Mitchell (1997) and Langley (1995) for a general introduction into this substantial field, and we restrict our attention to the decision tree learning.

The two core learning algorithms in decision tree learning ID3 (Quinlan, 1986) and its successor C4.5 (Quinlan, 1993) are both *greedy top-down strategies*. The implementation we use `rPart` for "Recursive Partitioning and Regression Trees" is one variant thereof based on the CART system (Breiman et al., 1984).

Top down says, we start with the empty tree and want to specialize. Greedy says we specialize by stepping that way that maximizes some defined *splitting criterion* the most within this single step — not looking further ahead, where another choice might lead to an overall larger profit.

To present the strategy to learn a univariate binary decision tree for some training set $\mathbf{L}$, we show how to learn a tree with no training errors, and then select the strategy to fight overfitting.

To start with, assume we were given some tree structure $D, \mathbf{N}, \vec{k}$ and corresponding threshold values $\vec{t}$. Let $\mathbf{X}^{[q]} \subseteq \mathbf{X}$ denote the set of all descriptions that potentially pass through node $q \in N$:

$$\mathbf{X}^{[q]} := \{\vec{x} \in \mathbf{X} : q \in \mathbf{pt}(\vec{x})\}.$$

And $\mathbf{L}^{[q]} \subseteq \mathbf{X}$ denote all training examples that pass through this node and $\mathbf{L}^{[q,c]} \subseteq \mathbf{X}$ all those among them that are members of concept number c, $c \in \mathbf{C}$:

$$\mathbf{L}^{[q]} := \{\vec{x}_n \in \mathbf{L} : q \in \mathbf{pt}(\vec{x_n})\},\ \mathbf{L}^{[q,c]} := \{c_n, \vec{x}_n \in \mathbf{L} : q \in \mathbf{pt}(\vec{x_n}), c_n = c\}$$

Define $\mathbf{X}^{[q,c]}$ to be the analogue of $\mathbf{L}^{[q,c]}$ on the input space $\mathbf{X}$.

A good quantification of the degree of match $w(c,q)$, $c \in \mathbf{C}$, $q \in \mathbf{N}$ of some description $\vec{x}$ passing through node q with concept number c would certainly be its probability to belong to concept number c given it is processed through node q:

$$w(c,q) \quad := \frac{P_{C,\vec{X}}\left(\mathbf{X}^{[q,c]}\right)}{P_{C,\vec{X}}\left(\mathbf{X}^{[q]}\right)}$$

Of course we do not know these, but we can use the empirical counterparts as estimates:

$$w(c,q|\mathbf{L}) \quad := \quad \frac{\left|\mathbf{L}^{[q,c]}\right|}{\left|\mathbf{L}^{[q]}\right|}, c \in \mathbf{C}, q \in \mathbf{N} \tag{3.5.4}$$

The vector $\vec{w}_{q^{[l]}|\mathbf{L}} = \left(w(1, q^{[l]}|\mathbf{L}), ..., w(G, q^{[l]}|\mathbf{L})\right) \in \mathbf{U}^G$ defines a probability distribution on $\mathbf{C}$.

A decision tree makes no error, if the leaf nodes $n^{[l]} \in \mathbf{N}^{[}l]$ in the tree are "pure" in the sense that only examples that belong to one concept can be found in a leaf node, that is, $w(c, q^{[l]}|\mathbf{L}) = 1$ for some $c \in \mathbf{C}$ Thus, our goal are pure nodes where weight vectors are unit vectors. These have maximum heterogeneity among all vectors $\vec{w} \in \mathbf{U}^G$. Consequently, splitting criteria in decision trees are based on measures for the heterogeneity of the weights in the nodes. We use the empirical entropy in the nodes:

$$\mathbf{et}(q|\mathbf{L}) \quad = \quad \sum_{c \in \mathbf{C}} w(c, q|\mathbf{L}) \log_2 w(c, q|\mathbf{L})$$

Starting in the root node $n = 1$, we have to find the predictor k to which a threshold $t(1) \in \mathbf{X}_{k(1)}$ exists such that the two sets $\mathbf{X}^{[2]}$ and $\mathbf{X}^{[3]}$ induced by the roots corresponding threshold function decrease the entropy:

$$\begin{aligned} f(x_{k(1)}) &= \mathbb{I}_{(-\infty, t(1)]}\left(\circ|\mathbf{X}_{k(1)}\right) \\ \mathbf{L}^{[2]}(k(1), t(1)) &:= \left\{\vec{x}_n \in \mathbf{L}^{[1]} : x_{k(1),n} \leq t(1)\right\} \\ \mathbf{L}^{[3]}(k(1), t(1)) &:= \left\{\vec{x}_n \in \mathbf{L}^{[1]} : x_{k(1),n} > t(1)\right\} \end{aligned}$$

with $\mathbf{L}^{[2]}(k(1), t(1)) \cup \mathbf{L}^{[3]}(k(1), t(1)) = \mathbf{L}^{[1]} = \mathbf{L}$ To decrease the entropy the weighted sum of entropies in the new leafs must be lower than the entropy of the root node. The corresponding splitting criterion that we use is the so-called *information gain*:

$$\begin{aligned} &\mathbf{ig}(q, k(q), t(q)|\mathbf{L}) \\ &\quad = \quad \mathbf{et}(q|\mathbf{L}) - \mathbf{et}(2q|\mathbf{L}, k(q), t(q)) - \mathbf{et}(2q+1|\mathbf{L}, k(q), t(q)) \end{aligned}$$

that can be defined for potential splits in any node.

The best pair $k(1) \in \{1, ..., K\}$ and $t(1) \in \mathbf{X}_{k(1)}$ are found by exhaustive search. This is possible for discrete as well as continuous attributes though theoretically the threshold value $t(1) \in \mathbf{X}_{k(1)}$ for a continuous attribute $k(1)$ can take on any value in the corresponding uncountable space $\mathbf{X}_{k(1)}, \in \{1, ..., K\}$. Yet, only a finite number of them can induce different partitions $\mathbf{L}^{[2]}(k(1), t(1)) \cup \mathbf{L}^{[3]}(k(1), t(1)) = \mathbf{L}^{[1]}$ of the training examples in the previous node, and thus we can

Going greedily from simple-to-complex and stopping with the first tree that fits the data the search strategy induces a so-called *search bias* that prefers trees that can be learnt greedily and that prefers short trees. The bias induced by greediness has to be taken care of in trees where nodes can have more than two successor nodes. If a node can split into as many successors as there are values of discrete attributes, the greedy search combined with the information gain measure would favor attributes with many values over those with fewer values. One should use other splitting criteria then (see e.g. Mitchell, 1997). The bias with respect to short trees is wanted, reasoning as usual with Ockham's razor.

To avoid overfitting one can follow two basic strategies: doing post-pruning or pre-pruning. Post-pruning will typically be based on a Bernoulli loss experiment. Given some large tree with no training errors, find a subtree that minimizes the validation error. Pre-pruning uses either statistical tests or some combination of error and complexity penalty to decide in each node, whether the potential benefit is large enough to justify the growth of the tree. We introduced a hard complexity penalty: tree growth is stopped, if the number of examples is below some minimum. This penalty parameter is optimized with

a Bernoulli loss experiment.

As a reward for the difficulty to solve variable selection problems, corresponding hypotheses spaces offer the possibility to find good (in terms of error probability) best-matching rules that are at the same time easy to interpret. For example, the hypotheses space of TREE is flexible, therefore one may find better rules than within very restricted spaces. And decision tree rules represent concepts that are decomposed in "subconcepts" – the paths – based on only small subsets of predictor variables – the predictor variables on a path. Therefore, they are potentially easier to understand compared to concepts according to flexible yet inherently high-dimensional hypotheses spaces like those of neural networks and SVM.

Chapter 4

Summing-Up and Looking-Out

The preceding chapters provide the theoretical basis for the development of new methods that will take place in the following chapters. This Chapter serves as transmitter.

4.1 Classification Rule Learning and Concept Learning

In chapters 2 and 3, we introduced statistical approaches for learning classification rules, and machine learning approaches for finding best-matching competing concepts. We embedded the statistical learning of classification rules in the concept learning framework. We derived an assimilated formulation of the problem relevant in this work. This can be comprised as follows:

1. We consider classification problems that are based on some Kdimensional real-valued vector $\vec{x}(\omega)\in\mathbf{X} \subset \mathbb{R}^K$ measured on objects $\omega\in\mathbf{\Omega}$. The task is to assign a new object $\omega\in\mathbf{\Omega}$ to one of the classes based on the information in $\vec{x}$, and training data consisting of $N_{\mathbf{L}}$ objects with known classes

$\{c_1, ..., c_{N_\mathbf{L}}\}$ and measurements $\{\vec{x}_1, ..., \vec{x}_{N_\mathbf{L}}\}$.

2. We consider comparative concept matching problems based on descriptions $\vec{x}\in\mathbf{X}$ that are a corrupted versions of optimal descriptions $\omega\in\mathbf{\Omega}$. Optimal descriptions are instances of one concept out of a set of competing ideal concepts $\vec{e}_\mathbf{C}(\circ) : \mathbf{\Omega} \to \mathbf{U}^G$. The task is to match a new description $\vec{x}\in\mathbf{X}$ with the number of the ideal concept $c\in\mathbf{C}$ given some training data $N_\mathbf{L}$ of corrupted descriptions $\{\vec{x}_1, ..., \vec{x}_{N_\mathbf{L}}\}$ and the numbers $\{c_1, ..., c_{N_\mathbf{L}}\}$ of corresponding ideal concepts.

We do not continue to use both terminologies in parallel. We will express ourselves in statistical technical terms, as we are more familiar with them.

4.2 Best Rules Theoretically

One of the main characteristics where machine learning and statistical approaches to classification rule learning differ, is the size of the hypotheses spaces learnt classification rules may come from. As we have shown, this difference is the impact of the different views on what can be or should be expected.

The hypotheses spaces in statistics are typically downsized by distributional assumptions on the underlying data generation process. If these assumptions are valid, optimal classification rules in terms of error probability can be efficiently learnt in these small spaces. Predictive Bayes rules are optimal in this respect if the distributional assumptions about the sample distribution is valid and the prior distribution is accepted. Plug-in Bayes rules are asymptotically

optimal if the distributional assumptions about the sample distribution are valid.

In machine learning, mostly one does not pose a certain distributional assumption other than there is some i.i.d. data generating process. One prefers broad hypotheses spaces over smaller ones to avoid large representational biases. One problem in large hypotheses spaces is over-fitting and appropriate performance measures have to be defined to avoid it. The other problem is the computational effort to search in these spaces. Heuristic search strategies like the ones for decision trees can find good solutions in the large spaces within acceptable computational time, yet they do not necessarily find optimal solutions in these spaces. In that way, search bias is favored over representational bias. An important feature of the support vector machines is the fact that one can guarantee to find optimal solutions with respect to the underlying performance measure. Yet, the connection of the optimality with respect to this performance measure to the true error probability of the learnt classification rule is not clear. They can be connected by prior assumptions to the error bounds in structural risk minimization. Yet, as long as these error bounds can be far above one, and are not below 0.5 for two competing concepts, the optimality with respect to these error bounds does not give valuable guaranteed information about the actual error probability of the learnt rule.

We presented the different approaches to show that each of them is justifiable and questionable at the same time. Even more so in data mining applications, because all the underlying performance criteria and measures rely on the assumption that the training data are appropriately modelled to

be the realization of some i.i.d. sample, and that future observations will be generated according to the same process as the training data was.

In data mining applications the data generating process is not under control, neither according to a specific deterministic mechanism nor according to a specific random mechanism. In consequence, we tape up the position that all these approaches are justifiable heuristic search strategies.

4.3 Learnt Rules Technically

Technically, learnt rules from machine learning and statistics are very much alike: most of them base their assignment of objects into classes on certain transformations of the respective observations for each of the considered classes. As we have shown in Chapter 2 in statistics classification rules are typically learnt Bayes rules, and thus transformations are most commonly based on the estimation of conditional class probabilities. In machine learning we saw in Chapter 3 that transformations can be understood as an assessment of the competitive degree of match of descriptions with concepts.

These transformations are taken from some hypotheses space $\mathcal{H} \subset \mathcal{F}\ (\mathbf{X} \rightarrow \mathbf{C})$ whose content depends on the representational choices defining the classification method (Section 3.3). Given some training set $\mathbf{L}$ of examples, and on the basis of a specific performance measure (Section 3.4) and a specific optimization or search strategy (Section 3.5) 'optimal' transformations are learnt:

$$m(c, \circ | \mathbf{L}, \mathbf{\Lambda}) : \mathbf{X} \rightarrow \mathbb{R},\ c \in \mathbf{C}.$$

The size of these transformations gives information on the strength of membership of the object in the classes. Without loss of generality, we assume higher values to indicate stronger membership. That is, these $m(c, \vec{x}|\mathbf{L}, \mathbf{\Lambda})$, $c \in \mathbf{C}$, $\vec{x} \in \mathbf{X}$ are interpreted as *membership values*. For the assignment into different classes, the learnt rule does not distinguish any longer between objects with the same membership vectors $\vec{m}(\vec{x}|\mathbf{L}, \mathbf{\Lambda}) := (m(1, \vec{x}|\mathbf{L}, \mathbf{\Lambda}), ..., m(G, \vec{x}|\mathbf{L}, \mathbf{\Lambda})) \in \mathbf{M}$ for some membership space $\mathbf{M} \subseteq \mathbb{R}^G$. That way, for any potentially high dimensional space of predictors $\mathbf{X}$, all these classifiers do a dimension reduction from $\mathbf{X}$ into $\mathbb{R}^G$.

Irrespective of the various derivations of membership values, the manner of assignment is always the same: The rule assigns to the class with highest membership value (cp. (2.2.18) and (3.3.8)):

$$\hat{c}(\vec{x}|\mathbf{L}, \texttt{met}) \quad = \quad \arg\max_{c \in \mathbf{C}} m(c, \vec{x}|\mathbf{L}, \texttt{met}).$$

If we had complete knowledge of the relationship between predictors and classes that can be expressed in a probability model τ — which includes deterministic relationships as special cases — we could use the Bayes rule with respect to τ (cp. (2.2.18)):

$$\hat{c}(\vec{x}|\texttt{opt}) \quad = \quad \arg\max_{c \in \mathbf{C}} p(c \mid \vec{x}, \tau), \quad \vec{x} \in \mathbf{X}.$$

This rule minimizes the error probability $\mathbf{EP}\left(\hat{c}, P_{C,\vec{X}}\right)$ for all $\hat{c} \in \mathcal{F}\,(\mathbf{X} \to \mathbf{C})$. That is, why Fukunaga (1990) calls the vector of the corresponding membership values *optimal features*:

$$\vec{p}(\vec{x}|\tau) \in \mathbf{U}^G, \ \vec{x} \in \mathbf{X}. \tag{4.3.1}$$

If these optimal features were known, we could use them to answer any question of interest about the interplay of predictors and class membership. The idea of standardized partition spaces as we will introduce them in the next Chapter is motivated by the attempt to make any argmax rule comparable to the 'true' or 'optimal' Bayes rule. We want to use learnt membership functions of argmax classifiers as surrogates for optimal features.

Chapter 5

Introduction to Standardized Partition Spaces for the Comparison of Classifiers

In the preceding chapters, the main performance criterion for the comparison of classification rules was the error probability and the corresponding appropriate performance measure – the error rate on a validation set. In this chapter, we will motivate, why one might want to go beyond that, and we will introduce standardized partition spaces as means to do so.

5.1 Introduction

In the days of data mining, the number of competing classification techniques is growing steadily. Thus, it is a worthy goal to rate the goodness of classification rules from a wide range of techniques related to diverse theoretical backgrounds. Restricting the term *goodness* to what can be most easily formalized and measured is unsatisfactory. That is, 'goodness' of a classification rule can stand for much more than only its ability to assign future objects

correctly to classes — the *correctness probability*. This is, however, the only aspect that is controlled by the most common performance measure, the error rate. Error rates do not cover the variety of demands on classification rules in practice.

In this context, Hand (1997) attaches importance to goodness concepts regarding the rule's quantitative assessment of the membership of objects in classes. This assessment typically determines the final assignment into classes: a high assessment (relative to the assessed membership in other classes) in the assigned class should be justified (*accuracy*), the relative sizes of membership in classes should reflect 'true' conditional class probabilities (*precision*), and membership values of objects in the different classes should be well-separated (*non-resemblance*).

Beyond the reliable quantitative assessment of the membership of new objects in classes, in many practical applications of classification techniques it is important that this assessment can be easily understood and is comprehensible. For that purpose one often is interested in the range of values of predictors assigned to the same class. This relates to the rule's induced partitions of the predictor space (cp. (3.3.6)):

$$\begin{aligned} \mathbf{X} &= \bigcup_{c \in \mathbf{C}} \mathbf{X}^{[c]}, \\ & \mathbf{X}^{[c]} \cap \mathbf{X}^{[\tilde{c}]} = \emptyset, c \neq \tilde{c}, c, \tilde{c} \in \mathbf{C}. \end{aligned}$$

It is not always the original space of measurement $\mathbf{X}$ in which considerations about decision regions or boundaries are performed. In multi-dimensional spaces $\mathbf{X}^K, K \in \mathbb{N}$ or/and with very flexible form of boundaries this often does not lead to an improvement of understanding. In these cases, boundaries and

regions are often described in some feature space $\mathbf{F}$ of suitably derived features $\Phi : \mathbf{X} \rightarrow \mathbf{F}$:

$$\begin{aligned} \mathbf{F} &= \bigcup_{c \in \mathbf{C}} \mathbf{F}^{[c]}, \\ & \mathbf{F}^{[c]} \cap \mathbf{F}^{[\tilde{c}]} = \emptyset, c \neq \tilde{c}, c, \tilde{c} \in \mathbf{C}. \end{aligned}$$

One example is the support vector machine. As has been shown in Chapter 3 the joint faces of the partitions of support vector machines in feature space are hyperplanes, and their form in the original space is typically "flexible" and beyond human imagination if the dimension of the original space is greater than three. Other examples of partitions described in feature spaces are common in the visualization of decision regions of linear and quadratic discriminant classifiers. For a more detailed discussion of this point, see Sondhauss and Weihs (2001a).

If decision boundaries are described in some feature space, a direct comparison of the partitions from different classifiers would only be possible, if all classification methods would deduce at least resembling entities as features. This is not true when comparing methods from machine learning and statistics.

Therefore, we will standardize the space of induced partitions with respect to the joint faces between the partitions, such that in the corresponding "feature" space we can compare and visualize the basic pattern of rules, and additionally we can measure performance with respect to goodness aspects beyond the correctness probability. This idea was first introduced in Weihs and Sondhauss (2000)

5.2 The Standardized Partition Space

The goal is to make argmax rules and their membership values comparable to the 'true' or 'optimal' Bayes rule $\hat{c}(\circ|\texttt{opt})$ and its membership values – the true conditional class probabilities. The space of these optimal features (cp. (4.3.1)) is the G-dimensional unit simplex $\mathbf{U}^G$. In future, this will also be the space for standardized partitions.

For up to four classes, the partition induced by Bayes rules can be visualized in the unit simplex, in a diagram also known as a barycentric coordinate system shown in Figure 5.1. These types of diagrams are well known in experimental design to represent mixtures of components, and are used e.g. by Anderson (1958) to display regions of risk for Bayes classification procedures.

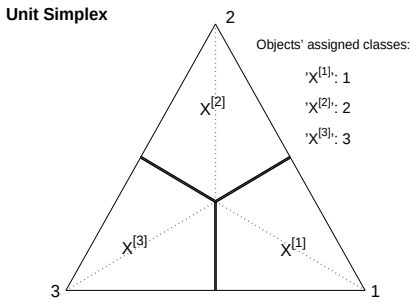

Figure 5.1: Unit simplex representing the partition of any Bayes rule in the space of conditional class probabilities in three classes: $\mathbf{U}^3$. Solid borders in separate regions of objects that get assigned to the same class. Dashed borders within these regions separate objects that differ in the class in which they have second highest class probability.

Now, as you can see in Figure 5.1, in this space induced partitions of any Bayes rule look just the same – all Bayes rules result in the same decision regions and boundaries! Rules differ now no longer in the image of their induced

partition, but where they place objects within these regions. Therefore, to visualize the behavior of learnt rules in these coordinates, we have to display some data points in these coordinates.

We use a *test set* $\mathbf{T}$ for that purpose. We require that the examples in the test set data were not utilized by any method under consideration for the learning of their rule, not as learning set $\mathbf{L}$, and not as validation set $\mathbf{V}$ to fight overfitting. Like in the Definition 2.3.1 of the training set we define the test set such that it allows all statements $c_n \in \mathbf{T}$, $\vec{x}_n \in \mathbf{T}$, and $(c_n, \vec{x}_n) \in \mathbf{T}$ to be valid:

$$\begin{aligned} \mathbf{T} &:= \{(c,\vec{x})(\omega_n), \omega_n \in \mathbf{\Omega}, n = 1, ..., N_{\mathbf{T}}\}, \\ &:= \{(c_n, \vec{x}_n), n = 1, ..., N_{\mathbf{T}}\}, \\ &:= \{c_n, \vec{x}_n, n = 1, ..., N_{\mathbf{T}}\}. \end{aligned}$$

5.3 The Bernoulli Experiment and the Correctness Probability

The basic view to analyze the behavior of some classification rule on the test set is that of a Bernoulli experiment. Remember in Definition 3.4.5 we defined the random variable

$$Z(C, \vec{X}|\mathbf{L}, \texttt{met}) := L^{[01]}\left(C, \hat{c}(\vec{X}|\mathbf{L}, \texttt{met})\right)$$

that represents the zero-one loss of some learnt matching rule $\hat{c}(\circ|\mathbf{L}, \texttt{met})$ on some new randomly drawn example $(C, \vec{X})$ according to $P_{C,\vec{X}}$. In standardized partition spaces, we define performance in terms of success rather than in terms

of loss therefore the random variable of interest is the counterpart — the zero-one gain:

$$\begin{aligned} G(C, \vec{X}|\mathbf{L}, \texttt{met}) &:= 1 - L^{[01]}(C, \hat{c}(\vec{X}|\mathbf{L}, \texttt{met}), \\ &= \mathbb{I}_C\left(\hat{c}(\vec{X}|\mathbf{L}, \texttt{met})\right). \end{aligned}$$

Given the i.i.d. assumption holds on the test set, the gains with respect to the examples in the test set are appropriately modelled as realizations of Bernoulli random variables

$$G_n \sim \mathcal{B}e\Big(\mathbf{CP}\left(\hat{c}(\vec{X}|\mathbf{L}, \texttt{met}), P_{C,\vec{X}}\right)\Big),\ n = 1, ..., N_{\mathbf{T}},$$

with

$$\mathbf{CP}\left(\hat{c}(\vec{X}|\mathbf{L}, \texttt{met}), P_{C,\vec{X}}\right) := P_{C,\vec{X}}\left(\left\{(C, \vec{X}) : \hat{c}(\vec{X}|\mathbf{L}, \texttt{met}) = C\right\}\right)$$

the correctness probability. And – analogously to the error probability and the error rate – the correctness rate on the examples of the test set is an appropriate empirical measure for the correctness probability:

$$\mathbf{CR} := \tfrac{1}{N_{\mathbf{T}}} \textstyle\sum_{(c_n, \vec{x_n} \in \mathbf{T}} \mathbb{I}_{c_n}(\hat{c}(x_n|\mathbf{L}, \texttt{met})$$

We are not so much interested in the overall correctness probability, but more in the correctness probability with respect to the assignment into each of the classes. Therefore, we divide the overall Bernoulli experiment for the analysis of the gain with respect to any assignment in G Bernoulli experiments for the analysis of potential gain with respect to the assignment into each class.

Let $\mathbf{X}^{[c]}(\mathbf{L}, \texttt{met}), c{\in}\mathbf{C}$ denote the induced partition on the predictor space according to the rule $\hat{c}(\vec{x}|\mathbf{L}, \texttt{met})$, $\vec{x}{\in}\mathbf{X}$.

The conditional random variable $G^{[c]} : \mathbf{C}\times\mathbf{X}^{[c]}(\mathbf{L}, \texttt{met}) \rightarrow \{0, 1\}$:

$$
\begin{aligned}
& G^{[c]}(C, \vec{X}|\vec{X}{\in}\mathbf{X}^{[c]}(\mathbf{L}, \texttt{met}), \mathbf{L}, \texttt{met}) \\
:= & \begin{cases} \mathbb{I}_c(C)\mathbb{I}_c\left(\hat{c}(\vec{X}|\mathbf{L}, \texttt{met})\right) & \text{if } \vec{X}{\in}\mathbf{X}^{[c]}(\mathbf{L}, \texttt{met}) \\ \text{undefined} & \text{otherwise,} \end{cases} \\
= & \begin{cases} \mathbb{I}_c(C) & \text{if } \vec{X}{\in}\mathbf{X}^{[c]}(\mathbf{L}, \texttt{met}), \\ \text{undefined} & \text{otherwise.} \end{cases}
\end{aligned}
$$

is for each $c{\in}\mathbf{C}$ a Bernoulli random variable. Therefore, the gains of test set examples in these regions can be modelled as realization of i.i.d. Bernoulli random variables

$$
G_n \sim \mathcal{B}e\Big(\mathbf{CP}\left(\hat{c}(\vec{X}|\mathbf{L}, \texttt{met}), P_{C,\vec{X}}\right)\Big),\ n = 1, ..., N_{\mathbf{T}},
$$

with

$$
\begin{aligned}
& \mathbf{CP}^{[c]}\left(\hat{c}(\vec{X}|\mathbf{L}, \texttt{met}), P_{C,\vec{X}}\right) \\
= & \ P_{C,\vec{X}}\left(\left\{(C, \vec{X}) :\ C = c,\ \hat{c}(\vec{X}|\mathbf{L}, \texttt{met}) = c\right\}\right), \\
=: & \ P_{C,\vec{X}|\mathbf{X}^{[c]}}\left(\left\{(C, \vec{X}) : C = c,\ \vec{X}{\in}\mathbf{X}^{[c]}\right\}\right),
\end{aligned}
$$

the conditional correctness probability. And the corresponding empirical correctness rates will be called *local* correctness rates.

$$
\mathbf{CR}^{[c]} \ := \frac{1}{N_{\mathbf{T}^{[c]}}} \sum_{c_n \in \mathbf{T}^{[c]}} \mathbb{I}_c(c_n)
$$

5.4 Goodness Beyond Correctness Probability

We are focussing on the goodness concepts of *accuracy, precision*, and *ability to separate* on refer to the concepts of *inaccuracy, imprecision*, and *resemblance* of Hand (1997). There are two main differences. First, we use counterparts, i.e. high values and not low values are desirable. Second, Hand restricts his attention to probabilistic classifiers where membership values are equal to estimated conditional class probabilities. Our aim is to generalize these concepts to be applicable for a wider range of techniques, by using scaled membership values instead of estimated conditional class probabilities.

Accuracy tells us something about the effectiveness of the rule in the assignment of objects into classes. Measures of accuracy in the literature typically assess whether true classes are the same as assigned classes. We call these measures *correctness* measures to distinguish them from Hands measures of accuracy that quantify the difference between a-posteriori class probabilities of an observed example (1-0) and the estimated conditional class probabilities of a probabilistic classifier. Given an accurate rule in the sense of Hand allows to interpret the size of the estimated probability in the assigned class as a reliable measure of the certainty we can have about that assignment. We want scaled membership values to potentially allow for the same interpretation.

Ability to separate tells us, how well classes are distinguished, given the transformations the classification rule uses to assess the membership of an object in the classes. Measures of the ability to separate are based on the diversity of the vectors of 'true' conditional class probabilities among objects assigned to different classes given their membership values. This is slightly

different from Hand's concept of non-resemblance, where the diversity of the 'true' conditional class probabilities among classes within probability vectors given membership values is of interest. If the ability to separate is high, the classifiers transformation for gaining membership vectors work out the characteristic differences between classes. We want to use scaled membership vectors to appropriately assess a classifiers ability to separate.

Precision tells us, how good the classification rule estimates 'true' conditional class probabilities. Measures compare the rule's membership values with 'true' conditional class probabilities. To measure precision, we obviously need knowledge of 'true' conditional class probabilities. Since our scaled membership values should reflect as precisely as possible the information in the original membership values and the rule's performance on the test set, the empirical precision will be enforced by our scaling procedure. Thus, scaled membership values can not be used to assess precision, but mirror information of the rule's performance on the test set.

Note that Hand (1997) also defines *separability* which is substantially different from the concepts above as it is a characteristic of classification problems and not of rules. It tells us, how different the 'true' conditional class probabilities of objects are, given the observed features. Separability determines an upper bound for any rule's ability to separate. A measure for separability is based on the diversity of 'true' conditional class probabilities given the values of the predictors of objects.

5.5 Some Example

For illustration, we generated two small data sets (27 examples each). Examples are generated according to three $\chi^2(\nu)$-distributions with parameters $\nu = 2$, $\nu = 9$, and $\nu = 29$, respectively. Figure 5.2 shows the dot plot of the realizations $\vec{x}_n \in \mathbf{T}$ from these classes, $c_n \in \{1, 2, 3\}$, with $1 : \chi^2(2)$, $2 : \chi^2(9)$, and $3 : \chi^2(29)$.

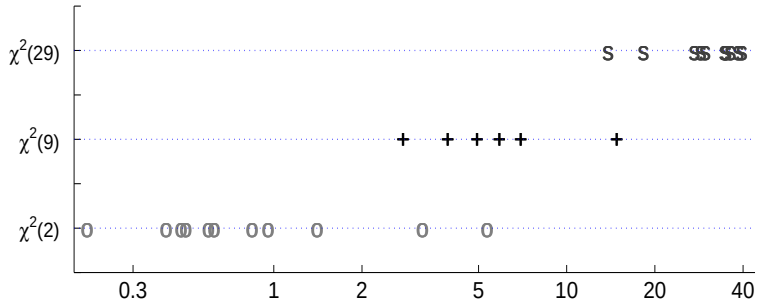

Figure 5.2: Dotplot on a logarithmic scale for the realized test set samples from the three χ^2 distributions. There is small overlap between the samples such that they are non-separable.

The simplex on the left in Figure 5.3 presents the vectors of true conditional class probabilities $\vec{p}(\vec{x}_n|\tau)$ of the realizations $\vec{x}_n \in \mathbf{T}$. On the right you see the respective estimated conditional class probability vectors $\vec{p}(\vec{x}_n|\mathbf{L}, \texttt{qda})$, $\vec{x}_n \in \mathbf{T}$ according to the learnt plug-in Bayes rule of the QDA classifier as presented in Chapter 2. Details of the implementation are given in the appendix.

The closer the marker of an example is to the class corner the higher is its (estimated) probability in that class. The general shape of the position of the markers in the two simplexes is parabolic. You can see e.g., that the QDA classifier underestimates the class probability for objects in class $\chi^2(2)$,

because the corresponding markers are farer away from this corner than in the simplex for the true Bayes classifier. On the other hand, it overestimates the class probability for objects from the $\chi^2(29)$-distribution, as these markers are closer to the corresponding corner. A comparison of the correctness rates $\mathbf{CR}_c$ in the different regions corresponding to classes $c, c \in \mathbf{C}$ reveals that the QDA classifier performs worse in the assignment to any class.

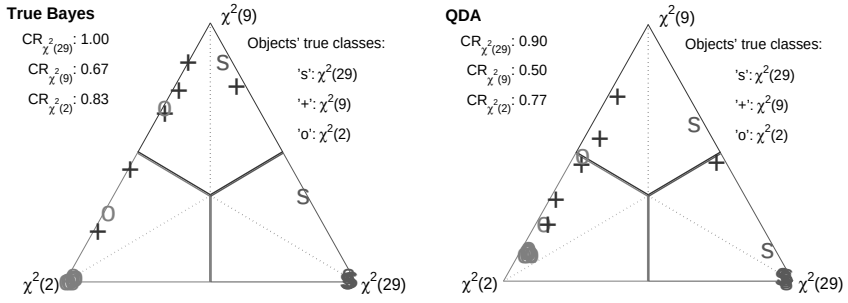

Figure 5.3: Simplexes representing the membership vectors of the test set observations for the true Bayes and the QDA classifier. $\mathbf{CR}_{\chi^2(2)}$, $\mathbf{CR}_{\chi^2(9)}$, and $\mathbf{CR}_{\chi^2(29)}$ denote the correctness rates for the assignments to the corresponding classes.

For obtaining comparable partitions from arbitrary argmax rules it is not appropriate to simply display their membership values. One obvious reason is that they neither have to be non-negative nor add up to 1. Moreover, any ad-hoc standardization of membership values into $\mathbf{U}^G$ might lead to patterns more influenced by the standardization procedure than by the rule's classification behavior.

For the χ^2-example, the NN classifier is modelled as a feedforward two-layer network with logistic hidden layer units and linear output units, the

standard approach to non-linear function approximation. The resulting membership values on the test set are not necessarily in the interval [0,1]. A typical transformation into $\mathbf{U}^G$ is the so-called *softmax* transformation `sof` (Bridle, 1990):

$$m(c, \vec{x}|\mathbf{L}, \texttt{met}, \texttt{sof}) \quad = \quad \frac{\exp\left(m(c, \vec{x}|\mathbf{L}, \texttt{met})\right)}{\sum_{c \in \mathbf{C}} \exp\left(m(c, \vec{x}|\mathbf{L}, \texttt{met})\right)}. \tag{5.5.1}$$

Figure 5.4 presents these ad-hoc scaled membership values. Obviously this transformation leads in this example to a concentration of membership values in the barycenter of the simplex, suggesting, among other things, the NN classifier was less decisive or more uncertain about its assignments into classes than the QDA classifier. This impression, though, is misleading as the correctness rates of the NN classifier are better than those of the QDA classifier. Actually they are as good as those of the true Bayes classifier (Figure 5.3).

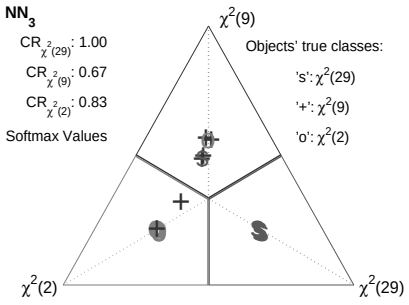

Figure 5.4: Simplex representing the membership vectors of the test set observations for the NN classifier. They were scaled to lie in $\mathbf{U}^G$ using the softmax transformation. $\mathbf{CR}^{\chi^2(2)}$, $\mathbf{CR}^{\chi^2(9)}$, and $\mathbf{CR}^{\chi^2(29)}$ denote the correctness rates for the assignments to the corresponding classes.

Having said that, maximum membership values of argmax rules based on learnt conditional class probabilities are – just as ad-hoc scaled membership values – not a reliable measure for the membership of objects in the assigned

classes, and thus not a reliable measure for the correctness of the rule's decision. They give information on the rule's performance from its own perspective only, whereas for comparisons, we would prefer a more objective view.

You can see this in Figure 5.3 in the χ^2-example . In the QDA simplex, membership values of objects that get assigned to class $\chi^2(29)$ are closer to the $\chi^2(29)$ corner than in the true Bayes simplex, because the QDA classifier gives these objects a higher membership in that class. In terms of separateness or non-resemblance, this incorrectly suggests that QDA was better in that assignment than the true Bayes classifier. Only in simulation studies like this we can know whether a higher separateness is justified by a real difference in the characteristics of objects, or rather by an overestimation of the relevance of differences by the classification method.

As both, non-probabilistic membership values and estimated class probabilities, need to be scaled to reflect the rule's true performance, from now on we do no longer distinguish between membership values of probabilistic classifiers and non-probabilistic classifiers, assuming the latter to be already 'appropriately' standardized into the space $\mathbf{U}^G$. Appropriately for our purposes means that for any $\vec{x}\in\mathbf{X}$ at least the order of the membership values of classes is kept whether it is based on the original membership values or on the standardized ones. In this sense, the softmax transformation defined above is a valid transformation. For classifiers with membership values not on a metric scale, we recommend to use the ranks of $m(\vec{x},1),...,m(\vec{x},G)$ for each $\vec{x}\in\mathbf{X}$ divided by the number of classes G, as the relative distance between membership values can not be interpreted. From now on we assume the membership

vectors $\vec{m}(\vec{x}|\mathbf{L}, \texttt{met})$, $\vec{x}\in\mathbf{X}$, to be members of the unit simplex, the placeholder `met` subsuming the learning method and – if necessary – the ad-hoc scaling method.

5.6 Scaling

As shown in the χ^2 example, membership values have to be appropriately scaled, before they can be used to assess the quality of a classification rule. Otherwise, membership values are misleading. With our scaling procedure we derive membership values where the size of the membership value of an object assigned to some class can be interpreted as an indicator for the certainty, with which the object really belongs to that class. In other words we are interested mainly in the interpretation of the membership values in the assigned class. In the following, these are called *assignment values*. We will denote them in dependence of the class of the final assignment:

$$m^{[c]}(\vec{x}|\mathbf{L}, \texttt{met}) \quad := \quad \begin{cases} \max_{g\in\mathbf{C}} m(g, \vec{x}|\mathbf{L}, \texttt{met}) & \text{if } \hat{c}(\vec{x}|\mathbf{L}, \texttt{met}) = c \\ \text{undefined} & \text{otherwise,} \end{cases} \tag{5.6.1}$$

for $c\in\mathbf{C}$. If the assignment values of one classification rule are on average higher than those of another classification rule, then for this to be justified, the assignment into classes according to the first rule should be more reliable than according to the latter. This is needed, if we want to go beyond the mere use of correctness rates for the comparison of classification methods.

The assignment value of some new example $(C, \vec{X})$ given $\vec{X}\in\mathbf{X}^{[c]}(\mathbf{L}, \texttt{met})$ is

just like its gain a conditional random variable:

$$M^{[c]}(\vec{X}|\vec{X}\in\mathbf{X}^{[c]}(\mathbf{L},\texttt{met}),\mathbf{L},\texttt{met}) := \begin{cases} m^{[c]}(\vec{X}|\mathbf{L},\texttt{met}) & \text{if } \vec{X}\in\mathbf{X}^{[c]}(\mathbf{L},\texttt{met}), \\ \text{undefined} & \text{otherwise.} \end{cases}$$

And thus, the assignment values of the test set examples in the regions can be modelled as realization of random variables from distribution $P_{M^{[c]}}$.

We have two sources of information about the certainty of the assignment of objects into classes:

1. the individual assessment of the membership of the objects in the classes according to the membership values of the classification rule,

2. the observed performance with respect to these assignments on the test set.

The former source has the disadvantage that it might be unreliable, the latter the disadvantage that it results in statements of the average certainty whereas we are interested in the certainty on individual assignments. Thus, we will use the information on the actual performance to "correct" the membership values such that on average they reflect the actual behavior and such that relative individual assessment is kept.

With these preliminaries, we can compare the information about the correctness probability as it is contained in the test set with the information about the conceived correctness probability according to the classifier's membership values.

In a Bayesian setting, the correctness probability can be approached as an uncertain parameter φ with a quantification of the current uncertainty about it according to a conjugate prior distribution P_φ. The Bernoulli distribution is a special case of the binomial distribution, and the binomial-beta model is the two-dimensional special case of the multinomial-Dirichlet model described in subsection 2.3.3.

[**Notation**] In standard notation of the Beta distribution the parameters α and β correspond to the parameterization of the Dirichlet distribution α_1, α_2 as we used it in Section 2.3.3. We will define another parameterization in terms of the *expected value* of the Beta distribution

$$p := \mathbf{E}(\gamma|\alpha,\beta) \quad = \quad \frac{\alpha}{\alpha+\beta}$$

and its equivalent sample size (cp. Section (2.3.3)) which we now refer to as *dispersion parameter*:

$$N := \alpha+\beta$$

One says N is the equivalent sample size only in context of the Binomial-Beta learning of the uncertain parameter γ. Approaching the Beta distribution in general (as we will do later) calling it "dispersion" parameter is motivated by the fact that for fixed p the parameter N determines the variance of the Beta distribution:

$$\begin{aligned}\mathbf{Var}(\gamma|\alpha,\beta) &= \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \\ &= \frac{p(1-p)}{N+1}\end{aligned}$$

We can quantify the uncertainty about the correctness probability in each region $\mathbf{X}^{[c]}$ with the information in the sample in the test set for the corresponding conditional Bernoulli experiment:

$$\mathbf{T}^{[c]}(\mathbf{L},\texttt{met}) := \{(c_n,\vec{x}_n)\in\mathbf{T} : \hat{c}(\vec{x}_n|\mathbf{L},\texttt{met}) = c\}$$

Let $N_{\mathbf{T}^{[c]}}$ be the potency of $\mathbf{T}^{[c]}$.

In standardized partition spaces and for the assessment of the information about the classifiers correctness probability in the test set we use the minimal informative improper prior $(\alpha^0, \beta^0) := (0, 0)$ rather than the other possible choice $(\alpha^0, \beta^0) := (1, 1)$ which we said we would prefer in general in Section 2.3.3. We say something about the reasons to do so later.

With this prior, the posterior distributions $P_{\varphi^{[c]}|\mathbf{T}^{[c]}}$ quantifying the uncertainty about the correctness probabilities $\varphi^{[c]}$ for the assignment into classes $c \in \mathbf{C}$ are Beta distributions $\mathcal{B}t(p_{\mathbf{T}^{[c]}}, N_{\mathbf{T}^{[c]}})$ where the expected value $p_{\mathbf{T}^{[c]}}$ is quantified by the local correctness rate $\mathbf{CR}^{[c]}$ in class c:

$$\begin{aligned} p_{\mathbf{T}^{[c]}} &:= \frac{1}{N_{\mathbf{T}^{[c]}}} \sum_{c_n \in \mathbf{T}^{[c]}} \mathbb{I}_c(c_n) \\ &= \mathbf{CR}^{[c]} \end{aligned}$$

and the dispersion parameter $N_{\mathbf{T}^{[c]}}$ is quantified by the number of examples in class $c \in \mathbf{C}$.

We will model the distribution of the assignment value $P_{M^{[c]}}$ also by a Beta distribution, $M^{[c]} \sim \mathcal{B}t(p_{M^{[c]}}, N_{M^{[c]}})$ with unknown parameters $p_{M^{[c]}} \in [0, 1]$ and $N_{M^{[c]}} \in \mathbb{N}$. We estimate these parameters by standard point estimation for that task, namely the method of moments (cp. Gelman et al., 1995). Let

$$\begin{aligned} \overline{m}^{[c]} &:= \frac{1}{N_{\mathbf{T}^{[c]}}} \sum_{\vec{x_n} \in \mathbf{T}^{[c]}} m^{[c]}(\vec{x}|\mathbf{L}, \texttt{met}) \\ \mathbf{S}^{[c]} &:= \frac{1}{N_{\mathbf{T}^{[c]}} - 1} \sum_{\vec{x}_n \in \mathbf{T}^{[c]}} \left(m^{[c]}(\vec{x}_n|\mathbf{L}, \texttt{met}) - \overline{m}^{[c]}\right)^2 \end{aligned}$$

be the sample mean and the sample variance of the assignment values for class

c. Then the parameters $p_{M^{[c]}}$ and $N_{M^{[c]}}$ can be estimated by:

$$\begin{aligned} p_{m^{[c]}} &:= \overline{m}^{[c]}, \\ N_{m^{[c]}} &:= \frac{\overline{m}^{[c]}\left(1-\overline{m}^{[c]}\right)}{\mathbf{S}^{[c]}} - 1. \end{aligned}$$

If the classifier's assignment values were a reliable reflection of the uncertainty of their assignments, then the estimated expected value should reflect the local correctness rate $\mathbf{CR}^{[c]}$ of the assignment to class c, as both are estimators of the local correctness probability $\gamma^{[c]}$. That is,

$$p_{m^{[c]}} \overset{!}{\approx} \mathbf{CR}^{[c]}.$$

If that is not the case, membership values need to be scaled to be trusted as quantifications to assess the certainty of assignments.

It is not necessary, though, that the two Beta distribution $\mathcal{B}t(p_{m^{[c]}}, N_{m^{[c]}})$ and $\mathcal{B}t\left(p_{T^{[c]}}, N_{T^{[c]}}\right)$ resemble also with respect to their dispersion parameters, because these have different interpretations:

$\mathcal{B}t\left(p_{T^{[c]}}, N_{T^{[c]}}\right)$ describes the uncertainty about the true value of the correctness probability $\gamma^{[c]}$. The entity of interest is $\gamma^{[c]}$. With respect to the membership values the Beta distribution $\mathcal{B}t(p_{m^{[c]}}, N_{m^{[c]}})$ describes the uncertainty of individual assignments to classes from the perspective of the classifier. The entities of interest are the individual assessments.

In the former context, the larger $N_{\mathbf{T}^{[c]}}$ is, the better, because the more certain we are about γ. In the latter, smaller $N_{m^{[c]}}$ is preferable, if one can reliably interpret the deviation from the mean in two respects:

1. if the assignment value is larger/lower than the mean then the assignment is more/less certain than on average, and

2. if the assignment value is lower than the mean and this results in a higher assessment of membership in another class then this should correspond to a higher probability that this object belongs to that other class than on average.

If membership values are reliable, then the dispersion parameter $N_{m^{[c]}}$ reflects the (true) diversity of objects with respect to assessed membership.

Our scaling aims at scaled membership values that meet these requirements. That is, we want to scale membership values such that they approximate a Beta distribution with expected value $\mathbf{CR}^{[c]}$ and "appropriate" dispersion parameter N. We iterate to find the best dispersion parameter $N^{[c,opt]}$:

1. Define $\underline{N} := \min\{N_{\mathbf{T}^{[c]}}, N_{M^{[c]}}\}$ as the minimum dispersion parameter and $\overline{N} := \max\{N_{\mathbf{T}^{[c]}}, N_{M^{[c]}}\}$ as the maximum dispersion parameter to describe the diversity of assessed scaled membership in class c among all objects that get assigned to class c. Do the following steps for all $N \in \mathbb{N}$, $\underline{N} \leq N \leq \overline{N}$:

   (a) Let $F_{p,N}$ denote the Beta distribution function with parameters p, N. For all $\vec{x}_n \in \mathbf{T}^{[c]}$, evaluate the equation

   $$F_{\mathbf{CR}^{[c]},N}\left(m(c,\vec{x}_n|\mathbf{L},\texttt{met},\mathbf{T},N)\right) \tag{5.6.2}$$

   $$= \quad F_{p_{m^{[c]}},N_{m^{[c]}}}\left(m(c,\vec{x}_n|\mathbf{L},\texttt{met})\right) \tag{5.6.3}$$

   to yield $m(c,\vec{x}_n|\mathbf{L},\texttt{met},\mathbf{T},N)$. Scaled assignment values that solve this equation approximate a Beta distribution $\mathcal{B}t\Big(\mathbf{CR}^{[c]},N\Big)$. This is a basic trick mainly used when generating random numbers. The trick is based on a fundamental property of any distribution P_Y with

continuous distribution function F_Y. According to this property the transformation of Y with its own distribution function is uniformly distributed

$$U := F(Y) \Rightarrow U \sim \mathcal{U}[0,1],$$

with $\mathcal{U}[0,1]$ denoting the uniform distribution on the interval $[0,1]$. And for an uniformly distributed random variable $U \sim \mathcal{U}[0,1]$ and any continuous distribution function $\tilde{F}$ of a distribution $\tilde{P}$ it is true that

$$\tilde{F}^{-1}(U) \sim \tilde{P}.$$

In consequence it is true that

$$\tilde{F}(F^{-1}(Y)) \sim \tilde{P}.$$

(b) Scaled membership vectors are members of the unit simplex

$$\vec{m}(\vec{x}|\mathbf{L}, \mathtt{met}, \mathbf{T}, N) \in \mathbf{U}^G.$$

Therefore, after the scaling of the assignment value of some object $(c_n, \vec{x}_n) \in \mathbf{T}^{[c]}$ the membership values $m(g, \vec{x}_n|\mathbf{L}, \mathtt{met})$ of this object in the other classes $g \neq c,\ g, c \in \mathbf{C}$ have to be recalculated, such that the scaled membership values in all classes sum up to one. We do this, keeping the ratio of membership values in the other classes fixed:

$$\begin{aligned} & m(g, \vec{x}_n|\mathbf{L}, \mathtt{met}, \mathbf{T}, N) \\ := \quad & \frac{1 - m(c, \vec{x}|\mathbf{L}, \mathtt{met}, \mathbf{T}, N)}{1 - m(c, \vec{x}|\mathbf{L}, \mathtt{met})} m(g, \vec{x}|\mathbf{L}, \mathtt{met}). \end{aligned}$$

(c) We calculate the average euclidian distance of the scaled membership vector $\vec{m}(\vec{x}_n|\mathbf{L}, \mathtt{met}, \mathbf{T}, N)$ to the class indicator function of the true class $\vec{e}(c_n)$, $(c_n, \vec{x}_n) \in \mathbf{T}^{[c]}$:

$$D(N) := \frac{1}{N_{\mathbf{T}^{[c]}}} \sum_{(c_n, x_n) \in \mathbf{T}^{[c]}} \left\| \vec{e}(c_n) - \vec{m}(\vec{x}|\mathbf{L}, \mathtt{met}, \mathbf{T}, N) \right\|$$

The smaller it is, the more reliable is the information on deviations in dependency of N. We do not want to increase the number of false assignments with our scaling, therefore we select as best $N^{[c,opt]}$ the one that minimizes $D(N)$ relative to the prediction error rate on $\mathbf{T}^{[c]}$ we would get, if we used $m(\circ|\mathbf{L}, \mathtt{met}, \mathbf{T}, N)$ for prediction.

The scaled membership vectors with the optimal dispersion parameters in each class are the scaled membership values we use in standardized partition spaces:

$$\vec{m}(\vec{x}|\mathbf{L}, \mathtt{met}, \mathbf{T}, \mathtt{sca}) \in \mathbf{U}^G.$$

5.6.1 Justification

The scaled membership vectors $\vec{m}(\vec{x}|\mathbf{L}, \mathtt{met}, \mathbf{T}, \mathtt{sca})$ describe the classification performance of the rule, because on average the scaled memberships in the assigned classes reflect the correctness of that decision, namely $\mathbf{CR}^{[c]}$ and the dispersion parameter $N^{[c,opt]}$ is chosen such that the reliable information of the scattering of vectors around the mean is maximized, $c \in \mathbf{C}$.

Given these properties, we can interpret the scaled membership values $m(s, \vec{x}|\mathbf{L}, \mathtt{met}, \mathbf{V}, \mathtt{sca})$ as some estimator for the probability to be in a certain

state s given the vector of membership values $\vec{m}(\vec{x}|\mathbf{L}, \texttt{met})$.

$$p\left(s|\vec{m}(\vec{x}|\mathbf{L}, \texttt{met}), \mathbf{T}, \texttt{sca}\right) \quad \leftrightarrow \quad p\left(s \mid \vec{m}(\vec{x}|\mathbf{L}, \texttt{met}), \tau\right)$$

Their mean value on $\mathbf{X}^{[c]}$ is determined by the performance on the test set, the inner ranking of the membership vectors $\vec{m}(\vec{x}|\mathbf{L}, \texttt{met}, \mathbf{T}, \texttt{sca})$ with $\vec{x}\in\mathbf{X}^{[c]}$ is determined by the original membership vectors, and the dispersion by the reliability of the dispersion around the mean according to the performance on the test set.

The procedure can be understood in terms of the binomial-beta model as we introduced it, but also as some way to scale assignment values such that their ordering (within regions) is kept, and their average value reflects the local correctness rate of an assignment in this region (see Weihs and Sondhauss, 2000; Sondhauss and Weihs, 2001c). The use of the Beta distribution for the approximations in the latter interpretation is justified by its flexibility. The use of the improper prior $(\alpha^0, \beta^0) := (0, 0)$ for the parameters of the Beta distribution leads to the standard estimation of the correctness probability with the correctness rate, and thus it was the prior of choice here. Scaling should not be done if there are only few data points in a region, and thus the difference in the posterior distributions will be small.

Why the local correctness rate $\mathbf{CR}^{[c]}$ is the target mean of scaled assignment values is intuitively clear. An appropriate choice for the dispersion parameter $N^{[c,\texttt{opt}]}, c\in\mathbf{C}$, is less obvious. Using $N_{\mathbf{T}^{[c]}}$ is not appropriate, because we are not really interested in the modelling of our uncertainty with respect to the parameter $\gamma^{[c]}$ of the Bernoulli distribution. This would lead to a non-intuitive behavior of our scaling as, e.g. for $N_{\mathbf{T}^{[c]}} \to \infty$, all scaled assignment values of

objects in that region approach the empirical mean $\mathbf{CR}^{[c]}$ which approaches the true correctness probability. The parameter $N^{[c,\mathtt{opt}]}$ should rather have an interpretation as a measure of the dispersion of assignment values. Because of that, no huge scaling of assignment values near to true conditional class probabilities should take place. This is an argument favoring the dispersion parameter $N_{m^{[c]}}$ estimated with the original assignment values, but only for probabilistic classifiers. For non-probabilistic classifiers $N_{m^{[c]}}$ is dependent on the ad-hoc standardization of the corresponding membership vector into $\mathbf{U}^G$, and might be misleadingly high or low. Our approach avoids unwarranted large certainty parameters $N_{m^{[c]}}$ for each $c \in \mathbf{C}$.

5.7 Performance Measures for Accuracy and Ability to Separate

We now define measures for the rating of the performance of argmax classification rules with respect to accuracy and ability to separate. These measures are based on Euclidean distances between scaled membership vectors of test set examples and specified corresponding corners of the simplex: either it is the corner of the true class or it is the corner of the assigned class.

The definition in terms of these Euclidean distances is not only useful for the understanding of the measures as such but also for a visualization of the performance of classifiers for classification problems with up to four classes.

The measure of accuracy is based on the Euclidean distances between scaled

membership vectors $\vec{m}(\vec{x}_n|\mathbf{L}, \texttt{met}, \mathbf{T}, \texttt{sca})$ and the vector representing the corresponding *true* class corner $\vec{e}(c_n)$ for the examples $(c_n, \vec{x}_n) \in \mathbf{T}$. We standardize the mean of these distances such that a measure of 1 is achieved if all vectors lie in the correct corners, and zero if they all lie in the centroid of the simplex. The measure of accuracy is thus:

$$\mathbf{Ac}\left(\texttt{met} \mid \mathbf{L}, \mathbf{T}, \texttt{sca}\right) \tag{5.7.1}$$

$$:= \quad 1 - \frac{G}{(G-1)} \frac{1}{N_{\mathbf{T}}} \sum_{n=1}^{N_{\mathbf{T}}} \|\vec{e}(c_n) - \vec{m}(\vec{x}_n|\mathbf{L}, \texttt{met}, \mathbf{T}, \texttt{sca})\| \tag{5.7.2}$$

On the basis of this performance measure, we can now compare the behavior of the QDA classifier with the behavior of the NN classifier in the χ^2-example. In the scaled simplexes in Figure 5.5, the average distance of objects to the assigned corners can be interpreted as the average correctness of such an assignment. The nearer objects are to the corner, the better the classifier is. So we easily can see that the NN classifier is better in the assignment to the $\chi^2(2)$ and the $\chi^2(29)$ class, because apparently most objects in these regions are closer to these corners. The accuracy, though, is not that much better. The reason is, that for some individual objects, the NN classifier has very high assignment values, though the assignment is wrong. The measure of accuracy penalizes this over-confidence.

Analogously, the measure of the ability to separate is based on the Euclidean distances between scaled membership vectors $\vec{m}(\vec{x}_n|\mathbf{L}, \texttt{met}, \mathbf{T}, \texttt{sca})$ and the vector representing the corresponding *assigned* class corner $\vec{e}(\hat{c}\,(\vec{x}_n \mid \mathbf{L}, \texttt{met}))$, of the observed measurements $\vec{x}_n \in \mathbf{T}$.

Note that in particular for poor classifiers the assignment of an observation based on its scaled membership values might be different from the assignment based on the original membership values, such that

$$\begin{aligned} \hat{c}\,(\vec{x}_n \mid \mathbf{L}, \texttt{met}) &= \arg\max_{c \in \mathbf{C}} m(c, \vec{x}_n | \mathbf{L}, \texttt{met}) \\ &\neq \arg\max_{c \in \mathbf{C}} m(c, \vec{x}_n | \mathbf{L}, \texttt{met}, \mathbf{T}, \texttt{sca}) \end{aligned}$$

and that we really want to use the original assignment in our definition.

We do this, because we want to measure the diversity of the scaled membership values of objects that are assigned to the same class of the original classification rule. Only then, our measures are estimates for the behavior of the rule on its induced partitions $\mathbf{X}^c(\mathbf{L}, \texttt{met})$, $c \in \mathbf{C}$.

We standardize the mean of these distances in the same way as above, such that our measure of the ability to separate is defined as:

$$\mathbf{AS}\,(\texttt{met} \mid \mathbf{L}, \mathbf{T}, \texttt{sca}) \tag{5.7.3}$$

$$:= \quad 1 - \frac{G}{(G-1)} \frac{1}{N_{\mathbf{T}}} \sum_{n=1}^{N_{\mathbf{T}}} \|\vec{e}(\hat{c}(\vec{x}_n | \mathbf{L}, \texttt{met})) - \vec{m}(\vec{x} | \mathbf{L}, \texttt{met}, \mathbf{T}, \texttt{sca})\| \tag{5.7.4}$$

In the χ^2-example, the ability to separate of the NN classifier ($\mathbf{AS} = 0.78$) is noticeable higher than the ability to separate of the QDA classifier($\mathbf{AS} = 0.63$) as can be seen in Figure 5.6. The position of the membership values in their simplexes reflects that fact: in particular the objects in the $\chi^2(2)$ and the $\chi^2(29)$ areas are in the NN-simplex much closer to the corner than in the QDA-simplex, and therefore better separated from the objects in the other regions.

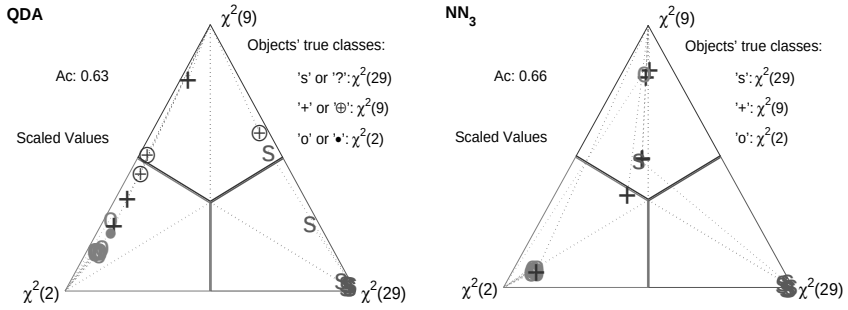

Figure 5.5: Simplexes representing the scaled membership vectors of the test set observations for the QDA and the NN classifier. For the QDA classifier, scaling moves some examples out of their original assignment areas, these are marked using an alternative symbol for their true class. One is moved from the $\chi^2(29)$ area into the $\chi^2(9)$ area, one from $\chi^2(2)$ into $\chi^2(9)$ (symbols: '$\oplus$'), and a third from $\chi^2(9)$ into $\chi^2(2)$ (symbol: '$\bullet$'). In these cases, the examples were moved into the areas of their true classes. But it can also happen that an example is moved out of the correct area. This happened to one $\chi^2(9)$ example that was moved into the $\chi^2(2)$ area (symbol: '$\oplus$'). **Ac** gives the estimated value of accuracy. Lines are drawn to illustrate the Euclidean distances that determine the **Ac** values.

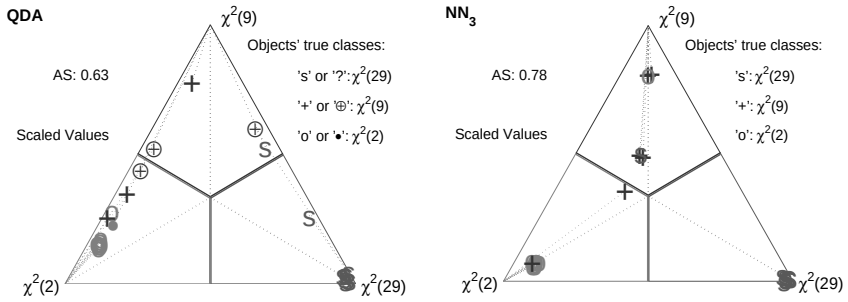

Figure 5.6: Simplexes representing the scaled membership vectors of the test set observations for the QDA and the NN classifier. For the QDA classifier, scaling moves some objects out of their original assignment areas. These are marked using an alternative symbol for their true class. **AS** gives the estimated value of the ability to separate. Dotted lines are drawn to illustrate the Euclidean distances that determine the **AS** values.

5.8 Applications

We use two benchmark problems from the UCI Repository (Blake and Merz, 1998) to show the potential benefit of standardized partition spaces for the comparison of classification methods.

One is the well-known Iris data set which dates back to Fisher (1936). The data set contains three classes of iris plant of 50 instances each, namely Setosa, Virginica and Versicolor. Using length and width of the sepals and petals of the plants, Setosa is linearly separable from the other two; Virginica and Versicolor are not linearly separable from each other. Altogether, the domain is very simple.

The other data set is the so-called Glass identification database created by German (1987). For 214 glass splinter, the origin from one of seven types of glass are to be identified. Refraction index and the weight percent in nine oxides of elements like Sodium, Magnesium and others are measured. We

comprised the original seven classes to float processed window glass (fpW), non-float processed window glass (nfW), and non-window glass (noW). This domain is pretty difficult and classes are not easy to separate.

We split the Iris data in two halves for training and testing. As the iris data set is balanced, we kept the balance in the test and training set, such that there are 25 observations from each plant in each data set. From the glass data we took a simple sample of about a third (72/214) of the observations for the test set. This resulted in a frequency in the classes fpW/nfW/NoW of 51/55/36 in the training set and 36/21/15 in the test set.

We compared a set of classification methods, that are quite different in the assumptions they make about any underlying generating process of the data as we elaborated in chapters 2-4: the classifiers from linear and quadratic discriminant analysis (LDA, QDA), a continuous naïve Bayes classifier (CNB), a Neural Network (NN), the k-Nearest neighbor classifier (k-NN), and a decision tree classifier (TREE). Details of the implementation are given in the appendix.

The main results on the iris data are presented in Table 5.1:

As expected, all classifiers perform pretty well on the Iris data set w.r.t. correctness rate, accuracy, and ability to separate. In Figure 5.7 you see the simplexes of the three best classifiers according to these measures. In the simplexes you see at a glance that these classifiers can perfectly identify Setosa plants: all scaled membership values of Setosa-objects lie strictly (except for jittering) in the Setosa-corner, and all other objects lie on the side line between Virginica and Versicolor. QDA misidentifies one Viginica plant as Versicolor,

Method	CR	Ac	AS
QDA	0.97	0.93	0.95
LDA	0.96	0.91	0.93
k-NN_3	0.96	0.89	0.94
NB	0.95	0.87	0.91
$TREE_3$	0.95	0.84	0.92
NN_4	0.92	0.77	0.88

Table 5.1: Performance of compared methods on the Iris database, ordered by declining correctness rate and accuracy. QDA, LDA, and k-NN_3 are the three best with respect to all performance measures, but the ranking differs when the ability to separate instead of the accuracy is used for ordering: LDA is second best in accuracy and k-NN is second best in ability to separate.

and one Versicolor plant as Virginica, apparent from one '+' in Versicolor's area and one 'o' in Virginica's area. The scaled assignment values lie comparatively near to the border between these areas, say, the QDA classifier detected these assignments correctly as slightly ambiguous.

LDA and k-NN_3 both have the same correctness rate, resulting from three misclassifications: LDA misidentifies two Versicolors as Virginica and one Virginica as Versicolor, and k-NN_3 does it vice versa. Objects are better separated (more densely grouped) in the simplex of k-NN_3. Thus, k-NN_3 has a higher ability to separate than LDA — simply because k-NN_3 only has a few distinct membership vector values, namely $\binom{3+3-1}{3} = 10$, corresponding to the observable frequencies in three groups in a sample of three objects. On the other side, k-NN_3 has a lower accuracy than LDA, mainly because one of the misspecified Virginicas (the 'o' near the Versicolor's corner) has in the training set three Versicolor neighbors, and thus an original membership value of one in the Versicolor class. Scaling does not move the objects in the Versicolor region too much, because on average, k-NN_3 is pretty successful in correctly

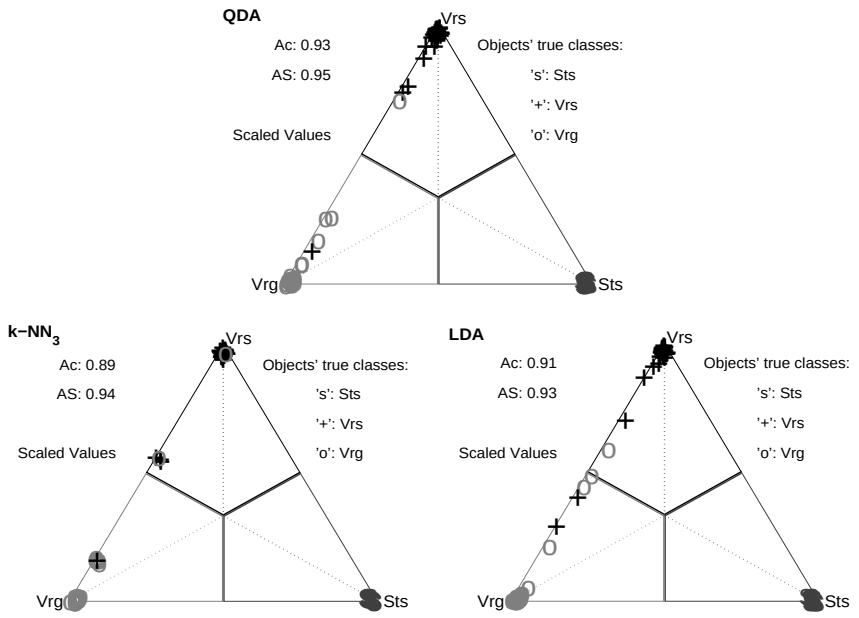

Figure 5.7: Simplexes representing the scaled membership vectors of the QDA, the k-NN_3 and the LDA classifiers of the iris test set observations. **AS** and **Ac** give the estimated ability to separate and accuracy of these classifiers.

assigning to Versicolor:

$$p\left(\text{Vrs} \mid \mathbf{X}^{[Vrs]}(\mathbf{L}, \texttt{knn}), \mathbf{T}, \texttt{sca}\right) \quad = \quad \frac{23}{25} = 0.92$$

This value of 0.92 is not too far from the average original membership value in this region,

$$\overline{m}^{[Vrs]}(\mathbf{T}|\mathbf{L}, \texttt{knn}) \quad = \quad 0.949.$$

The contribution of the misspecifying membership vector alone on the estimated accuracy **Ac** is approximately $-\frac{3}{50}\sqrt{2} \cong -0.08$.

Loosely speaking, the ability to separate is higher the lesser the classifier's membership vectors reflect the individuality of objects within regions. This is what we want, if we are mainly interested in the commonness of objects in the

same region. Yet, at the same time, it raises the risk that ambiguous individuals are not detected as such, which decreases the accuracy of the classification rule.

As expected, the overall performance is noticeably worse on the Glass data compared to the Iris data set w.r.t. correctness rate (≤ 0.85), accuracy (≤ 0.55), and ability to separate (≤ 0.72). With Figure 5.8 we mainly want to demonstrate the importance of scaling for a comparison of probabilistic and non-probabilistic classifiers. If we had measured the ability to separate of the naïve Bayes classifier with its own estimates of class conditional probabilities it would be very high, namely $\mathbf{AS}^{o}_{\mathbf{T}} = 0.96$, because most objects lie in the corners. But this would not reflect the diversity of the vectors of 'true' conditional class probabilities given their membership values, but only the classifiers own estimate thereof. And given the high rate of misclassifications, this is not reliable. For the NN_4 classifier, the softmax transformation in equation (5.5.1) leads to concentration of objects in the barycenter of the simplex like in the χ^2-example. Thus, determining the accuracy and the ability to separate with these values, would lead to very low estimates:

$$\mathbf{Ac}(\texttt{nn}|\mathbf{L}, \mathbf{T}, \texttt{sof}) = 0.16,\ \mathbf{AS}(\texttt{nn}|\mathbf{L}, \mathbf{T}, \texttt{sof}) = 0.26.$$

The arbitrariness of this may become apparent, when knowing that the softmax transformation on the doubled values of the original membership values:

$$m(c, \vec{x}|\mathbf{L}, \texttt{met}, 2\texttt{sof}) \quad := \quad \frac{\exp\left(2m(c, \vec{x}|\mathbf{L}, \texttt{met})\right)}{\sum_{c \in \mathbf{C}} \exp\left(2m(c, \vec{x}|\mathbf{L}, \texttt{met})\right)}.$$

would have led to measures of accuracy

$$\mathbf{Ac}(\texttt{nn}|\mathbf{L}, \mathbf{T}, 2\texttt{sof}) = 0.28,\ \mathbf{AS}(\texttt{nn}|\mathbf{L}, \mathbf{T}, 2\texttt{sof}) = 0.49.$$

We can continue with this: on the basis of original membership values times 4 performance would seem to be even better with

$$\mathbf{Ac}(\texttt{nn}|\mathbf{L},\mathbf{T},4\texttt{sof}) = 0.38,\ \mathbf{AS}(\texttt{nn}|\mathbf{L},\mathbf{T},4\texttt{sof}) = 0.72.$$

These ad-hoc scalings are as appropriate as the softmax transformation with factor 1 on the original membership values, thus, this is an example of the earlier stated problem, that measuring accuracy and ability to separate on the basis of ad-hoc scaled membership values reflects scaling procedure rather than the behavior of the classification rule. One can take advantage of the dependency of the softmax scaling on factorizing the original membership values to find a scaling that is appropriate and meaningful in another context. This will be shown in Chapter 7.

Method	CR	Ac	AS
TREE	0.85	0.55	0.72
k-NN_1	0.83	0.55	0.75
NN	0.67	0.32	0.50
QDA	0.67	0.31	0.34
NB	0.67	0.27	0.44
LDA	0.65	0.27	0.41

Table 5.2: Goodness of various methods on the glass database, ordered first by declining correctness rate and second by declining accuracy. TREE and k-NN_1 are by far the two best classifiers w.r.t. all three measures, but the inner ranking would be different, if we had used the ability to separate instead of the correctness rate for the ordering.

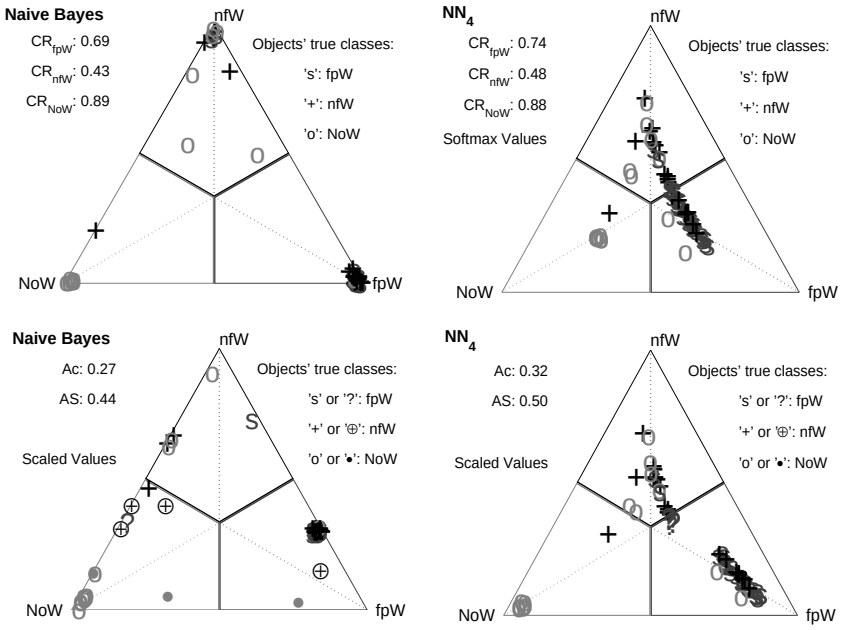

Figure 5.8: Simplexes representing the original and the scaled membership vectors of the glass test set observations for the naïve Bayes and the NN_4 classifier. **AS** and **Ac** give the estimated ability to separate and accuracy. For both classifiers, scaling moves some objects out of the assignment area of nfW. These are marked using an alternative symbol for their true class.

5.9 Conclusions

In this chapter, we introduced standardized partition spaces as means to compare argmax classification rules.

With the scaling given in section 5.6, one can overcome at the same time the arbitrariness of typical ad-hoc scaling procedures for non-probabilistic classifiers, and the unreliability of heavily model-based class conditional estimates. This is, because the empirical distribution of scaled membership vectors reflects the distribution of the original membership vectors corrected for the information in the test set.

One can understand standardized partition spaces in terms of an inversion of the standard approach to compare induced partitions of various classification rules by means of the decision boundaries in the predictor space $\mathbf{X}$ — the joint faces of the induced partitions. In high-dimensional predictor spaces, and with respect to a wide variety of classifiers with very different decision boundaries, such a comparison may be difficult to understand. In these cases, it can be helpful not to use the predictor space as the constant entity for the comparison but to standardize decision boundaries and regions — and to compare the classification rules with respect to the different patterns they generate when placing examples from some test set in these standardized regions.

The layout of examples in standardized partition spaces reflects the classifiers characteristic of classification and give a realistic impression of the classifiers performance on the test set. In this sense, standardized partition spaces are a valuable tool for the comparison of rules with respect to their quantitative assessment of the membership of objects in classes, in particular by means of accuracy and ability to separate.

Chapter 6

Comparing and Interpreting Classifiers in Standardized Partition Spaces using Experimental Design

In this Chapter we realize the comparison of classification rules in standardized partition spaces with the methods derived in Chapter 5. We will introduce a visualization of scaled membership values that can be used to analyze the influence of predictors on the class membership.

We demonstrate the use of standardized partitions spaces in analyzing the astonishing phenomenon that simple classification methods do pretty well on real data sets though their underlying premises about the true relation between predictors and classes can not reasonably be assumed to hold. We will implement a screening experiment to detect the main influencing factors for the performance of various classifiers.

6.1 Introduction

We will perform a simulation study to analyze the following problem:

Given a medium number of predictors, (10-20), and a potentially complex relation between classes and predictors, one would expect flexible classification methods like support vector machines or neural networks to do better than simple methods like e.g. the linear discriminant analysis or cart. Nevertheless, one often observes on real data sets, that the simple procedures do pretty well. Our assumption is, that simple methods are more robust against instability, and that the effect of instability superposes the effect of complexity of the relation. By instability we mean the deviation from the assumption that the collected data are an independent and identically distributed sample from a certain joint distribution of predictors and classes.

We analyze this problem with a simulation study using experimental design.

6.2 Experiment

In general, influencing factors for the goodness of any classification methods are data characteristics like the number of classes (we fix as three), the number of predictor variables (we fix as 12), the number of training objects as such (we vary), the number of training objects in classes (we use balanced design only) the number of missing values (we ignore this factor at this stage), and the form of the joint distribution of classes and predictors on objects.

Our main interest is on the influence of the form of the joint distribution of classes and predictors on objects.

The form determines the shape of the optimal partition. The joint faces $f_{c,g}$ of the partitions between two classes for continuous densities are given implicitly by the equations

$$p_\tau(g|\vec{x}) \;\; = \;\; p(h|\vec{x},\tau), \;\; g \neq h, \; g,h \in \mathbf{C}.$$

We view these implicit functions as random variables $Y_{g,h} := f_{g,h}(\vec{X})$.

For a direct systematic scanning of this space of implicit functions via simulations, we would have to fix various functions $f_{g,h}(\vec{X})$ of increasing complexity, and determine from there possible joint distributions of C and $\vec{X}$.

This is rather complicated, and moreover, the connection of this to something one can potentially know about the problem at hand, or see by exploration of the data, is difficult to understand. Thus we do not want to model the complexity of the relation via these implicit functions, but via the conditional distributions $P(\vec{X}|c,\tau)$, $c = 1, ..., G$.

One aspect of the joint distribution is the dependency structure between predictors. Often this makes the learning results less stable. Some classification methods, like e.g. the naïve Bayes, even assume independence of variables. In all cases, the analysis of the relevance of predictors for the detection of classes is obscured by this inner dependency.

Concerning the shape of the bivariate distributions between one predictor and the class, we define easiness from the perspective of the classifier from a linear discriminant analysis: easy are linear functions $f_{g,h}$, which we know can be generated from multivariate normal distributions with equal covariance matrices. These are the assumptions, a linear discriminant analysis is based on. In that case, $Y_{g,h}$ — as a sum of normally distributed variables — is also

normally distributed.

To see the effect of instability on the different types of classification methods, we model instability by deflected observations accounting for three factors: the percentage of deflected observations, the percentage of relevant variables in which the deflection takes place and the direction of the deflection.

Deflection only takes place on the test data. Thus the "true" generating process for the test data differs from that of the training data.

6.2.1 Classification Methods

We compared a set of classification methods, that are quite different in the assumptions they make about any underlying generating process of the data as we elaborated in chapters 2 and 3: the classifiers from linear and quadratic discriminant analysis (LDA, QDA), a continuous naïve Bayes classifier (CNB), a Neural Network (NN), the k-Nearest neighbor classifier (k-NN), and a decision tree classifier (TREE). In the appendix you find the description of the implementation of these classification methods.

6.2.2 Quality Characteristics

The target values in our experiment are correctness rate (**CR**), accuracy (**Ac**), and ability to separate (**AS**) just as we derived them in Chapter 5. The more overlapping the true distribution are the more difficult is the problem as such. Thus we use as target values not the performance measures as such but their relation ratios (r**CR**, r**Ac**, r**AS**) to the best that can be achieved: the corresponding performance values of the Bayes rule with respect to the true

V	G1	G2	G3	relevant for	V	G1	G2	G3	relevant for
V1	0	-1.64	1.64	G2 vs G3	V7	0	-1	1	G2 vs G3
V2	1.64	0	-1.64	G1 vs G3	V8	1	0	-1	G1 vs G3
V3	-1.64	1.64	0	G1 vs G2	V9	-1	1	0	G1 vs G2
V4	0	1.64	1.64	G1 vs rest	V10	0	0	0	Annoyance
V5	1.64	0	1.64	G2 vs rest	V11	0	0	0	Annoyance
V6	1.64	1.64	0	G3 vs rest	V12	0	0	0	Annoyance

Table 6.1: Definition of Expectation of Predictors in Groups in the non-dependent case

probability model.

6.2.3 Predictors

We also want to demonstrate the use of scaled membership values for analyzing the relevance of variables for the different classes. For that purpose we generate our predictor variables such that we have a clear concept for the relevance at least in the non-dependent case.

All univariate distributions have variance 1. They only differ in expectation, kurtosis and skewness. Table gives the defined expectations, which are either zero, one, or the upper or lower 5%-quantile of the standard normal distribution, $u_{.05} \doteq -1.64$ and $u_{.95} \doteq 1.64$.

We fix this expectations for the normal and the non-normal case. The resulting expectations, variances an covariances in the dependent case can be calculated.

6.2.4 Factors

We define the low (level=-1) and the high level (level=1) for our experimental design by looking at the easiness of the learning task.

The more training data we have, the more information we have to learn classification rules, thus we consider the number of examples in the training set as influencing factor **TO**. We set $N = 1000$ as low (difficulty) level and $N = 100$ as high (difficulty) level.

To model the deviance of the true distribution from the normal distribution, we use the Johnson System (Johnson, 1949), where random variables can be generated such that they have pre-defined first four central moments. Low and high skewness values (**S**: 0.1^2 and 1.15^2), and low and high kurtosis values (**K**: 2.7 and 5.0) are selected such that all combinations are valid, and that a wide range of distributions in the skewness-kurtosis-plane is spanned.

On the low level of dependency **Dp**, we generate independent predictor variables $X_k, k = 1, ..., K$. The high level of dependence is constructed by calculating "inverted" variables: $\tilde{X}_k := \sum_{i=1}^{K} X_i - X_k$, $k = 1, ..., K$.

We do want deflection to be the deviance from the ordinary, thus the chosen high level of 40% for the percentage of deflected observations **DO** assures that more than a half of the observations is "ordinary". We define the low level as 10%. For the low level of the percentage of deflected variables **DV** only one of the nine relevant predictor variables is deflected, on the high level, all but one.

The direction of deflection **DD** of an observation is either determined by a shift of the value in each affected predictor variable towards its mean in the

Plan No.	TO	K	S	DP	DO	DV	DD
1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	1	-1	1	1	1
3	1	-1	-1	1	-1	1	1
4	1	1	-1	-1	1	-1	1
5	1	1	1	-1	-1	1	-1
6	-1	1	1	1	-1	-1	1
7	1	-1	1	1	1	-1	-1
8	-1	1	-1	1	1	1	-1

Table 6.2: Plackett-Burmann design for seven factors. In the experiment according to each plan, factors with "-1" are set on low values, and those with "1" on high values.

true group of the observation, or away from it towards the nearest true mean of this variable in another group.

6.2.5 Experimental Design

To do the screening for detecting the relevant factors for the relative performance of the analyzed set of classifiers, we use a standard Plackett-Burman design for the seven factors under investigation. It is presented in Table 6.2.

6.3 Results

In Table 6.3 we present the values of those coefficients of the approximated linear response functions that are significant at a five percent level, the measure of the fit of the linear model based on all factors R^2, and based only on the significant factors $R^2_{5\%}$.

For all performance measures and all classification methods, the size of the training set is always significant. In general, all classification methods react

stronger in terms of the relative accuracy and the relative ability to separate, than with respect to the relative correctness rate. This is not astonishing, because deviations from the underlying model always influence the quantification of the class membership but have to reach a certain level to influence the class assignment. In that sense, the performance measures derived in Chapter 5 are more sensitive detectors of violated model assumptions than the correctness rate is.

With respect to the starting question of this experiments about the good performance of simple classification methods this is only confirmed for the LDA classifier. Its negative reactions to any disturbance is always very small. The TREE classifier though reacts quite strongly on dependency, as well as the NB and the k-NN which all considered to be simple, yet often successful methods. All of them react positively on kurtosis. The NN classifier is almost as insensitive as the LDA classifier, only its accuracy suffers from deflected observations and variables – indication that outlier detection would be an important issue in neural networks if scaled membership values were to be interpreted to analyze the relationship between predictors and classes.

The next step in the analysis of these classifiers will be the fitting of models with interactions, especially for LDA, QDA, and NN, where the low R^2-values indicate that more complex models should be fitted. In such analyses, the astonishing result that LDA also reacts positively on skewness, may turn out to be the result of interactions. as the main effects or a Plackett-Burmann plan are confounded with two-factor interactions (cp. Weihs and Jessenberger, 1999).

Meth.	Crit.	Intc.	TO	K	S	DP	DO	DV	DD	R^2	$R^2_{5\%}$
LDA	rCR	.9997	−.0189	−.0071	.0062				.0072	.89	.83
	rAc	.9973	−.0503	−.0158	.0173				.0175	.87	.81
	rAS	.999	−.0347	−.013	.0117				.0142	.89	.82
QDA	rCR	.9963	−.0583							.90	.82
	rAc	.9917	−.1334	−.0238						.91	.86
	rAS	.9933	−.0965	−.0192			.0165			.91	.88
TREE	rCR	.929	−.0966	.0479		−.1171				.91	.90
	rAc	.779	−.166	.1005		−.237				.91	.90
	rAS	.8873	−.1342	.0688		−.1806				.92	.91
NB	rCR	.999	−.1498	.1241		−.2816				.97	.96
	rAc	.9973	−.2308	.1665		−.5473	.0733			.97	.97
	rAS	.9983	−.2406	.1926		−.4381				.97	.96
k-NN	rCR	.9973	−.0586	.0286	.0167	−.0528	.0129			.98	.98
	rAc	.9927	−.1345	.0933		−.1213				.93	.91
	rAS	.9967	−.0983	.0426	.0319	−.0868	.0251			.98	.98
NN	rCR	.976	−.0384				.0139			.73	.65
	rAc	.9247	−.0894	.0396				−.0401	−.0586	.74	.72
	rAS	.9657	−.06				.0212			.74	.67

Table 6.3: Descriptions of the Approximated Response Functions

6.4 Influence Plots

We also want to demonstrate the use of scaled membership values for analyzing the relevance of variables for the different classes. As an example, we plotted the scaled membership values of the NN classifier against the ranks of V1, V2, and V3 – variables designed to be relevant to the discrimination between groups 2 and 3, groups 1 and 3, and groups 1 and 2 respectively (see Table 6.1). We propose to use the ranks rather than the predictor values as such to be independent from the measurement scale of different predictors, and to gain comparable plots for all predictor variables as it is done in Figure 6.1.

You see that the relevance of each of these variables for the separation between the respective groups can easily be read off the membership-predictor plots for plan No. 1, where all factors are set to their low values. Because of the scaling, the high level of the assignment values reflect the fact that these assignments can be trusted. Due to the black-box-nature of neural networks this information could typically not be read off that easily.

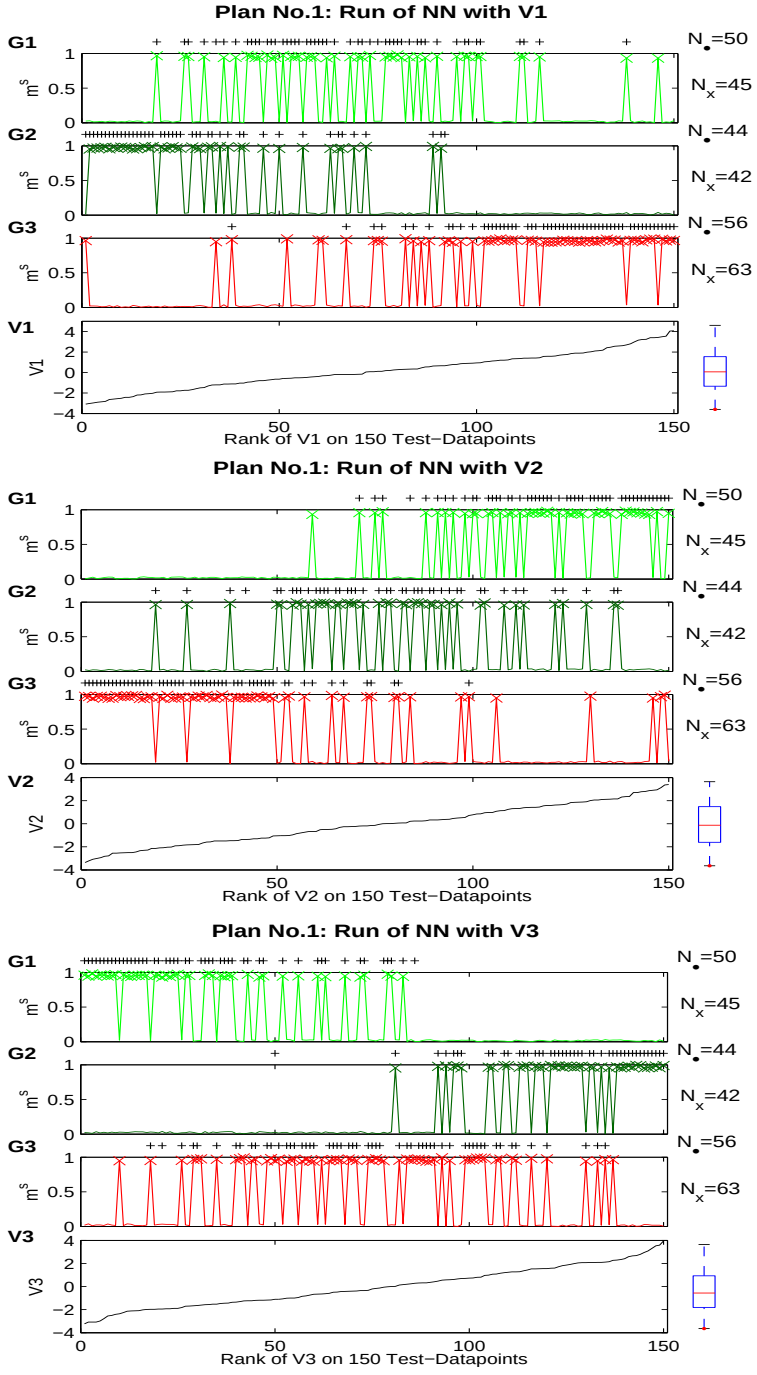

Figure 6.1: Example of a membership-predictor plot. Scaled membership values in all three groups are plotted versus the ranks of V1, V2, and V3 within a subsample of 150 points of the test set. Crosses mark the assignment values. The forth plot in each subplot displays the run of the actual value of the variables, and the shape of their marginal distribution is visualized in the boxplot at the bottom on the right side.

Chapter 7

Dynamizing Classification Rules

In this chapter, we show how to combine evidence on the class membership gained by static classification rules with background knowledge about a dynamic structure.

Combining current and past evidence is important in dynamic domains, where the task is to predict the state of the system in different time periods. We sometimes know that the succession of states is cyclic. This is for example true for the prediction of business cycle phases, where an upswing is always followed by upper turning points, and the subsequent downswing passes via lower turning points over to the next upswing and so on.

We present several ideas how to implement this background knowledge in popular static classification methods. We show that the basic strategy of this work to scale membership values into the standardized partition space is very useful for that purpose. Different from the preceding chapters, the goal of the scaling in that respect is not the comparability to other classification rules, but the compatibility with a coding of dynamic background information as evidence. Therefore, we will present an additional scaling procedure

that weights current information of an objects membership in a class with information about past memberships and compare the effect with the scaling introduced in Chapter 5.

7.1 Introduction

In the literature, business cycles are typically either treated as a univariate phenomenon and tackled by univariate time series methods, or they are modelled as a *multivariate* phenomenon and tackled by static multivariate classification methods (Meyer and Weinberg, 1975; Heilemann and Münch, 1996). As a consequence, either the time-dependency or the interplay of different variables is ignored. The same is true for many of the analyses of other cyclic phenomena that are very common particularly in economy and in biology. In biology, the importance is manifested by the fact that there is an own branch called chronobiology that studies biological rhythms, like e.g. menstruation (cp. Belsey and Carlson, 1991), cyclic movements of the small bowel (cp. Aalen and Husebye, 1991), or sleep (cp. Guimaraes et al., 2001).

In a preliminary comparative study we showed that multivariate classification methods (ignoring knowledge of time dependencies) and a dynamic Bayesian network that generalizes the Naive Bayes classifier for time dependencies (ignoring dependencies between predictors) generated about the same, unsatisfying, average prediction accuracy for the prediction of business cycle phases.

That means, some static multivariate classification methods generated the same error rates as the dynamic Bayesian network without using background

knowledge of time dependencies in business cycles. Therefore, there was hope that in order to improve prediction accuracy for the multivariate methods advantage could be taken of the cyclical structure of business cycle phases for which the following pattern is true: lower turning points $\hookrightarrow$ upswing $\hookrightarrow$ upper turning points $\hookrightarrow$ downswing $\hookrightarrow$ lower turning points $\hookrightarrow$ and so on.

In this thesis, we introduce and analyze several ideas on the incorporation of this background knowledge in different types of classification rules. The general problem of predicting cycles is formulated in Section 7.2. In Section 7.3 ideas on adapting static classification rules to cyclical structures are described. The data used for learning and testing the prediction models for business cycle phases and the design of comparison are presented in Section 7.4. Also we introduce the model assumptions of the dynamic Bayesian network we developed for the prediction of business cycles and the considered static classifiers. In Section 7.5 we present the performance of the implemented ideas for the prediction of business cycle phases. To validate the findings, we performed additional simulations that are presented in Section 7.6. And finally, consequences are summarized in Section 7.7.

7.2 Basic Notations

As in the chapters before, we consider classification problems that are based on some K-dimensional real-valued vector $\vec{x}(\omega) \in \mathbf{X} \subset \mathbb{R}^K$ measured on objects $\omega \in \mathbf{\Omega}$. And we want to decide about the class $c(\omega) \in \mathbf{C} := \{1, ..., G\}$ the object belongs to. And we drop ω in our notations, unless it is useful for understanding.

Unlike before, in case of prediction of cycle phases, we classify not really

various objects, but rather one object — called *system* — in different time periods $t = 1, ..., T$. And in each time period the system is situated in one out of G possible *states* $s \in \mathbf{S} := \{1, ..., G\}$. We will further on no longer distinguish between states and classes and denote both by s, and their space $\{1, ..., G\}$ by $\mathbf{S}$.

The chronological order of how the system passes through states is fixed: Given the system is in time period $t-1$ in state $s_{t-1} := s$, it either stays there or moves on to a certain other state $s^{\oplus}$ so that $s_t \in \{s, s^{\oplus}\} \subset \mathbf{S}$, $t = 1, ..., T$. In the following, we assume a corresponding numbering of states where $s^{\oplus} = s + 1$ for $s = 1, 2, ..., G-1$ and $s^{\oplus} = 1$ for $s = G$.

7.3 Adaption of Static Classification Rules for Prediction of Cycle Phases

For multi-class problems, there are two distinct basic structures to decide on a certain elementary class $s \in \mathbf{S}$ where the cyclical structure can easily be implemented: multi-class argmax rules or certain compositions of binary argmax rules.

Multi-class argmax rules use membership values for each elementary class $m(s, \circ | \mathbf{L}, \texttt{met}) : \mathbf{X} \to \mathbb{R}$, $s \in \mathbf{S}$:

$$\hat{s} = \hat{s}(\vec{x} | \mathbf{L}, \texttt{met}) = \arg\max_{s \in \mathbf{S}} m(s, \vec{x} | \mathbf{L}, \texttt{met}).$$

Other rules for multi-class problems define membership values not for elementary states but for various sets out of the product set over $\mathbf{S}$, $m : \mathbf{X} \to \wp(\mathbf{S})$. This is true, for example, if the final assignment is the result of a path of binary

argmax decisions, where in each step a decision is made between exactly two elements of $\wp(\mathbf{S})$.

Some of our strategies to "dynamize" static rules by integrating knowledge of a cyclic structure will rely on the assumption, that not only the size of the assessed membership of one object in different classes can be meaningfully compared but also the size of the membership values of different objects in the same class. Some combinations of binary argmax rules fulfill this need, others not. We will give examples of both:

One strategy from which we can easily read off membership values that are comparable in size between objects, is the so-called *one-against-rest* strategy introduced by Schölkopf et al. (1995) for SVM. Each class s is trained against the other classes $\urcorner s := \mathbf{S} \backslash s$. Thus G binary argmax rules are trained, each on the complete training set $\mathbf{L}$, by learning membership functions $m(s, \circ < lset, et)$ and $m(\urcorner s, \circ | \mathbf{L}, \texttt{met})$. As these membership functions are all based on the same training set, their values are comparable in size and we can combine them easily to a multi-class argmax rule by assigning to the class with the highest value among $m(s, \circ | \mathbf{L}, \texttt{met})$, $s \in \mathbf{S}$.

Not comparable in size are membership values of objects, when for the learning of the membership functions each state is trained against every other state with a binary SVM. The collection of $\frac{G(G-1)}{2}$ membership functions

$$m((s, s'), \circ | \mathbf{L}(s, s'), \texttt{met}) : \mathbf{X} \rightarrow \mathbb{R}, \ s' = 1, ..., s-1, \ s = 2, ..., G.$$

is learnt on different training sets, $\mathbf{L}(s, s')$, $\{s, s'\} \subset \mathbf{S}$ only consisting in those objects that belong to the relevant classes s and s':

$$\mathbf{L}(s, s') \ = \ \{(s_n, \vec{x}_n) \in \mathbf{L} : s_n \in \{s, s'\} \subset \mathbf{S}\}$$

The max win strategy of Friedman (1996) and the decision directed acyclic graphs (DDAG) of Platt et al. (2000) are of that type.

7.3.1 ET: Method of Exact Transitions

For any of the above mentioned argmax rules, we can take advantage of the cyclical structure by restricting the comparison of membership values to admissible transitions. That is, we start in the last known state of the system s_0 and predict the next state by

$$\hat{s}(\vec{x}_1|s_0, \mathbf{L}, \texttt{met}) \;=\; \arg \max_{s \in \left\{s_0, s_0^{\oplus}\right\}} m(s, \vec{x}_1|\mathbf{L}, \texttt{met}).$$

For the consequent time periods $t = 2, ..., T$ the predicted state $\hat{s}_{t-1}$ from the preceding time period is used as if it was the true one:

$$\begin{aligned} \hat{s}(\vec{x_t}|s_0, \hat{s}_1, ..., \hat{s}_{t-1}, \mathbf{L}, \texttt{met}) &= \hat{s}(\vec{x}_t|\hat{s}_{t-1}, \mathbf{L}, \texttt{met}) \\ &= \arg \max_{s \in \left\{\hat{s}_{t-1}, \hat{s}_{t-1}^{\oplus}\right\}} m(s, \vec{x}_t|\mathbf{L}, \texttt{met}). \end{aligned}$$

This adaption was proposed by Weihs et al. (1999) for the prediction of business cycle phases and is called *classification with exact transitions (ET)*. The classification with ET decomposes in two steps:

1. the information "$\hat{s}_{t-1}$" is used to decide on the set of admissible states $\hat{s}_t \in \left\{\hat{s}_{t-1}, \hat{s}_{t-1}^{\oplus}\right\} \subset \mathbf{S}$,

2. then, using the information in $\vec{x}_t$, we assign to that state $s \in \left\{\hat{s}_{t-1}, \hat{s}_{t-1}^{\oplus}\right\}$ with higher membership.

In the following, we will drop the time-index t and denote variables from time-slice $t-1$ with a minus: $v_- := v_{t-1}$, $t = 2, ..., T$, if statements are valid for all $t = 2, ..., T$, and where indexing is not needed for understanding.

7.3.2 WET: Method of Weighted Exact Transitions

If a system is in a certain state s, the willingness to skip to $s^{\oplus}$ in the next time period may be higher or lower than the inertia that keeps it in s. Thus, we may gain further improvement of the rules, if we exploit the information that our last assignment was into $\hat{s}_-$ not only for the decision on the admissible states, but also for the decision in which of them to assign now.

One idea is to weight membership values with an estimator of the willingness to pass over, e.g. estimated transition probabilities from the training set like observed frequencies:

$$m(s, \vec{x}|\hat{s}_-, \mathbf{L}, \texttt{met}) \;=\; w(\hat{s}_-, s)m(s, \vec{x}|\mathbf{L}, \texttt{met}) \overset{e.g.}{=} p(s|\hat{s}_-, \mathbf{L}, \texttt{met})m(s, \vec{x}|\mathbf{L}, \texttt{met}) \tag{7.31}$$

With such weighting, we assign to class $s^{\oplus}$ if the ratio of the membership in $s^{\oplus}$ to the membership in s is higher than the ratio of the quantified inertia that keeps the system in s to the quantified willingness to go from $s \rightarrow s^{\oplus}$.

$$\begin{aligned} \hat{s}(\vec{x}|\hat{s}_-, \mathbf{L}, \texttt{met}) &= \arg \max_{s=\hat{s}_-, \hat{s}_-^{\oplus}} w(\hat{s}_-, s)m(s, \vec{x}|\mathbf{L}, \texttt{met}) \\ \leftrightarrow \hat{s}(\vec{x}|\hat{s}_-, \mathbf{L}, \texttt{met}) &= \begin{cases} \hat{s}_- & \text{if } \frac{m(\hat{s}_-, \vec{x}|\mathbf{L}, \texttt{met})}{m(\hat{s}_-^{\oplus}, \vec{x}|\mathbf{L}, \texttt{met})} \frac{w(\hat{s}_-, \hat{s}_-)}{w(\hat{s}_-, \hat{s}_-^{\oplus})} \geq 1 \\ \hat{s}_-^{\oplus} & \text{otherwise.} \end{cases} \end{aligned}$$

Therefore, a necessary condition for weighting is that membership values and the willingness to skip or stay are measured on a ratio scale. Simple multiplication is used to combine these evidences, giving both of them same importance.

Yet, if membership values contain some (estimated) information on prior state probabilities, as e.g. all learnt conditional state probabilities of Bayes rules do, we would inappropriately combine conflicting information on the current prior state probability, namely:

$$"p(s|\hat{s}_-, \mathbf{L}, \texttt{met}) * p(s|\vec{x}, \mathbf{L}, \texttt{met}) = p(s|\hat{s}_-, \mathbf{L}, \texttt{met}) * p(s|\mathbf{L}, \texttt{met}) \frac{p(\vec{x}|s, \mathbf{L}, \texttt{met})}{p(\vec{x}|\mathbf{L}, \texttt{met})}".$$

But when multiplying probabilities, we assume independence. The learnt conditional state probability reflects a level of knowledge of the current state that assumes the last predicted state to be true and that the current transition follows the same generating process as the transitions in $\mathbf{L}$. Using $p(s|\mathbf{L}, \texttt{met})$ we pretend to know nothing about the current state, but that it comes from the same population as the states in the learning set. These two ideas can not be correctly represented as containing independent information. Therefore, whenever membership values involve information on prior state probabilities, an appropriate choice of weights is $\frac{p(s|\hat{s}_-, \mathbf{L}, \texttt{met})}{p(s, \mathbf{L}, \texttt{met})}$.

We simply *replace* $p(s|\mathbf{L}, \texttt{met})$ by $p(s|\hat{s}_-, \mathbf{L}, \texttt{met})$ in the calculation of membership values for Bayes rules in equation (cp. (2.2.18)):

$$\begin{aligned} m(s, \vec{x}|\hat{s}_-, \mathbf{L}, \texttt{met}) &= m(s, \vec{x}|\mathbf{L}, \texttt{met}) \frac{p(s|\hat{s}_-, \mathbf{L}, \texttt{met})}{p(s|\mathbf{L}, \texttt{met})} \\ &= \frac{p(\vec{x}|s, \mathbf{L}, \texttt{met}) p(s|\hat{s}_-, \mathbf{L}, \texttt{met})}{p(\vec{x}|\mathbf{L}, \texttt{met})}. \end{aligned}$$

The resulting membership values can be interpreted as estimated conditional state probabilities given $\vec{x}$ and $\hat{s}_-$ under the additional assumption of conditional independence of $\vec{X}$ and S_- given $S = s$. This is the well-known assumption in Hidden Markov Models (HMM), see e.g. Rabiner and Juang (1993): all relevant past information $s_0, \vec{x}_0, ..., s_{t-1}, \vec{x}_{t-1}$ is summarized in the

last state s_{t-1} and is propagated solely through the transition probabilities $p(s_t|s_{t-1}) \equiv p(s|s_-),\ s_t = 1, ..., G,\ t = 1, ..., T$. The idea to combine static probability classifiers with a Markov chain to build a dynamic classifier was introduced by Koskinen and Öller (1998).

7.3.3 HMM: Propagation of Evidence for Probabilistic Classifiers

Thus, from WET to the propagation in HMM models is only a small step for any probabilistic static classifier. We add on the Markov chain and predict states using the forward step in the forward-backward procedure for finding the next state in HMMs (cp. Rabiner and Juang, 1993). The parameters of the distributions in an HMM are the states' *transition probabilities* and the so-called *emission probabilities* of HMMs, the $p(\vec{x}|s)$, $\vec{x}\in\mathbf{X}$, $s\in\mathbf{S}$. We use the observed frequencies on the training set $\mathbf{L}$ as estimators for the transition probabilities, and the estimated conditional probabilities $p(\vec{x}|s, \mathbf{L}, \texttt{met})$ of a probabilistic classifier as emission probabilities.

We no longer propagate evidence assuming the predicted state was the true one, but we propagate the probability that a certain state is true, given the state s_0 in time period $t_0 := 0$ and the past observations of predictor variables. The general justification of this strategy is based on probability calculus and therefore relies on the interpretation of membership values as conditional class probabilities.

The first step is the same as in WET. We predict $\hat{s}_1$ using $\vec{x}_1$ and s_0:

$$\hat{s}_1 \;=\; \hat{s}(\vec{x}_1|s_0,\mathbf{L},\mathtt{met}) = \arg\max_{s=s_0,s_0^{\oplus}} p(s|\vec{x}_1,s_0,\mathbf{L},\mathtt{met}).$$

The next step is different. We do not assume $\hat{s}_1 = \hat{s}(\vec{x}_1|s_0,\mathbf{L},\mathtt{met})$ to be the true state, but we propagate to be in state s_0 with probability $p(s_0|\vec{x}_1,s_0,\mathbf{L},\mathtt{met})$ and in state $s_0^{\oplus}$ with probability $(1-p(s_0|\vec{x}_1,s_0,\mathbf{L},\mathtt{met}))$.

Thus, the joint probability to be in state $s_0^{\oplus}$ in the second time period and to observe $\vec{x}_2$ and $\vec{x}_1$ is the sum of the probabilities of the two paths that can lead from s_0 to $s_0^{\oplus}$: $s_0 \rightarrow s_0 \rightarrow s_0^{\oplus}$ and $s_0 \rightarrow s_0^{\oplus} \rightarrow s_0^{\oplus}$:

$$p(s_2,\vec{x}_2,\vec{x}_1|s_0,\mathbf{L},\mathtt{met}) \;=\; \sum_{s=s_0,s_0^{\oplus}} p(\vec{x}_2|s_2,\mathbf{L},\mathtt{met})p(s_2|s,\mathbf{L},\mathtt{met})p(s,\vec{x}_1|s_0,\mathbf{L},\mathtt{met}).$$

Later, more than two states are possible and the joint probabilities are recursively calculated by:

$$\begin{aligned} & p(s_t,\vec{x}_t,...,\vec{x}_1|s_0,\mathbf{L},\mathtt{met}) \\ &= \sum_{s\in\mathbf{S}} p(\vec{x}_t|s_t,\mathbf{L},\mathtt{met})p(s_t|s,\mathbf{L},\mathtt{met})p(s,\vec{x}_{t-1},...,\vec{x}_1|s_0,\mathbf{L},\mathtt{met}). \end{aligned}$$

The argmax rule based on these membership values is the same as from the derived conditional state probabilities:

$$p(s_t|\vec{x}_t,...,\vec{x}_1,s_0,\mathbf{L},\mathtt{met}) \;=\; \frac{p(s_t,\vec{x}_t,...,\vec{x}_1|s_0,\mathbf{L},\mathtt{met})}{\sum_{s\in\mathbf{S}} p(s,\vec{x}_t,...,\vec{x}_1|s_0,\mathbf{L},\mathtt{met})}.$$

We will call these conditional state probabilities "prior probabilities" of states $s\in\mathbf{S}$ in time period t given all past information in time period t, namely $\{\vec{x}\}_1^{t-1} := \{\vec{x}_1,...,\vec{x}_{t-1}\}$ and s_0.

Another possibility is to use the forward step of the viterbi-algorithm for propagating the evidence (cp. Rabiner and Juang, 1993). This results in a

propagation only along the currently most probable path given the observational series $\{\vec{x}\}_1^t$ and s_0. We abbreviate these two HMM algorithms as *HMM SOP*, meaning HMM propagation via the *s*um *o*f *p*aths, and *HMM MPP* meaning HMM propagation along the *m*ost *p*robable *p*ath.

7.3.4 Propagation of Evidence for Non -Probabilistic Classifiers

In cases, where membership values are not estimated conditional probabilities, technically we can apply the HMM SOP and the HMM MPP algorithms after any scaling of the membership values into the space of conditional class probabilities **U**. Particularly in artificial neural networks (ANN) it is common to scale membership vectors by the so-called softmax transformation (Bridle, 1990) that we introduced in Chapter 5 in equation (5.5.1):

$$m(s, \vec{x}|\mathbf{L}, \texttt{met}, \texttt{sof}) \quad = \quad \frac{\exp\left(m(s, \vec{x}|\mathbf{L}, \texttt{met})\right)}{\sum_{c \in \mathbf{S}} \exp\left(m(c, \vec{x}|\mathbf{L}, \texttt{met})\right)}.$$

The whole field of ANN/HMM hybrid literature is concerned with aspects of combining HMM propagation with evidence, a survey is given by Bengio (1999). Note, that our problem differs essentially in two points from typical ANN/HMM hybrid applications: the very small data size, and the goal to do forward prediction only, because our challenge is to identify the current business cycle phase, not a series of past ones. Typically, the latter is relatively easy to do in business cycles. As we have to rely only on current and past information for the classification of the current state, many ANN/HMM Hybrid procedures are not applicable.

All considerations about necessary characteristics of membership values for the WET procedure also apply, if we want to use HMM propagation. There is no problem, if the $m(s, \circ|\mathbf{L}, \texttt{met}, \texttt{sof})$ serve as good estimators for $p(\circ|s, \mathbf{\Lambda})$, $s \in \mathbf{S}$ for some probability model $\mathbf{\Lambda}$, like e.g. softmax-scaled discriminant functions of LDA. If the assumptions $\mathbf{\Lambda}$ of a multivariate normal distribution of predictors given the class and a common covariance matrix are justified, these discriminant functions are shifted logarithms of class conditional probability estimators (cp. McLachlan, 1992). In that case the softmax transformation yields the probability estimators as such:

$$\begin{aligned} \frac{\exp\left(\ln(p(c|\vec{x}, \mathbf{L}, \texttt{lda})) + k\right)}{\sum_{g \in \mathbf{S}} \exp\left(\ln(p(g|\vec{x}, \mathbf{L}, \texttt{lda})) + k\right)} &= \frac{\exp(k) p(c|\vec{x}, \mathbf{L}, \texttt{lda})}{\sum_{g \in \mathbf{S}} \exp(k) p(g|\vec{x}, \mathbf{L}, \texttt{lda})} \\ &= p(c|\vec{x}, \mathbf{L}, \texttt{lda}) \end{aligned}$$

with ln denoting the natural logarithm, and $k \in \mathbb{R}$. Also, for an increasing number of independent training samples in $\mathbf{L}$, and specific additional assumptions about the algorithm and the underlying "true" distribution, Hampshire and Pearlmutter (1990) have shown that softmax-scaled ANN outputs converge to conditional probability estimators.

But there is no general justification of the softmax transformation as leading to "good" probability estimators in terms of e.g. unbiasedness or minimum risk, nor any other general justification, why this transformation should be chosen among all potential transformations from $\mathbf{X}$ to $\mathbf{U}$. In particular in small data sets like ours, where in addition the independence assumption of the sample is violated, the interpretation of membership values from e.g. ANNs as class conditional probabilities is risky. The same is true for estimated conditional probabilities based on venturous probability model assumptions,

and there is no justification for this interpretation in case of softmax-scaled membership values of SVMs. Yet, a good scaling is essential for successfully combining static classifiers with HMM propagation (cp. Bengio, 1999). Thus, we introduce two justified ways to do the scaling.

Scaling by aiming at probability estimators: P-Scaling

For the purpose of comparing static classification rules, we developed a scaling $\texttt{sca}$ such that scaled membership values $m(s, \vec{x}|\mathbf{L}, \texttt{met}, \mathbf{V}, \texttt{sca})$ can be interpreted as an estimator $p(s|\vec{m}(\vec{x}|\mathbf{L}, \texttt{met}), \mathbf{V}, \texttt{sca})$ for the probability $p\left(s \mid \vec{m}(\vec{x}|\mathbf{L}, \texttt{met}), \tau\right)$ to be in a certain state s given the vector of membership values $\vec{m}(\vec{x}|\mathbf{L}, \texttt{met})$. The p-scaling is the scaling procedure described in Section 5.6.

Scaling by optimally weighting evidences: E-scaling

Alternatively to p-scaling, we scale membership values with a parameterized softmax transformation. We minimize the error rate on the validation set among all softmax transformation with a parameter κ:

$$m(s, \vec{x}|\mathbf{L}, \texttt{met}, \texttt{sof}(\kappa)) \;=\; \frac{\exp\left(\kappa m(s, \vec{x}|\mathbf{L}, \texttt{met})\right)}{\sum_{c \in \mathbf{S}} \exp\left(\kappa m(c, \vec{x}|\mathbf{L}, \texttt{met})\right)},\ \kappa \in \mathbb{R}^{+}.$$

We use the HMM algorithms to combine the evidence available in the observations $\{\vec{x}\}_1^{t-1}$ – quantified in the scaled membership values – with the prior knowledge of the last known state s_0. Analogously to the probabilistic case we define and regard $m(s|\{\vec{x}\}_1^{t-1}, s_0, \mathbf{L}, \texttt{met}, \mathbf{V}, sof(\kappa))$ as the prior evidence in time period t for state s given all past information.

The introduced parameter κ has the nice property to be interpretable as a weighting between the logarithm of the current prior evidence for state s and the unscaled evidence $m(s, \vec{x}|\mathbf{L}, \texttt{met})$ from the current observation $\vec{x}$. You can see this by simple transformations of the dynamized argmax rule for the assignment:

$$\begin{aligned}
&\hat{s}\left(\vec{x}_t|\left\{\vec{x}\right\}_1^{t-1}, s_0, \mathbf{L}, \texttt{met}, \texttt{sof}(\kappa)\right) \\
&= \arg\max_{s\in\mathbf{S}}\left\{m(s|\left\{\vec{x}\right\}_1^{t-1}, s_0, \mathbf{L}, \texttt{met}, \texttt{sof}(\kappa))m(s, \vec{x}|\mathbf{L}, \texttt{met}, \texttt{sof}(\kappa))\right\} \\
&= \arg\max_{s\in\mathbf{S}}\left\{\begin{array}{c} \ln\left(m(s|\left\{\vec{x}\right\}_1^{t-1}, s_0, \mathbf{L}, \texttt{met}, \texttt{sof}(\kappa))\right) \\ +\ln\left(\exp\left(\kappa m(s, \vec{x}|\mathbf{L}, \texttt{met})\right)\right) \\ -\ln\left(\sum_{c\in\mathbf{S}}\exp\left(\kappa m(c, \vec{x}|\mathbf{L}, \texttt{met})\right)\right)\end{array}\right\} \\
&= \arg\max_{s\in\mathbf{S}}\left\{\ln\left(m(s|\left\{\vec{x}\right\}_1^{t-1}, s_0, \mathbf{L}, \texttt{met}, \texttt{sof}(\kappa))\right) + \kappa m(s, \vec{x}|\mathbf{L}, \texttt{met})\right\}.
\end{aligned}$$

7.4 Design of Comparison

7.4.1 Data

The data set consists of 13 "stylized facts" Lucas (1987) for the German business cycle and 157 quarterly observations from 1955/4 to 1994/4 (price index base is 1991). The stylized facts are given in Table 7.1.

The experts' classification of the data into business cycle phases (abbreviated as PH) was done by Heilemann and Münch (1996) using a 4-phase scheme. Phases are called *lower turning points*, *upswing*, *upper turning points*, and *downswing*.

IE	real investment in equipment-gr
C	real private consumption-gr
Y	real gross national product-gr
PC	consumer price index-gr
PY	real gross national product price deflator-gr
IC	real investment in construction-gr
LC	unit labor cost-gr
L	wage and salary earners-gr
M1	money supply M1
RL	real long term interest rate
RS	nominal short term interest rate
GD	government deficit
X	net exports

Table 7.1: **Our predictors** of business cycle phases are based on economic aggregates that cover all important economic fields: real activity (labor market, supply/demand), prices, and monetary sphere. The abbreviation 'gr' stands for growth rates with respect to last year's corresponding quarter.

7.4.2 Design

There are six full cycles in the considered quarters. All methods (have to) rely on the assumption of structural stability over this period, though this is not really valid. One goal of our analysis of business cycles was to uncover the stable part of this process, valid in all six cycles. As an appropriate cross-validation design to compare methods for that purpose, we decided to perform a leave-one-cycle out (L1CO) analysis. Given our data $\mathbf{D}$ this cross-validation resulted in six test sets $\mathbf{T}_i$ consisting in the observations in the i-th cycle and corresponding training sets, $\mathbf{L}_{\setminus i} := \mathbf{D} \setminus \mathbf{T}_i, i = 1, ..., 6$.

For a fair comparison, all optimization in order to gain a rule has to be done on each of the training sets separately. Rules are then compared with respect to their prediction accuracy measured as the average prediction error

on the test sets:

$$\mathbf{APE} := \frac{1}{6}\sum_{i=1}^{6}\left(\frac{1}{T_i}\sum_{t=1}^{T_i}\mathbb{I}_{s_t}(\hat{s}_t^{\mathbf{L}\setminus i})\right),$$

where $\hat{s}_t^{\mathbf{L}\setminus i}$ is the predicted state at time period t of the i-th cycle using a classification rule learnt on $\mathbf{L}_{\setminus i}$, T_i is the number of time periods in the i-th cycle, $i = 1, ..., 6$, and $\mathbb{I}_s$ is the indicator function for state $s \in \mathbf{S}$.

This gives an average error on a new cycle, which seems to be more appropriate as performance measure than the standard, namely the average error on a single new observation. Cycles form a natural entity in the given task, and the structural instability across cycles together with a performance measure based on single observations would lead to an unwanted preference of methods that predict well on long cycles.

Whenever model selection based on cross validation is part of a classification method, we performed on each of the six training sets – each with five cycles – an inner loop of leave-one-cycle out validation. This results in a doubled leave-one-cycle out (DL1CO) strategy .

7.4.3 Methods

CRAKE

In our study, there is one classification method that is based on a multivariate time-series model: the so-called *Rake model* of Sondhauss and Weihs (1999), respectively the continuous Rake model (CRAKE). This is a dynamic Bayesian network with two time-slices, where the multivariate distribution of predictors and state in a time-slice is dependent on their realization in the preceding

time-slice in a certain way. The assumed stochastic independencies within a time-slice reflect those of the Naive-Bayes classifier. The independence assumptions between time-slices broaden those of HMMs to allow for time dependencies between predictor variables. The Rake model is a multivariate version of so-called *Markov regime switching models* (MRSM) introduced by Hamilton (1989). These are typically applied for predicting switches between two regimes based on one or a maximum of two predictor variables forming auto-regressive processes of various depths and parameters depending on the regimes(Diebold and Rudebusch, 1996).

In the Rake model (in the following denoting its model assumptions by π), the distribution of each predictor variable $X_{t,k}$ is modelled to be dependent not only on the current state s_t (like in HMMs), but also on its predecessor $x_{k,t-1}$. Yet, it is assumed to be conditionally independent of all other past and current variables (like in the Naive Bayes classifier), given s_t and $x_{k,t-1}$. Therefore, for all $\vec{x}_t \in \mathbf{X}$, $s_t \in \mathbf{S}$, $k = 1, ..., K$, $t = 1, ..., T$, it is true that:

$$p_\pi\left(x_{k,t} | \{s\}_0^t, \{x_{1,t}, ..., x_{K,t}\} \setminus x_{k,t}, \{\vec{x}\}_1^{t-1}\right) \;=\; p_\pi\left(x_{k,t} | s_t, x_{k,t-1}\right).$$

And the current state S_t is conditionally independent of the past $\{s\}_0^{t-1}, \{\vec{x}\}_1^{t-1}$ given the preceding state s_{t-1}, $t = 1, ..., T$:

$$p_\pi\left(s_t | \{s\}_0^{t-1}, \{\vec{x}\}_0^{t-1}\right) \;=\; p_\pi\left(s_t | s_{t-1}\right),\ s_t, s_{t-1} \in \mathbf{S}.$$

This is different from the Naive Bayes classifier, where (non-conditional) independence of S_t and the past is assumed, $t = 1, ..., T$.

The conditional independence assumptions in the Rake model lead to a decomposition of the joint probabilities of state variable and predictor variables

in time-slice t given s_{t-1} and $\vec{x}_{t-1}$, such that the conditional state probabilities can be calculated as follows:

$$p_\pi\left(s_t|\vec{x}_t, s_{t-1}, \vec{x}_{t-1}\right) \;=\; \frac{p_\pi\left(s_t|s_{t-1}\right) p_\pi(\vec{x}_t|\vec{x}_{t-1}, s_t)}{p_\pi(\vec{x}_t|\vec{x}_{t-1}, s_{t-1})}, t = 1, ..., T.$$

The joint conditional probabilities of the predictor variables decompose into

$$p_\pi(\vec{x}_t|\vec{x}_{t-1}, s_t) \;=\; \prod_{k=1}^{K} p_\pi(x_{k,t}|x_{k,t-1}, s_t). \tag{7.4.1}$$

Often, the local conditional distributions as defined in (7.4.1) in Bayesian networks are discrete distributions. The growth rates of the stylized facts in our data, though, would need to be discretized for that purpose. More naturally they are modelled by continuous distributions. We assumed each predictor variable $X_{k,t}|x_{k,t-1}, s_t$ to be normally distributed with autoregressive mean $\mu_{k,t} = \alpha_k(s_t) + \beta_k(s_t)x_{k,t-1}$ and variance $\sigma_k(s_t)$ and estimated the parameters according to Bayesian learning in the normal regression model(Gelman et al., 1995).

For modelling the discrete and finite distribution of transitions $S|s_-$ for each $s_- \in \mathbf{S}$ we choose a Bayesian multinomial distribution model. The cyclic structure is taken care of by an informative Dirichlet prior that sets inadmissable probabilities to zero. For further details, see Gelman et al. (1995).

Exact forward propagation of evidence in dynamic Bayesian networks was used to predict the phase of the cycle in time period t, $t = 1, ..., T$, given the respective evidence of the past and the present:

$$\hat{s}_t \;=\; \arg\max_{s \in \mathbf{S}} p(s|\left\{\vec{x}\right\}_1^t, s_0, \mathbf{L}, \texttt{met})$$

7.4.4 Linear Discriminant Analysis and Support Vector Machines

In the past, mainly static classification methods were used for the multivariate prediction of business cycle phases. One reason is the fact that typically the last true phase is not observed to do the prediction for the next one. It is only by observing the continuing evolution of the economy for some more quarters that it becomes apparent what phase the business cycle was in. Another reason for using static methods is that we are not only reaching for a good prediction, but also for a description of phases in terms of the stylized facts. Thus, methods were applied that use as predictors observed entities, and for which we want to understand the connection they have to business cycle phases.

The ideas of modifying static methods outlined in Section 7.3 now allow for both, description and prediction: we describe phases using their membership functions $m(s, \circ|\mathbf{L}, \texttt{met}) : \mathbf{X} \rightarrow \mathbb{R}$, $s \in \mathbf{S}$ and we try to get better predictions by combining this evidence with the knowledge on the cyclical structure.

Starting point of this investigation were the results on a comparative study on the performance of various multivariate classification methods for the stable prediction of business cycle phases (Sondhauss and Weihs, 2001b). "Best" results of 37% average prediction error were gained with a certain *artificial neural network* (ANN) algorithm and a specific bootstrapped optimization procedure added on QDA — both time consuming procedures. The good result of the ANN is highly dependent on the randomly chosen starting values of the parameters of the network, and could never be confirmed with other starting values. With 43 % APE the performance of the simple LDA procedure based

on a selection of best two predictors was not much worse and comparable to the performance of CRAKE. Thus there was hope by using the knowledge of the cyclical structure to find a simple model that does a better job in extracting the stable part of the multivariate phenomenon "business cycles".

The hope was justified: combining LDA with HMM propagation of evidence and a DL1CO selection of the best two predictors, we found a model with a lower APE than any of the more complex models, and in each of the six inner loops of the DL1CO design, always the same two predictors were chosen as the best, namely L and LC.

In this paper, we will concentrate on the comparison of the various methods introduced in Section 3 to incorporate cyclic knowledge in general argmax classifiers. Therefore, we picked from the various originally considered argmax classifiers LDA, as it led to the best results, and a specific binary support vector machine (SVM) representing non-probabilistic classifiers. We prefer SVM over neural networks, because they do not suffer from the problem with the starting values. The results of the various add-on strategies will be compared to those with CRAKE, the fully dynamic model and motivator for dynamizing static classifiers. We apply these methods to the complete set of 13 stylized facts, and only to the two stable predictors, L, and LC, found with LDA. This will reveal the discriminating reaction of these basic classifiers on noisy signals in lower and higher dimensions.

The LDA as we realized it, is a Bayes rule with uniform class priors and equal costs. Given the phase of a certain quarter s_t, the predictor variables $\vec{X}_t$ are assumed to be distributed according to a multivariate normal distribution.

The distributions differ in the vector of means $\vec{\mu}_t = \vec{\mu}(s_t)$ of the predictor variables, but not in the arbitrary symmetric and positive definite covariance matrix $\Sigma_t \equiv \Sigma$, $t = 1, ..., T$. We used the implementation in `R` (Ihaka and Gentleman, 1996). For the e-scaling not the estimated conditional probabilities were presented as membership values, but their logarithms for the reasons given in Section 3.4.

The SVM was realized with radial basis function (RBF) kernels (cp. Vapnik, 1995; Schölkopf et al., 1995)) as implemented in the SVM toolbox 2.51 for Matlab by Schwaighofer (2002).

More details on these classifiers are given in chapters 2 and 3.

One of the parameters in SVM with RBF-kernels, typically denoted by C, controls the trade off between margin maximization and error minimization. This was set to be the maximum of 10 and the number of predictors. The second parameter defines the width of the RBF-kernel. This was set to be equal to the number of the predictors. These settings were found to match the findings in our preliminary studies, where we optimized the L1CO errors in the inner loop of both of these parameters using experimental design with a quadratic loss function.

7.5 Results

In Tables 7.2 and 7.3 you see the performance of the LDA, SVM, and CRAKE procedures based on all predictors and based on two predictors.

Overall best for the prediction of business cycles is LDA using L and LC as predictors only, when combined with an HMM SOP prediction and without

			Added cyclic information via			
classifier	scaling	original	ET	WET	HMM SOP	HMM MPP
LDA	none	47.84	53.91	50.79	47.66	**46.61**
	p-scal	54.66	56.58	51.15	**46.05**	47.27
	e-scal	**46.63**	53.91	65.88	49.29	53.74
SVM	none	52.11	50.73	59.28	**44.48**	50.20
	p-scal	49.05	49.39	61.19	**36.15**	49.22
	e-scal	50.61	50.73	58.54	**38.57**	42.33
CRAKE	none	**42.04**				

Table 7.2: Comparison of average prediction errors for all classifiers based on all predictors

			Added cyclic information via			
classifier	scaling	original	ET	WET	HMM SOP	HMM MPP
LDA	**none**	43.56	47.32	53.34	**33.27**	35.81
	p-scal	41.36	51.95	54.48	**41.19**	50.09
	e-scal	39.69	47.32	49.64	**36.62**	38.64
SVM	**none**	48.76	56.12	**43.97**	44.26	45.01
	p-scal	53.45	58.08	64.13	**52.24**	54.26
	e-scal	**46.74**	56.12	50.57	47.38	49.40
CRAKE	none	**44.93**				

Table 7.3: Comparison of average prediction errors for all classifiers based only on L and LC as predictors

scaling (33.27 % APE) , followed by SVM using all predictors with HMM SOP prediction and p-scaling (36.15 % APE). SVM and LDA react diametrically on the dimension of the presented data: SVM is much better (36.15 % vs. 43.97 % APE) on the 13 dimensional data and LDA on the two dimensional (33.27 % vs. 46.05 % APE). Apparently, LDA gets irritated by additional predictors with possibly low signal, whereas SVM needs overall more signal, and can find it in higher dimensional data without being so much distracted by noise.

Comparing the different ways to incorporate information on the cyclic structure, obviously, using HMM SOP or HMM MPP is superior to ET or

true	4 4 4 4 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 3 3 3 3 3 3 3 3
original	4 3 4 3 3 1 4 3 1 1 1 1 1 1 1 1 1 1 1 1 1 2 3 3 3 4 3 3 3 3 4
ET	4 1 1 1 1 1 2 3 4 1 1 1 1 1 1 1 1 2 2 3 4 1 2 3 4 4 4 4 4 4 4 4
WET	4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Table 7.4: P-scaled original static predictions of the BSVM classifier vs. those with ET or WET compared to the true phases in the fifth cycle. 4 is a lower turning point, 1 an upswing, 2 an upper turning point, and 3 an downswing

WET, as well as to the original results. Actually, with ET of WET, most of the times results are even worse than the original. At first, this was unexpected. Yet, looking at some specific series of predictions reveals the pitfalls one can run into with these methods. A characteristic example is the course of predictions of the SVM classifier presented in Table 7.4.

With ET once the classifier has mispredicted, it has big difficulties to predict the phase for the consequent quarters, because it is only allowed to compare for example upper turning points (2) with downswing (3), where the evidence in the predictor variables potentially indicates the true upswing (1). After an error, either the classifier 'waits' in the mispredicted state for the cycle to pass that state, or it passes through all states, until prediction and true state meet again. Obviously, WET simply sticks much too long in phase 1.

The issue-related problem to reveal the stable part of the multivariate phenomenon "business cycles" is solved by the good – compared to what we can expect to achieve – performance of the LDA classifier with HMM SOP propagation. Method-related, though, the results up to now were disappointing in that no clear pattern of superiority of any method to incorporate cyclic information can be read off Tables 7.2 and 7.3. Since this might be an artefact of

the sample systematic simulations were carried out, the designs and the results of which will be discussed in the next section.

7.6 Simulations

The dependencies in economic aggregates themselves are dependent on political frameworks and on world trade contracts, as well as on incidents as wars and global crises. Within the time-period of our observation at least the oil crises and the German Reunification will have had an influence of that type. Thus, we do not expect our data on different quarters to really give us examples of the same concept.

Yet, to see some examples from at least almost the same is necessary for any learning (algorithm). Therefore, for the evaluation of methods, the instability is disastrous: No method can be convincingly good, and differences in the performance tell as much about the data as about the methods. Therefore, we simulated data from stable processes.

7.6.1 Data

From the over countable space of processes, one will always be able to find some process that supports the method, one is in favor of. Thus, to avert arbitrariness we selected three processes motivated from the original problem, and with parameters estimated according to Bayesian learning from the original problem. By Bayesian learning, we account for the uncertainty we have about the estimated parameters.

The transition between phases is generated according to a Markov process. For the likelihood distributions of the predictors, given the course of phases, we use the distributional assumptions of three different generating models: LDA based on all 13 predictors (GLDA), LDA based on L and LC as predictors only (GLDL), and CRAKE based on all predictors (GCRA).

GLDL is chosen, because the classifier that estimated state conditional probabilities according to that two-dimensional model performed best on our real data set. GLDA will enable us to compare low dimensional and higher dimensional behavior of our algorithms. And finally, GCRA as a generating model was selected, because the performance of the corresponding classifier was not too bad, and because random variables from this model violate of the HMM assumption. Thus, we can see the effect of an extreme violation of the HMM assumption on our algorithms.

We did 20 replications in our simulation. The series of phases of each replication consists in six business cycles, and drives the observational series in each of the three models.

7.6.2 Design

We compare the behavior of algorithms in two different cross-validation designs:

1. DL1CO just as on the real data to understand the performance of our algorithms on small data sets and

2. a learning- validation-, and test set (LVT) design. Here the data from

7 replications is used for learning, the data from the next seven replications for validation (used to optimize the scaling), and the data from the remaining 6 replications is used to estimate the prediction error rates. This analysis will show us the potential benefit of algorithms for large data sets.

7.6.3 Results

DL1CO Design

			Added cyclic information via			
classifier	scaling	original	ET	WET	HMM SOP	HMM MPP
LDA	**none**	27.57	38.30	29.56	**17.86**	21.46
	p-scal	33.06	42.22	30.74	**21.47**	24.99
	e-scal	26.58	38.30	31.13	**18.07**	22.41
SVM	none	45.96	52.31	61.86	**41.54**	42.95
	p-scal	48.75	53.97	61.61	**41.25**	45.17
	e-scal	43.80	52.31	51.12	**38.28**	40.85
CRAKE	none	**40.40**				

Table 7.5: Mean average percentages of prediction errors in 20 replications of a DL1CO design on data generated according to GLDA

			Added cyclic information via			
classifier	scaling	original	ET	WET	HMM SOP	HMM MPP
LDA	none	41.11	45.69	51.07	**29.70**	32.71
	p-scal	45.14	49.73	51.68	**34.13**	38.06
	e-scal	39.63	45.69	40.30	**29.02**	32.77
SVM	none	49.47	53.78	61.24	**46.23**	49.82
	p-scal	51.83	54.33	65.80	**44.35**	53.34
	e-scal	48.30	53.78	58.61	**45.51**	48.08
CRAKE	none	**30.19**				

Table 7.6: Mean average percentages of prediction errors in 20 replications of a DL1CO design on data generated according to GLDL

Comparing the basic classifiers LDA, SVM, and CRAKE

As you can see in Table 7.5, for the GLDA process, the LDA classifier that estimates probabilities according to the correct model resulted in the best performance of 17.86 % mean average percentage of prediction errors (MAPE). This is the benchmark for the performance of CRAKE and SVM that both performed substantially worse with MAPEs around 40 %.

In case of the GLDL process, there is less information in the two predictors on the phase series compared to the 13 predictors of the GLDA process. Therefore, the performance of the benchmark classifier is worse compared to GLDL (29.02 % vs. 17.86 %). These 29.02 % MAPE are supporting the best result (33.27 % APE, Table 7.3) of LDA on the real two dimensional data: Even if the real business cycles would be generated by a smooth and stable GLDL process, we would not expect to be better than 29.02 % APE!

CRAKE almost predicted as good (30.19 % MAPE) as LDA in this setting. The CRAKE model can in a way emulate the GLDA or GLDL model, as the predictors in the CRAKE model have a common ancestor (the preceding phase) which results in a dependency. Thus, the dependency of the predictors within a time-slice of the true generating process is imitated by the predictors dependency on their predecessors having a common parent. Of course this emulation is better, when there is only one pairwise dependency to cover, and not $\binom{13}{2} = 78$ dependencies. The difficulties of SVM on this data is not surprising, as SVMs were designed to be good on high dimensional data, not necessarily on low dimensional data.

We omitted the table of the performances of the algorithms on the GCRA

data, because the general performances of LDA and SVM were too bad for a meaningful comparison of different ways to incorporate cyclic information: the 'best' LDA prediction was based on the originally estimated class conditionals with 56.63%. The 'best' SVM performance was even worse, with e-scaled membership values without cyclic information on average 59.87% predictions were wrong! This is far away from benchmark performance of the CRAKE classifier with 24.70 %. We can conclude, that on small data sets the quality of the methods depends highly on the validity of the Hidden-Markov assumption.

Comparing the different ways to incorporate cyclic information

The poor performance of ET and WET is affirmed by these simulations. Given the HMM assumption is justified, it is better to propagate the evidence along all paths (HMM SOP) than restricting the propagation to the most probable path (HMM MPP).

Comparing the scaling methods

In the DL1CO simulation we can see that scaling improves the SVM performance, though it can not turn poor prediction to really good one. No clear superiority between p-scaling and e-scaling can be read off the results in the Tables 7.5 and 7.6.

LVT Design

Comparing the basic classifiers LDA, SVM, and CRAKE

Given enough data, SVM with 14.81 % prediction errors (PE) clearly outperfomed CRAKE with 20.09 % PE on the 13 dimensional GLDA data (Table

			Added cyclic information via			
classifier	scaling	original	ET	WET	HMM SOP	HMM MPP
LDA	none	22.73	22.86	28.05	**11.69**	11.82
	p-scal	22.73	22.99	23.38	11.43	**10.91**
	e-scal	22.99	22.86	23.12	11.04	**10.65**
SVM	none	21.43	19.87	30.39	**16.23**	18.31
	p-scal	21.56	19.87	30.26	**14.94**	16.62
	e-scal	20.00	20.13	29.09	**14.81**	15.32
CRAKE	none	**20.09**				

Table 7.7: Percentages of Ppediction errors in LVT design on data generated according to GLDA

			Added cyclic information via			
classifier	scaling	original	ET	WET	HMM SOP	HMM MPP
LDA	**none**	36.75	48.18	53.77	**22.73**	25.19
	p-scal	35.97	50.78	49.87	**22.99**	23.09
	e-scal	36.75	48.18	53.77	**22.73**	25.19
SVM	none	37.40	47.01	65.97	**28.83**	35.06
	p-scal	37.40	47.14	58.18	**26.88**	31.17
	e-scal	35.84	46.88	56.23	**26.88**	**26.88**
CRAKE	none	**22.86**				

Table 7.8: Percentages of prediction errors in LVT design on data generated according to GLDL

7.7). The emulating quality of CRAKE for GLDL becomes abundantly clear on the low dimensional GLDL data (Table 7.8): with 22.86% PE it was only 0.13 percentage points worse than the prediction with the correct classifier LDA combined with HMM SOP (22.73% PE)! Compared to the results in the DL1CO design, one can see that SVM profited a lot from more data. With 26.88 % PE its relative performance to the benchmark was not so bad any more. Also, the predicting performance on the GCRA data (Table 7.9) of LDA (35.58% PE) and SVM (21.95% PE) was no longer beyond being presentable, yet still very poor compared to the benchmark of 10.13% PE according to

classifier	scaling	original	Added cyclic information via ET	WET	HMM SOP	HMM MPP
LDA	**none**	**36.49**	46.23	57.92	42.21	47.92
	p-scal	**35.97**	47.66	58.31	46.62	50.78
	e-scal	**35.58**	46.23	50.13	42.21	42.86
SVM	none	**23.25**	27.01	35.06	25.84	26.62
	p-scal	**23.25**	27.01	33.64	25.19	26.36
	e-scal	**21.95**	26.88	28.96	22.86	23.38
CRAKE	none	**10.13**				

Table 7.9: Percentages of prediction errors in LVT design on data generated according to CRAKE

CRAKE. SVM is substantially superior to LDA that obviously can not revert the emulation of CRAKE on GLDL.

Comparing the different ways to incorporate cyclic information

With more data, HMM SOP no longer generally outperformed HMM MPP. On the GLDA data and added on the conditional state probability estimators in the true model (LDA) propagation along the most probable path was always superior, irrespective of the scaling. As HMM SOP based on the true conditional state probabilities would result in the best Bayes predictions, this is astonishing. HMM MPP rests upon sums of fewer yet basically the same estimated probabilities as HMM SOP, thus a reduced variance of the predictions is our working hypothesis about the causes of these findings. This would support the intuitive appeal of the HMM MPP procedure to drop "circumstantial" evidences and to concentrate on the most probable. On the GCRA data, where the HMM assumption is heavily violated, all of our ideas to incorporate cyclic information was harm- instead of useful.

Comparing the scaling methods

For SVM, the improvement with scaled membership values levelled off at about 1-2 percentage points of PE. E-scaling that optimizes the HMM combination of evidence for prediction is better than p-scaling, even if the HMM assumption is not justified. As it should in case of the LDA classifier, the e-scaling parameter was approximately 0.9 for GLDA or equal to one for GLDL that is, the softmax transformation resulted in (almost) identity mapping.

7.7 Summary

Summarizing, from the analysis of the results one might deduce the following general conclusions on the incorporation of background knowledge of a cyclical state structure into classification rules:

- Prediction based on classification with (weighted) exact transitions is risky, because one false prediction might entail many succeeding errors.
- With HMM propagation we found a simple and convincing method to extract the stable part of the business cycle process on the real data.
- Yet, when the basic HMM assumption that the past evidence of a multivariate time series is comprised in the current state was heavily violated, in our simulations all of our ways to incorporate knowledge of the underlying cyclical structure was harmful.
- Both scaling procedures did a good job on scaling membership values

from the non-probabilistic classifier SVM. E-scaling is easier and optimizes propagation, therefore it is favorable for this purpose. There is no risk in using it, because even if it is performed on the "correct" estimators, it does no harm.

- On the basis of very well scaled membership values, HMM propagation along the most probable path is a worthy alternative to full HMM propagation of evidence.

Chapter 8

Conclusions

In this thesis a general approach to the comparative presentation of classification rules is introduced: an adequate scaling into the standardized partition space. This offers a unifying framework for the representation of a wide variety of classification rules.

For that purpose in chapters 2 and 3 we give a summary on varying theoretical frameworks of classification techniques in statistics and machine learning. By a schematic representation of their proceedings similarities and differences are worked out. It becomes clear that the differences in their basic approaches can be understood in terms of their different definitions of what is expected – the "expected loss". The main impact these varying expectations have is on the functional space in which potential classification rules lie.

Irrespective of the dissimilarity of the approaches on the theoretic level, on the technical level their proceedings are quite similar: based on training data membership functions are learnt that enable to quantify the membership of some object in classes on the basis of its predictor values.

The unified representation of these membership functions in the standardized partition space is presented in Chapter 5. Linchpin is the adequate scaling of the values from different membership functions into this space. A method that results in a presentation of membership assessments that reflect the classifiers characteristic of classification and give a realistic impression of the classifiers performance on a test set is introduced.

In the scaling procedure as we present it, the focus is on the membership values in the assigned classes – the assignment values. The scaling of the membership values in the other classes is treated secondarily. It is both challenging in theory and implementation to extend this such that scaled membership values in all classes reflect the actual behavior of the classification rule.

The benefit of the presentation of classification rules in standardized partition spaces is manifold: firstly, it facilitates comparison of classification rules. In standardized partition spaces one can motivate and visualize performance criteria that enable to evaluate the quality of membership functions.

Given this quality is high, scaled membership values provide an excellent basis for the analysis of the interplay of class membership and predictor values that now can be presented in multifarious ways.

In Chapter 6 we demonstrate the comparison of classification rules in standardized partition spaces. This is intended to impart an impression of the possibilities of the procedure.

In Chapter 7 we show that the unified presentation is not only useful for the comparison of classification methods and for the interpretation of classification rules. It can also be utilized to combine evidence on class membership

from varying sources. In Chapter 7 past and current information on class membership are merged. This changes the aim of the scaling of membership values and accordingly an alternative scaling procedure is shown to be better for this purpose. Its superiority refers only partly to the prediction results – they are better but not to a large extent. The main advantages are that it is easier to perform and that it provides further insight into the way evidences are combined.

Overall, this thesis gives a first insight into the possibilities of a unified representation of classification rules in standardized partition spaces. It remains to be seen in further application whether the effort of adequate scaling shows as much profit as the author believes.

Appendix A

References for Implemented Classification Methods

Linear and Quadratic Discriminant Analysis The general outline of linear and quadratic discriminant analysis and the specific features of their implementation we chose to use in this work are given in Sections 2.3.2, 3.3.2, and 3.5.1. For the calculations we used the procedures `lda` and `qda` in `R` (Ihaka and Gentleman, 1996) with standard moment estimation of means and variances.

Continuous naïve Bayes The distributional assumptions and the learning of parameters of the implemented continuous naïve Bayes classification method are derived in Sections 2.3.4, 3.3.2, and 3.5.1. For the programming we used `Matlab`®(Matlab).

Univariate Binary Decision Tree The representation of univariate binary decision trees and the basic strategies for learning them have been described in Sections 3.3.1 and 3.5.3. For the implementation, we use the `rPart` procedure in `R` (Ihaka and Gentleman, 1996). The chosen splitting criterion is the 'information' split. We optimize the minimum number of observations in a leaf node via lowest validation error. The validation set consists of a sampled quarter of the training set. The final tree with optimized number of observations in the leaf node is learnt on the complete training set.

Support Vector Machine Support vector machines as we use them are described in Sections 3.3.3, 3.4.2, 3.4.4 and 3.5.2. We implemented support vector machines with radial basis functions and solved the quadratic optimization problem for the learning of the best separating hyperplane with the (SVM) toolbox 2.51 for Matlab by Schwaighofer (2002). For selecting the best parameter pair for the training error penalty and the width of the kernels, we minimized the validation set error using experimental design, as described in Section 3.5.2. The validation set consists of a sampled quarter of the training set. The final hyperplane with optimized parameters is learnt on the complete training set.

K-Nearest Neighbor The k-nearest neighbor procedure is a specific instance-based learning method, where learning consists mainly of storing the presented training data (Mitchell, 1997). For k-nearest neighbor learning one assumes that objects that are "near" to each other according to their predictor values will most likely belong to the same class. The main representation task consists

in defining nearness on the space of predictor variables **X**. A second modelling decision is about the number k of nearest neighbors one wants to consider. This explains the "k" in the name of the method. Given the notion of distance and the number of neighbors to consider, some new object is assigned to the class most of its neighbors belongs to. A detailed description can be found e.g. in Mitchell (1997).

We implemented the k-nearest neighbor algorithm with Euclidian distance. Overfitting is no problem in k-nearest neighbor learning, therefore we selected the number k of neighbors by minimum training error among all natural numbers smaller or equal to the factorial of the number of classes.

Neural Network Alongside the implemented support vector machine, neural networks result in the most flexible decision boundaries in the predictor space among the classification methods in this work. From a statistician's point of view, they can be understood in terms of a highly non-linear regression method. Support vector machines and neural networks are highly related via their common ancestor: Rosenblatt's perceptron (see Vapnik, 1995). Neural networks are inspired and built on the basis of a rough analogy to biological learning systems that consist out of very complex webs of interconnected neurons. For an introduction to neural networks, see (Mitchell, 1997), and for a deeper understanding of neural networks as classifiers, see Bishop (1995).

We modelled the neural network as a feedforward two-layer network with logistic hidden layer units and linear output units, the standard approach to non-linear function approximation. For the optimization of the parameters of the net given its structure, we used the Levenberg-Marquardt backpropagation

algorithm and minimized the mean squared error. We selected good starting values and the best number of hidden nodes with respect to the error rates of the learnt net on a validation set. The validation set consists in a sampled quarter of the training set. The final net is learnt on the complete training set using Bayesian regularization of the Levenberg-Marquardt training. For details, see Demuth and Beale (1998)).

Bibliography

101 Zen Stories. The stone mind. In P. Reps, editor, *Zen Flesh, Zen Bones: A Collection of Zen and Pre-Zen Writings*. Tuttle Publishing, Boston, 1957.

AAAI. Dynamic library of introductory information about artificial intelligence — machine learning, 2000-2002. URL `http://www.aaai.org/AITopics/html/machine.html`.

O. O. Aalen and E. Husebye. Statistical analysis of repeated events forming renewal processes. *Statistics in Medicine*, 10:1227–1240, 1991.

T. W. Anderson. *An Introduction to Multivariate Statistical Analysis*. John Wiley & Sons, New York, 1958.

V. Barnett. *Comparative Statistical Inference*. John Wiley & Sons Ltd, Chichester, 3 edition, 1999.

E. M. Belsey and N. Carlson. The description of mentrual bleeding patterns: Toward fewer measures. *Statistics in Medicine*, 10:267–284, 1991.

Y. Bengio. Markovian models for sequential data. *Neural Computing Surveys*, 2:129–162, 1999.

J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, New York, 2 edition, 1995.

J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. John Wiley & Sons, Chichester, 1994.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, England, 1995. ISBN 0-19-853864-2.

C. Blake and C. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

J. Blythe. An overview of planning under certainty. In M. Wooldridge and M. M. Veloso, editors, *Artificial Intelligence Today: Recent Trends and Developments*, volume 1600 of *Lecture Notes in Computer Science*. Springer, 1999.

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, Pacific Grove, 1984.

J. S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fogelman Soulié and J. Hérault, editors, *Neuro-computing: Algorithms, Architectures and Applications*, pages 227–236, Berlin, 1990. Springer.

J. S. Bridle. Optimization and search in speech and language processing, 1995.

C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

R. T. Carroll. The skeptic's dictionary, 1994-2002. URL `http://skepdic.com/homepage.html`.

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159, 2002.

H. Demuth and M. Beale. *Neural Network Toolbox User's Guide*, 1998.

F. X. Diebold and G. D. Rudebusch. Measuring business cycles: a modern perspective. *The Review of Economics and Statistics*, 78:67–77, 1996.

T. S. Ferguson. *Mathematica Statistics — A Decision Theoretic Approach.* Academic Press, New York, 1967.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

J. H. Friedman. Another approach to polychotomous classification. Technical report, Stanford Department of Statistics, 1996.

K. Fukunaga. *Introduction to Statistical Pattern Recognitionen.* Academic Press, New York, 2nd edition, 1990.

U. Garczarek and C. Weihs. Comparing classifiers in standardized partition spaces using experimental design. Technical report, SFB 475, University of Dortmund, 2002. 21/02.

U. Garczarek, C. Weihs, and U. Ligges. Prediction of notes from vocal time series. Technical report, SFB 475, University of Dortmund, 2002. to be published ??/02.

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis.* Chapman & Hall, New York, 1995.

B. German. Glass identification database, 1987. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

G. Guimaraes, J.-H. Peter, T. Penzel, and A. Ultsch. A method for automated temporal knowledge acquisition applied to sleep related breathing disorders. *Artificial Intelligence in Medicine*, 23:211–237, 2001.

J. D. Hamilton. A new approach to the anylysis of nonstationarity time series and the business cycle. *Econometrica*, 57:357–384, 1989.

J. B. I. Hampshire and B. A. Pearlmutter. Equivalence proofs for multilayer perceptron classifiers and the Bayesian discriminant function. In *Proceedings of the 1990 Connectionist Models Summer School, 1990. D. Touretzky, J. Elman, T. Sejnowski, and G. Hinton, eds. Morgan Kaufmann, San Mateo, CA.*, 1990.

D. J. Hand. *Construction and Assessment of Classification Rules*. John Wiley & Sons, Chichester, 1997.

D. J. Hand. Data mining: statistics and more? *The American Statistician*, 52:112–118, 1998.

M. H. Hansen and B. Yu. Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, 96(454): 746–774, 2001.

U. Heilemann and H. J. Münch. West german business cycles 1963-1994: A multivariate discriminant analysis. In *CIRET–Conference in Singapore, CIRET–Studien 50*, 1996.

E. J. Horvitz, J. S. Breese, and M. Henrion. Decision theory in expert systems and artificial intelligence. *International Journal of Approximate Reasoning*, 2:247–302, 1988.

R. Ihaka and R. Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.

M. Jaeger. Relational Bayesian networks. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, pages 266–273. Morgan Kaufmann, 1997.

N. L. Johnson. Bivariate distributions based on simple translation systems. *Biometrika*, 36:149–176, 1949.

L. Koskinen and L.-E. Öller. A hidden markov model as a dynamic bayesian classifier, with an application to forecasting business-cycle turning points. Technical report, National Institute of Economic Research, 1998. 59.

P. Langley. *Elements of Machine Learning*. Morgan Kaufmann, San Francisco, CA, 1995.

E. L. Lehmann. *Theory of Point Estimation*. Springer-Verlag, New York, 1983.

D. D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In C. Nédellec and C. Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, pages 4–15, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

R. E. Lucas. *Models of business cycles*. Basil Blackwell, New York, 1987.

D. G. Luenberger. *Introduction to Linear and Nonlinear Programming*. Addison-Wesley Publishing, Reading, Mass, 1973.

M. E. Maron. Automatic indexing: An experimental inquiry. *Journal of the Association for Computing Machinery*, 8:404–417, 1961.

M. E. Maron and J. L. Kuhns. On relevance, probabilistic indexing and information retrieval. *Journal of the Association for Computing Machinery*, 7: 216–244, 1960.

Matlab. *Using Matlab*, 1998.

G. J. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition*. Wiley, New York, 1992.

J. R. Meyer and D. H. Weinberg. On the classification of economic fluctuations. *Explorations in Economic Research*, 2:167–202, 1975.

R. S. Michalski. Understanding the nature of learning: issues and research directions. In R. S. Michalski, J. Carbonnel, and T. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, volume 2, pages 3–25. Kaufmann, Los Altos, 1986.

T. M. Mitchell. *Machine Learning*. McGraw-Hill, Boston, Massachusetts, 1997.

A. M. Mood, F. A. Graybill, and D. C. Boes. *Introduction to the Theory of Statistics*. McGraw-Hill, Singapore, 3 edition, 1974.

J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. *Advances in Neural Information Processing Systems*, 12: 547–553, 2000.

J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

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

L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall Inc., New Jersey, 1993.

N. Sauer. On the density of families of sets. *Journal of Combinatorial Theory Series A*, 13:145–147, 1972.

B. Schölkopf, C. J. C. Burges, and V. N. Vapnik. Extracting support data for a given task. In U. M. Fayyad and R. Uthurusamy, editors, *Procceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 252–257. AAAI Press, Menlo Park, 1995.

B. Schölkopf and A. Smola. Kernel machines, 2000-2002. URL `http://www.kernel-machines.org/index.html`.

B. Schölkopf, K.-K. Sung, C. J. C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with Gaussian kernels to radial basis function classiers. *IEEE Transactions on Signal Processing*, 45 (11):2758–2765, 1997. URL `citeseer.nj.nec.com/sch97comparing.html`.

A. Schwaighofer. Svm toolbox for matlab. `http:// www.igi.tugraz.at /aschwaig /software.html`, 2002.

J. Shawe-Taylor and P. L. Bartlett. Structural risk minimization over data-dependent hierarchies. *IEEE Trans. on Information Theory*, 44(5):1926–1940, 1998.

J. Shawe-Taylor, P. L. Bartlett, R. C. Williamson, and M. Anthony. A framework for structural risk minimisation. In *Computational Learing Theory*, pages 68–76, 1996.

U. M. Sondhauss and C. Weihs. Dynamic bayesian networks for classification of business cycles. Technical report, SFB 475, University of Dortmund, 1999. 17/99.

U. M. Sondhauss and C. Weihs. Combining mental fit and data fit for classification rule selection. Technical report, SFB 475, University of Dortmund, 2001a. 24/01.

U. M. Sondhauss and C. Weihs. Incorporating background knowledge for better prediction of cycle phases. Technical report, SFB 475, University of Dortmund, 2001b. 24/01.

U. M. Sondhauss and C. Weihs. Standardizing the comparison of partitions. Technical report, SFB 475, University of Dortmund, 2001c. 31/01.

V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 1995.

K. Vogtländer and C. Weihs. Business cycle prediction using support vector methods. Technical report, SFB 475, University of Dortmund, 2000. 21/00.

Webster's Dictionary. *Webster's Encyclopedic Unabridged Dictionary of the English Language.* Gramercy Books, New York, Avenel, 1994.

C. Weihs and J. Jessenberger. *Statistische Methoden zur Qualitätssicherung und -optimierung in der Industrie.* Wiley-VCH, Weinheim, 1999.

C. Weihs, M. C. Röhl, and W. Theis. Multivariate classification of business phases. Technical report, SFB 475, University of Dortmund, 1999. 26/99.

C. Weihs and U. M. Sondhauss. Business phase classification and prediction: How to compare interpretability of classification methods? Technical report, SFB 475, University of Dortmund, 2000. 24/00.

D. H. Wolpert. A rigorous investigation of 'evidence' and 'Occam factors' in Bayesian reasoning. Technical Report 92-03-013, The Santa Fe Institute, Santa Fe, 1992.