Active Learning with Kernel Machines

Klaus Brinker

Dissertation

in Computer Science

submitted to the

Faculty of Electrical Engineering,
Computer Science and Mathematics
University of Paderborn

in partial fulfillment of the requirements for the degree of

doctor rerum naturalium
(Dr. rer. nat.)

Paderborn, November 2004

To my wife Kerstin.

Contents

Acknowledgements

I would like to thank the *International Graduate School of Dynamic Intelligent Systems* at the University of Paderborn, initiated and supported by the *Ministry of Education, Science and Research* of the state North Rhine-Westphalia, for establishing a stimulating and supportive environment for research, and for providing financial support in the form of a scholarship.

I would like to express my gratitude to my supervisor Professor H. Kleine Büning for creating a unique and productive research atmosphere. Being a member of his research group *Knowledge-Based Systems*, I benefitted from his precise advice in many respects. I am also indebted to my co-supervisor Professor E. Hüllermeier for fruitful and inspiring discussions and especially for introducing me to the field of preference learning. Thanks also go to Professor F. Meyer auf der Heide and Professor J. Fürnkranz for their readiness to review the present thesis.

I enjoyed extended discussions going far beyond the scope of a scientific corporation with my colleague and friend Chuan-Kang Ting. Moreover, he contributed to the present thesis by thoroughly proofreading the manuscript. I am indebted to Julia Schlüter for carefully proofreading the English text leading to many improvements in the exposition.

During the process of writing this thesis, my wife Kerstin patiently listened to me deliberating about artificial learning processes though her primary focus of interest is on learning and teaching strategies in practice. I could always rely on her encouragement and caring support in general and exquisite culinary support in particular.

Finally, I would like to express my deepest gratitude and appreciation to my parents. They have accompanied all my studies with their warm and constant support and, without doubt, laid the foundations of all my work.

Klaus Brinker

Paderborn, December 2004

Abstract

The ability to learn from experience is a fascinating and distinguishing feature of intelligent life. In the course of evolution, increasingly more sophisticated strategies of adapting behavior to particular surrounding conditions have emerged. Inspired by the efficiency of learning in nature, supervised machine learning considers a formal model of learning specific input-output relationships and adopts the paradigm of *learning from examples* to induce functions which predict target objects associated with input patterns. In conventional machine learning, the process of learning is characterized by a unidirectional information flow between the two idealized protagonists, i.e., information is submitted only from the teacher to the learner. In contrast to this, natural learning processes typically are based on complex interactions between the learner and the teacher. Bidirectional communication - from learners to the teaching entity and vice versa - is a fundamental component of learning. As motivated by nature, active machine learning substitutes the conventional passive model by an extended model which incorporates a restricted form of interactive learning in order to expedite the artificial learning process. We study the concept of active learning, which facilitates the learning process by reducing the amount of representative examples required, in the prospering field of kernel machines for various categories of learning problems.

A comprehensive corpus of flexible techniques for constructing prediction systems with excellent generalization capabilities has evolved as the constructive outcome from the inspiring synergy of statistical learning theory, optimization theory and more applied areas of research in machine learning in recent years. Among the variety of methods which have been developed, support vector machines are the most popular. With increasing computational power being available today, optimized training algorithms are able to cope with large-scale learning problems. However, advances in computational speed and more efficient training algorithms do not solve the inherent problem which consists in the fact that conventional supervised machine learning relies on a set of input patterns which have to be assigned to the corresponding target objects. In many areas of application, the task of assigning target objects cannot be accomplished in an automatic manner, but depends on time-consuming and expensive resources, such as complex experiments or human decisions. Hence, the underlying assumption that a set of *labeled* examples is submitted to the learning algorithm disregards the labeling effort that is necessary in many cases.

The superordinate concept of active learning refers to a collection of approaches which aim at reducing the labeling effort in supervised machine learning. We consider the pool-based active learning model, where the essential idea is to select promising *unlabeled* examples from a given finite set in a sequential process in the sense that

the corresponding target objects contribute to a more accurate prediction function. In contrast to conventional supervised learning, pool-based active learning considers an extended learning model in which the learning algorithm is granted access to a set of initially unlabeled examples and provided with the ability to determine the order of assigning target objects with the objective of attaining a high level of accuracy without requesting the complete set of corresponding target objects.

The present thesis pursues the general objectives of improving and generalizing existing approaches in pool-based active learning with kernel machines to a broader field of learning problems and of providing a thorough analysis of underlying theoretical aspects. More precisely, our main contributions can be summarized as follows:

- *Improvement of Efficiency:* In the context of binary classification learning, we propose a novel strategy for the selection of batches of multiple examples. As the fundamental component of our approach, we incorporate a measure of diversity to provide a more efficient strategy in terms of the number of labeled examples necessary to attain a particular level of classification accuracy. Moreover, we suggest modified multiclass selection strategies for different binary decomposition methods which yield substantial improvements over previous research.
- *Generalization:* Label ranking forms a category of preference learning problems which has not been investigated in active learning research previously, despite the fact that the labeling effort is an even more essential matter of relevance here than in classification learning. Employing the constraint classification and the pairwise ranking techniques, we propose generalizations of pool-based active learning and demonstrate a substantial improvement of the learning progress.
- *Theoretical Foundations:* We investigate a linear learning setting to analyze theoretical drawbacks of volume-based selection strategies and show that the minimization of the volume of the version space can be viewed as a *necessary* precondition for the minimization of the proposed improved selection criterion. Moreover, we prove a novel convergence theorem for the volume-based SIMPLE selection strategy in the case of the maximum radius ball approximation.

To this end, we present a general view of the concept of supervised machine learning and formalize the considered learning model in order to establish a common basis in terms of notation. We focus on the hypothesis class of linear classifiers which can be extended by the elegant and flexible concept of kernels. Among the variety of kernel classifiers, support vector machines and Bayes point machines are well-studied examples of linear learning algorithms which exhibit an appealing geometric interpretation in the so-called version space model. Moreover, linear classifiers serve as fundamental components in solving problems of more complex categories in supervised learning. In order to improve numerical stability and accuracy, we propose modifications to a kernel billiard algorithm for approximating the Bayes point.

We embark on a detailed presentation of the class of active learning selection strategies which aim at reducing the volume of the version space by means of approximating the center of mass and discuss the underlying theoretical line of reasoning. Moreover, we analyze a linear learning setting where volume-based strategies exhibit some potential shortcomings and propose an improved binary selection strategy to address this problem. In order to reduce the computational complexity of the envisaged selection strategy, we present a sophisticated subsampling technique which preselects a subset of unlabeled examples according to a less demanding volume-based strategy as candidates for our novel selection strategy. From a theoretical point of view, the minimization of the volume of the version space is a *necessary* precondition for the minimization of the improved selection criterion. As we anticipated on account of our theoretical analysis, experimental results demonstrate that the two-layered subsampling approach substantially decreases the computational complexity without a concomitant loss of classification accuracy. Apart from practical issues, the considered setting sheds light on potential shortcomings of volume-based strategies from a theoretical perspective.

The basic course of action in pool-based active learning consists in the sequential process of selecting *individual* examples from a set of unlabeled examples and requesting the corresponding target objects. However, both from a computational point of view and, more significantly, with regard to common characteristics of learning problems in practice, generalizing this scheme such that *sets* of unlabeled examples are selected and submitted to the labeling component yields beneficial effects. We propose a generalized selection strategy which incorporates a measure of diversity in the active selection of batches of multiple examples in binary classification learning. Furthermore, we present experimental results indicating that this novel approach provides a more efficient method in terms of the number of labeled examples necessary to attain a particular level of classification accuracy than a commonly employed extension of a volume-based selection strategy.

While most research on active learning in the field of kernel machines has focused on binary problems, less attention has been paid to the problem of learning classifiers in the case of multiple classes. We consider three common decomposition methods for expressing multiclass problems in terms of sets of binary classification problems and propose novel active learning selection strategies in order to reduce the labeling effort. A variety of experiments conducted on real-world datasets demonstrates the merits of our approach in comparison to previous research.

The effort necessary to construct sets of labeled examples in a supervised learning scenario is often disregarded, though in many applications, it is a time-consuming and expensive procedure. While the process of labeling constitutes a major issue in classification learning, it becomes an even more substantial matter of relevance in label ranking learning, which considers the more complex target domain of total orders over a fixed set of alternatives. We introduce a novel generalization of pool-based active learning to reduce the labeling effort based on both the pairwise ranking and the constraint classification techniques for representing label ranking functions.

The final part of this thesis is devoted to a more thorough theoretical analysis of volume-based selection strategies in binary classification learning. We derive a convergence theorem for a common volume-based selection strategy in the case of the maximum radius ball approximation of the center of mass and present a survey of related results.

I FUNDAMENTAL CONCEPTS

1 Introduction

Overview

This introductory chapter recapitulates the general concept of supervised machine learning and motivates the pool-based active learning framework from a practical point of view. Moreover, we will formalize the general learning model to be considered and establish a common basis in terms of notation. Finally, we will outline the structure of this thesis.

1.1 Supervised Machine Learning

A fundamental concept in machine learning is the supervised learning scenario.[1] The basic objective is to learn a prediction function from a given input space $\mathcal{X}$ to the space of target objects $\mathcal{Y}$. More precisely, we seek to induce a mapping $h : \mathcal{X} \to \mathcal{Y}$ based on a representative training set of examples $\{(x_1, y_1), \ldots, (x_m, y_m)\} \subset \mathcal{X} \times \mathcal{Y}$ consisting of pairs of input patterns and associated target objects for the accurate prediction of target objects for unseen patterns.

Classification Learning

Depending on the given target space $\mathcal{Y}$, we differentiate between various categories of learning problems: Classification learning considers unordered target spaces of finite cardinality, which we assume to be given by $\mathcal{Y} = \{1, \ldots, d\}$. In the case of a binary target space (binary classification), we depart from this notational convention for mathematical convenience and assume that $\mathcal{Y} = \{-1, +1\}$. To point up the difference to binary classification, we refer to classification problems with $|\mathcal{Y}| > 2$ as multiclass classification problems. Beyond classification learning, we consider the more complex target space of total orders over a finite set of alternatives in this thesis. The target space of all permutations of the set of alternatives $\{1, \ldots, d\}$ is isomorphic to the symmetric group of degree d, i.e., $\mathcal{Y} = \mathcal{S}^{(d)}$. This setting is referred to as label ranking learning and belongs to the category of preference learning. The aforementioned categories cover an important part of the complete spectrum of supervised learning problems. Yet, there are categories of learning problems, such as regression learning ($\mathcal{Y} = \mathbb{R}$) and alternative approaches to preference learning, which are not in scope of the present thesis.

Label Ranking Learning

Arguably being the most simple nontrivial supervised learning problem, binary classification is a fundamental and well-studied learning problem which can serve as a starting point to investigate more complex domains. Let us consider a medical diag-

1. See (Mitchell, 1997) for a general introduction to machine learning.

nosis scenario, which can be cast as a binary classification problem in a straightforward manner, to illustrate this setting more concretely: Assume we are given a finite set of patients which are represented by input patterns consisting of certain features (such as age, weight, medical test results, etc.) and their corresponding diagnoses of the particular disease to be tested ($+1 \,\hat{=}$ infected, $-1 \,\hat{=}$ not infected). The learning task consists in inducing a function from patients to the binary set of diagnoses based on the representative training set of examples for the accurate prediction of diagnoses for unseen patients, i.e., patients not included in the training set. A common measure to specify the so-far vague notion of accurate prediction is to consider the probability of correct classification on unseen examples. While there are more complex measures of accuracy to capture various properties of particular classification learning problems, such as asymmetric class costs, we will focus on the probability of correct classification. Alternatively, we will refer to this measure as classification accuracy.

Binary Classification Example

In order to establish a common basis in terms of notation, we proceed to a formal presentation of the basic learning model to be considered in the following: We assume that examples (x, y) are drawn independently and identically distributed according to some underlying probability distribution $\mathrm{P}^{(\mathrm{X},\mathrm{Y})}$ on $\mathcal{X} \times \mathcal{Y}$ where X and Y denote the corresponding random variables on the input space $\mathcal{X}$ and the target space $\mathcal{Y}$, respectively. In particular, this model is able to represent probabilistic dependencies between input patterns and target objects in the case that there does not exist a deterministic mapping from every input pattern to a particular target object due to the nature of the given learning domain or noisy data. In contrast to this, the probably approximately correct (PAC) concept learning model (Valiant, 1984) considers the more restrictive assumption that there exists a deterministic mapping which generates the corresponding target objects.

Formal Learning Model

In the standard manner, we define a general framework for measuring the accuracy of prediction based on the notion of loss function and risk:

Evaluation Measure

Definition 1.1 Expected and Empirical Risk
Assume we are given a loss function $l : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ and a probability distribution $\mathrm{P}^{(\mathrm{X},\mathrm{Y})}$ on $\mathcal{X} \times \mathcal{Y}$. Then, the expected risk of a prediction function $h : \mathcal{X} \to \mathcal{Y}$ is defined as

$$\mathrm{R}(h) \overset{def}{=} \int_{\mathcal{X} \times \mathcal{Y}} l(y, h(x))\, d\mathrm{P}^{(\mathrm{X},\mathrm{Y})}. \tag{1.1}$$

Moreover, given a set of labeled examples $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ drawn independently and identically distributed according to $\mathrm{P}^{(\mathrm{X},\mathrm{Y})}$, the empirical risk of h is defined as

$$\mathrm{R}_{\mathrm{emp}}(h) \overset{def}{=} \frac{1}{m} \sum_{i=1}^{m} l(y_i, h(x_i)). \tag{1.2}$$

In classification learning, a common choice of loss function is the so-called 0-1-loss function which is defined as

$$l_{0-1} : \mathcal{Y} \times \mathcal{Y} \to [0, \infty) \tag{1.3}$$

$$(y, y') \mapsto \begin{cases} 0 & \text{if } y = y', \\ 1 & \text{otherwise.} \end{cases} \tag{1.4}$$

While the notion of risk allows for more sophisticated cost models through the choice of appropriate loss functions, we will restrict our discussion to the 0-1-loss function in classification learning. In this case, the expected risk evaluates to the probability of misclassification with respect to $\mathrm{P}^{(\mathrm{X},\mathrm{Y})}$ and the empirical risk evaluates to the empirical classification error.

Classification Accuracy

Moreover, the expected and the empirical classification accuracy relate to the notion of risk by $1 - \mathrm{R}(h)$ and $1 - \mathrm{R}_{\mathrm{emp}}(h)$, respectively. For general loss functions, $1 - \mathrm{R}(h)$ is referred to as generalization accuracy. If the empirical risk is evaluated with respect to the training set, then we refer to the empirical classification error as training error. Moreover, the test error is defined as the empirical risk with respect to an independently and identically distributed set of examples which has not been used for training the prediction function h. The probability of misclassification can be approximated by the test error as theoretically justified by the law of large numbers. Training and test accuracy are defined in an analogous manner.

We can concretize the fundamental objective in our learning model as finding a classifier $h : \mathcal{X} \to \mathcal{Y}$ minimizing the expected risk. In general, $\mathrm{P}^{(\mathrm{X},\mathrm{Y})}$ is assumed to be unknown and therefore the corresponding expected risk can neither be evaluated nor be minimized directly. Instead, we are given an independently and identically distributed training set of patterns along with the correct[2] target objects $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ as the sole source of information about the learning problem for the induction of a prediction function h.

1.2 Active Learning

With increasing computational power being available today, optimized training algorithms are able to cope with large-scale learning problems involving tens of thousands of training examples. However, advances in computational speed and more efficient training algorithms do not solve the inherent problem which consists in the fact that

Labeling Problem

conventional supervised machine learning relies on a set of patterns which have to be assigned to the correct target objects. In many applications, the task of assigning target objects cannot be accomplished in an automatic manner, but depends on time-consuming and expensive resources, such as complex experiments or human decisions. Hence, the assumption that a set of labeled examples is submitted to the

2. The term *correct* refers to the target object of a given realization of the random variable $\mathrm{Z} = (\mathrm{X}, \mathrm{Y})$. Note that this does not imply the existence of a deterministic relationship between input patterns and target objects as mentioned above.

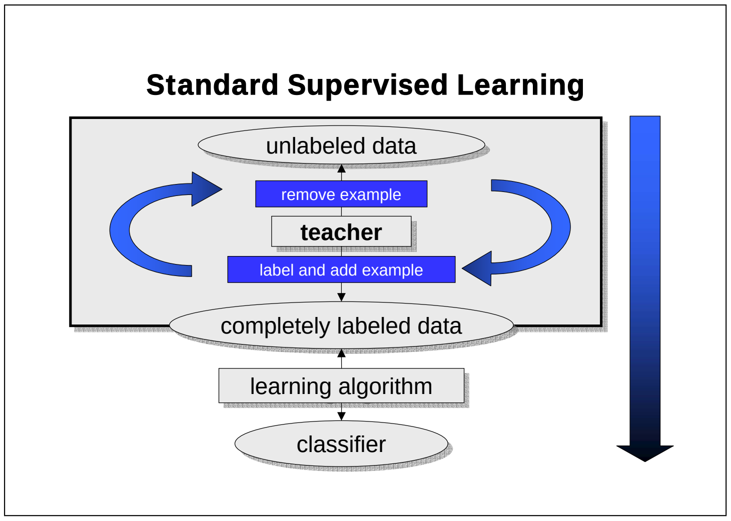


Figure 1.1 *In standard supervised machine learning, the labeling effort necessary to assign input patterns to target objects is disregarded.*

learning algorithm disregards the labeling effort that is necessary in many cases.

Active Learning

The superordinate concept of active learning refers to a collection of approaches which aim at reducing the labeling effort in supervised machine learning (see Chapter 3 for a more detailed presentation). We consider the pool-based active learning model, which was originally introduced by Lewis and Gale (1994) in the context of text classification learning. If not noted otherwise, we refer to the *pool-based active learning model* as *active learning* herein after to simplify our presentation. The essential idea behind active learning is to select promising patterns from a given finite set $\mathcal{U}$ (also referred to as the pool of unlabeled examples) in a sequential process in the sense that the corresponding target objects contribute to a more accurate prediction function. The active learning algorithm sequentially selects patterns from the set $\mathcal{U}$ and requests the corresponding target objects from a teacher component (also referred to as oracle). In contrast to standard supervised learning, pool-based active learning considers an extended learning model in which the learning algorithm is granted access to a set of unlabeled examples and provided with the ability to determine the order of assigning target objects with the objective of attaining a high level of accuracy without requesting the complete set of corresponding target objects. Moreover, we have to establish a stopping criterion which can either be of dynamic nature and depend on a measure of the learning progress or be of static nature such as a fixed number of requested target objects.

In particular, text classification is a characteristic learning problem which is amenable to the active learning approach: While there is a relatively cheap source of unlabeled examples, acquiring associated target labels is an expensive procedure. Large corpora of text documents are readily available in many domains and especially the world wide web is a comprehensive source of textual information. However, assigning given text

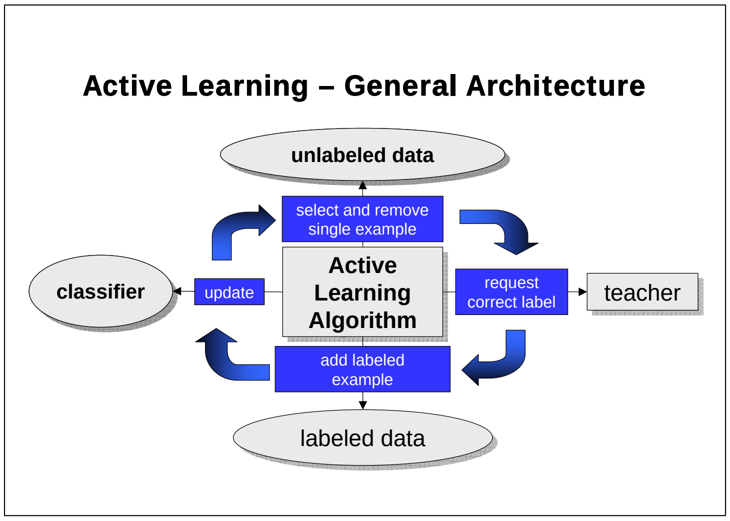

Active Learning – General Architecture
unlabeled data
select and remove single example
classifier
update
Active Learning Algorithm
request correct label
teacher
add labeled example
labeled data

Figure 1.2 *Pool-based active learning considers an extended learning model in which the learning algorithm is granted access to the set of unlabeled examples and provided with the ability to determine the order of assigning target objects.*

documents to target categories, such as *relevant/not relevant* or *politics, sports and economy*, to generate labeled training sets is a time-consuming task.[3] This general pattern is not only characteristic for text classification problems, but also arises in many other domains. In the process of chemical drug discovery, computational chemistry methods can readily generate large amounts of compounds (unlabeled examples), but it requires expensive biological testing to determine whether a compound binds to a particular target molecule.[4] Machine learning is a successful approach for building highly accurate pixel-based filters which discriminate different categories of clouds on satellite images. While images can be readily converted into hundreds of thousands of unlabeled pixel patterns using a suitable neighborhood definition, the procedure of assigning correct cloud types requires the expensive expertise of a meteorologist.

Applications of Active Learning

We have argued that the problem of how to cope with the labeling effort in supervised machine learning arises naturally in many fields of application. The crucial point in active learning is that by ordering the sequential process of requesting target objects with respect to an appropriate measure of the information content, it is possible to reduce the labeling effort. In many applications, active learning achieves the same level of accuracy as standard supervised learning, which is based on the entire set of labeled examples, while only requesting a fraction of all the target objects.

3. See the following references for applications of active learning to text classification (Tong and Koller, 2001c; McCallum and Nigam, 1998; Roy and McCallum, 2001).
4. See (Warmuth et al., 2003) for an active learning study on this domain.

1.3 Overview of Thesis

The present thesis consists of two principal parts: The first part, comprising Chapters 1 and 2, provides an introduction to the concepts and models which form the basis of our research. The second part, which contains Chapters 3 to 7, examines various aspects of pool-based active learning with kernel machines.

In the present chapter, we recapitulated the general concept of supervised machine learning and established a common basis in terms of notation. Furthermore, the pool-based active learning framework was introduced and motivated from a less formal perspective. The subsequent chapter focuses on the class of linear classifiers and discusses the powerful and elegant extension by the concept of kernels. In particular, support vector machines and Bayes point machines are well-studied examples of linear learning algorithms which can be extended to powerful and flexible classifiers by this means. In order to improve numerical stability and accuracy, we propose modifications to a kernel billiard algorithm for approximating the Bayes point. Moreover, Chapter 2 discusses the version space model in the context of binary classification learning, where both support vector machines and Bayes point machines exhibit an appealing geometric interpretation.

The second part of this thesis proceeds to the field of active learning. In Chapter 3, we embark on a detailed presentation of principal models and categorize various approaches to active learning. The pool-based active learning model, which forms the theoretical basis for both binary classification learning and subsequent generalizations to multiclass and label ranking learning in this thesis, is discussed from a more formal point of view.

A class of selection strategies which aims at reducing the volume of the version space by means of approximating the center of mass is reviewed in Chapter 4. Furthermore, in this chapter, we analyze a linear learning setting where these approaches exhibit some potential shortcomings and propose an improved binary selection strategy to address this problem. In order to reduce the computational complexity of the envisaged selection strategy, we present a sophisticated subsampling technique which preselects a subset of unlabeled examples according to a less demanding volume-based strategy as candidates for our novel selection strategy.

Chapter 5 considers a generalized active learning setting where, instead of individual examples, sets of unlabeled examples are selected in the sequential course of action. We propose a generalized selection strategy which incorporates a measure of diversity in the active selection of batches of multiple patterns and present experimental results indicating that this novel approach provides a more efficient method in terms of the number of labeled examples necessary to attain a particular level of classification accuracy than a commonly used extension of the SIMPLE volume-based selection strategy.

While most research on active learning in the field of kernel machines has focused on binary problems, less attention has been paid to the problem of learning classifiers in the case of multiple classes. In Chapter 6, we consider different decomposition methods for expressing multiclass problems in terms of sets of binary classification problems and propose novel active learning selection strategies in order to reduce the labeling effort. Various experiments conducted on real-world datasets demonstrate the merits of our approach in comparison to previous research.

Chapter 7 considers the more complex target domain of total orders over a fixed set of alternatives where the procedure of labeling input patterns constitutes an even more substantial matter of relevance than in classification learning. We introduce a novel generalization of pool-based active learning to reduce the labeling effort based on both the pairwise ranking and the constraint classification techniques for representing label ranking functions.

The final chapter embarks on a more thorough theoretical analysis of volume-based selection strategies in pool-based active learning. We derive a novel convergence theorem for the common SIMPLE volume-based selection strategy in the case of the maximum radius ball approximation of the center of mass and present a survey of related results.

Bibliographical Note: A subset of the results presented in this thesis has been published in (Brinker, 2003a,b,c, 2004a,b).

2 Kernel Machines

The introductory chapter presented a general view of the concept of supervised machine learning. In the present chapter, we focus our presentation on the hypothesis class of linear classifiers and discuss a powerful and elegant extension by the concept of kernels. The subsequent sections are restricted to binary classification learning, however, later on linear classifiers will serve as the fundamental underlying component in solving problems of more complex categories in supervised learning.

Overview

The concept of kernels provides a general method for modeling the notion of similarity between patterns and can be view as an elegant and efficient means of embedding input patterns into Euclidean feature spaces, where dot products allow us to calculate geometric quantities such as distances and angles. In particular, support vector machines and Bayes point machines are well-studied examples of linear learning algorithms which can be extended to powerful and flexible classifiers by this means. In order to improve numerical stability and accuracy, we propose modifications to a kernel billiard algorithm for approximating the Bayes point. Moreover, this chapter discusses the version space model in the context of binary classification learning, where both support vector machines and Bayes point machines exhibit an appealing geometric interpretation.

2.1 Linear Classifiers

In this thesis, we focus on the hypothesis class of linear classifiers to solve both classification problems and problems of more complex categories in supervised machine learning. More precisely, let us consider the finite-dimensional, real Euclidean vector space $\mathbb{R}^N$ and proceed to a formal definition of the notion of linear function and linear classifier. While we restrict our view to the case $\mathcal{X} = \mathbb{R}^N$ to provide a more intuitive presentation in the present section, we will consider a more general definition by incorporating the concept of kernels later on.

Definition 2.1 Linear Function and Linear Classifier
Suppose we are given the input space of patterns $\mathcal{X} = \mathbb{R}^N$ (with $N \in \mathbb{N}$) endowed with the canonical dot product $\langle \cdot, \cdot \rangle$, i.e., for any $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$: $\langle \mathbf{x}, \mathbf{x}' \rangle \overset{def}{=} \sum_{i=1}^{N} [\mathbf{x}]_i [\mathbf{x}']_i$ where the componentwise representations are denoted by $\mathbf{x} = ([\mathbf{x}]_1, \ldots, [\mathbf{x}]_N)^\top$ and $\mathbf{x}' = ([\mathbf{x}']_1, \ldots, [\mathbf{x}']_N)^\top$, respectively. Then, for any $\mathbf{w} \in \mathcal{X}$ we define a corresponding

linear function $f_w : \mathcal{X} \to \mathbb{R}$ *by*

$$f_w(\mathbf{x}) \overset{def}{=} \langle \mathbf{w}, \mathbf{x} \rangle = \sum_{i=1}^{N} [\mathbf{w}]_i \, [\mathbf{x}]_i . \tag{2.1}$$

Denoting the binary target space by $\mathcal{Y} = \{-1, +1\}$, *we define a linear classifier* $h_w : \mathcal{X} \to \mathcal{Y}$ *by thresholding the real-valued output of* f_w *at zero:*

$$h_w(\mathbf{x}) \overset{def}{=} \text{sign}(f_w(\mathbf{x})) = \text{sign}\left(\langle \mathbf{w}, \mathbf{x} \rangle\right) \tag{2.2}$$

with $\text{sign}(t) \overset{def}{=} +1$ *for* $t > 0$ *and* $\text{sign}(t) \overset{def}{=} -1$ *otherwise. We refer to* $\mathbf{w}$ *as the weight vector of the linear function* f_w *and the linear classifier* h_w*, respectively.*

Classification Boundary

For a given linear classifier h_w, the classification boundary between the positive and the negative class is the hyperplane $\{\mathbf{x} \in \mathbb{R}^N \mid \langle \mathbf{w}, \mathbf{x} \rangle = 0\}$ with normal vector $\mathbf{w}$ (see Figure 2.1). Note that rescaling the weight vector with a positive constant does not have an effect on the classification outcome of the corresponding linear classifier. Therefore, if we impose a suitable normalization constraint on the set of weight vectors, such as normalization to unit length, we do not effectively restrict the set of corresponding linear classifiers.

Definition 2.2 Hypothesis Space of Linear Classifiers
Suppose the input space and the target space are given by $\mathcal{X} = \mathbb{R}^N$ *and* $\mathcal{Y} = \{-1, +1\}$, *respectively. The hypothesis space of weight vectors (weight space) is defined as*

$$\mathcal{W} \overset{def}{=} \{\mathbf{w} \in \mathcal{X} \mid \|\mathbf{w}\| = 1\} \tag{2.3}$$

where $\|\cdot\|$ *denotes the 2-norm in* $\mathbb{R}^N$, *i.e., for any* $\mathbf{w} = ([\mathbf{w}]_1, \ldots, [\mathbf{w}]_N)^\top \in \mathbb{R}^N$: $\|\mathbf{w}\| \overset{def}{=} \sqrt{\langle \mathbf{w}, \mathbf{w} \rangle} = \sqrt{\sum_{i=1}^N [\mathbf{w}]_i^2}$. *Furthermore, we define the hypothesis space of linear classifiers in* $\mathcal{X}$ *as*

$$\mathcal{H} \overset{def}{=} \{h_w : \mathcal{X} \to \mathcal{Y} \mid \mathbf{w} \in \mathcal{W}\} \tag{2.4}$$

where linear classifiers h_w *are defined as in* (2.2).

Isomorphism to Hypersphere

As a consequence of the normalization of weight vectors to unit length, we can state a canonical isomorphism, $\mathbf{w} \mapsto h_w$, between the hypothesis weight space $\mathcal{W}$ (the unit hypersphere in $\mathcal{X}$) and the hypothesis space of linear classifiers $\mathcal{H}$. Therefore, in the following, we will identify a normalized weight vector $\mathbf{w}$ with the corresponding linear classifiers h_w, if appropriate.

2.2 Kernel Feature Map and Feature Space

We restricted our presentation to the real Euclidean vector space $\mathbb{R}^N$ in the previous section to provide a more intuitive and less technical presentation of linear classifiers. However, various learning problems require a more flexible setting to become amenable

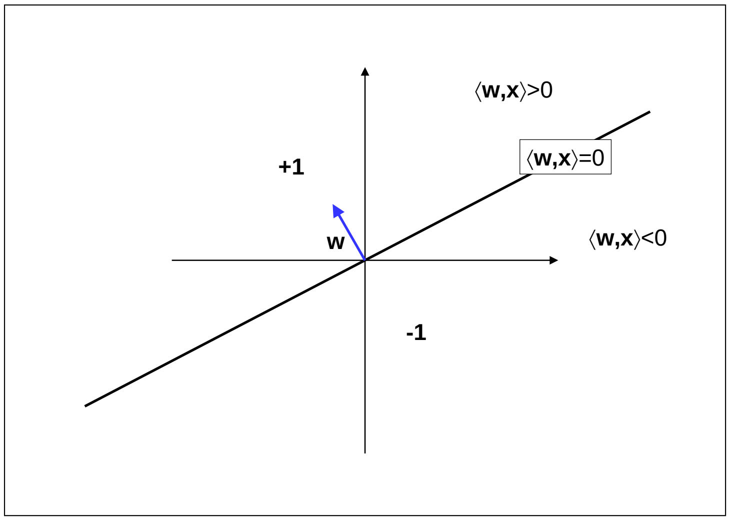


Figure 2.1 *Linear classifier dividing the plane into two halfspaces which are assigned to the positive and the negative class, respectively.*

to learning linear classifiers. First of all, whenever input patterns are not represented as real-valued vectorial data, e.g., variable-length string representations of DNA sequences are commonly used in bioinformatics[1], the above-defined linear setting is not applicable. Furthermore, even though a given problem does have a natural representation as vectorial data in the input space $\mathbb{R}^N$, linear classifiers may not be able to model a decision boundary for the *accurate* prediction of target class labels (see Figure 2.2).

In order to generalize the linear setting and circumvent the aforementioned drawbacks, we employ an embedding of input patterns $x \in \mathcal{X}$ into a suitable real Euclidean feature space $\mathcal{F}$ via a map $\phi : \mathcal{X} \to \mathcal{F}$ where linear classifiers are well-defined. Furthermore, a broad class of learning algorithms can be stated such that they depend on input patterns only in terms of dot products. Therefore, it is sufficient to compute dot products of embedded patterns, $\langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$ with $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ denoting a dot product in $\mathcal{F}$, in order to conduct the training process. We will demonstrate that combining the operations of mapping patterns and calculating dot products in feature space is much more efficient in various practically relevant cases where a direct stepwise approach is infeasible since the dimension of the feature space is high or even infinite.

To be more formal, we have to study the question of which functions $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ admit the construction of a map into some real Euclidean feature space $\mathcal{F}$ such that

1. See (Baldi and Brunak, 2001) for an introduction to bioinformatics from a machine learning point of view.

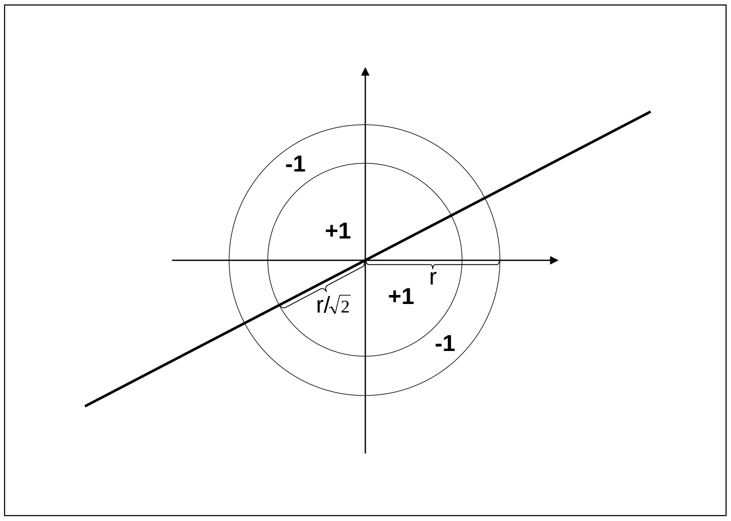


Figure 2.2 *Assume that input patterns are uniformly distributed within the larger disk of radius r. Fixing the radius of the inner disk to $\frac{r}{\sqrt{2}}$, the probability mass of the inner disk and the outer torus are equal. Furthermore, assume that input patterns within the inner disk are assigned to class $+1$ and class -1 otherwise, i.e.,* $\mathrm{P}(\mathrm{Y}=1 \mid \|\mathrm{X}\| \leq \frac{r}{\sqrt{2}}) = \mathrm{P}(\mathrm{Y}=-1 \mid \frac{r}{\sqrt{2}} < \|\mathrm{X}\| \leq r) = 1$. *Then, any linear classifier has a misclassification probability of* 0.5.

they can be represented as

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}. \tag{2.5}$$

This question is sufficiently answered by the following theorem, which originated from the study of integral operators in functional analysis. We state a version of Mercer's theorem given in (Schölkopf and Smola, 2002; Williamson et al., 2001):

Theorem 2.3 Mercer Kernel Map (Mercer, 1909)
Assume we are given a finite measure space $(\mathcal{X}, \mu)$. Let us denote by $L_\infty(\mathcal{X}^2)$ the space of μ-integrable functions on $\mathcal{X}^2$ with respect to the ∞-norm and by $L_2(\mathcal{X})$ the space of μ-integrable functions on $\mathcal{X}$ with respect to the 2-norm. Suppose $k \in L_\infty(\mathcal{X}^2)$ is a symmetric real-valued function such that the integral operator

$$T_k : L_2(\mathcal{X}) \to L_2(\mathcal{X}) \tag{2.6}$$

$$(T_k f)(x) \overset{def}{=} \int_{\mathcal{X}} k(x, x') f(x') \, d\mu(x') \tag{2.7}$$

is positive semidefinite; that is, for all $f \in L_2(\mathcal{X})$, we have

$$\int_{\mathcal{X}^2} k(x, x') f(x) f(x') \, d\mu(x) d\mu(x') \geq 0. \tag{2.8}$$

Let $\psi_i \in L_2(\mathcal{X})$ denote the normalized orthogonal eigenfunctions of T_k associated with the eigenvalues $\lambda_i > 0$, sorted in non-increasing order. Then,

1. $(\lambda_i)_{i=1,\ldots,N_\mathcal{F}} \in l_1^{N_\mathcal{F}}$ *where $l_1^{N_\mathcal{F}}$ denotes the space of absolutely convergent sequences of length $N_\mathcal{F}$ and $N_\mathcal{F} \in \mathbb{N} \cup \{\infty\}$ denotes the dimension of the Mercer kernel feature space $\mathcal{F}$.*

2. $k(x, x') = \sum_{i=1}^{N_\mathcal{F}} \lambda_i \psi_i(x) \psi_i(x')$ *holds for almost all (x, x'). In the case $N_\mathcal{F} = \infty$, the series converges absolutely and uniformly for almost all (x, x').*

Conversely, any integral operator kernel k admitting a uniformly convergent dot product representation on some compact set $\mathcal{X} \times \mathcal{X}$,

$$k(x, x') = \sum_{i=1}^{\infty} \psi_i(x) \psi_i(x'), \tag{2.9}$$

is positive semidefinite.

Remark 2.4
Condition (2.8) *can be substituted by the equivalent condition that for any finite subset $\{x_1, \ldots, x_m\} \subset \mathcal{X}$, the corresponding $m \times m$ kernel matrix K with $K_{ij} \overset{def}{=} k(x_i, x_j)$ is positive semidefinite.*

Let us emphasize the key point of Theorem 2.3: Any positive semidefinite kernel k defined on a nonempty set $\mathcal{X}$ corresponds to a dot product in the so-called Mercer kernel feature space $\mathcal{F} = l_2^{N_\mathcal{F}}$, where $l_2^{N_\mathcal{F}} \subset \mathbb{R}^{N_\mathcal{F}}$ denotes the space of square summable sequences of length $N_\mathcal{F}$ with $N_\mathcal{F} \in \mathbb{N} \cup \{\infty\}$, by $k(x, x') = \langle \phi(x), \phi(x') \rangle_\mathcal{F}$ where

$$\phi : \mathcal{X} \to \mathcal{F} \tag{2.10}$$

$$x \mapsto (\sqrt{\lambda_i} \psi_i(x))_{i=1,\ldots,N_\mathcal{F}} \tag{2.11}$$

denotes the so-called Mercer kernel map.

The Mercer kernel map is defined on the entire input space. However, the dimension of the corresponding Mercer kernel feature space can be infinite as in the case of RBF kernels defined on non-trivial input spaces (Schölkopf and Smola, 2002). In contrast to this, it is possible to construct a data-dependent kernel map and kernel feature space based on a given set of input patterns which is of finite dimension:

Theorem 2.5 Data-Dependent Kernel Map (Schölkopf, 1997)
Suppose we are given a finite set of input patterns $\{x_1, \ldots, x_m\} \subset \mathcal{X}$ and a kernel k such that the $m \times m$ kernel matrix K with $K_{ij} \overset{def}{=} k(x_i, x_j)$ is positive semidefinite. Then, there exists a map ϕ from $\{x_1, \ldots, x_m\}$ into the feature space $\mathcal{F} = \mathbb{R}^{N_\mathcal{F}}$ with $N_\mathcal{F} = \text{rank}(K) \leq m$ such that

1. *The kernel k corresponds to a dot product in $\mathcal{F}$ by*

$$k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle. \tag{2.12}$$

2. *$\mathcal{F}$ is the linear span of $\{\phi(x_1), \ldots, \phi(x_m)\}$.*

Proof Since K is positive semidefinite, we can perform a spectral decomposition into

$$K = SDS^\top \tag{2.13}$$

with S being an orthogonal $m \times m$ matrix and D being the diagonal $m \times m$ matrix consisting of the (nonnegative) eigenvalues $\lambda_1, \ldots, \lambda_m$ of K. We assume that the eigenvalues are in nonincreasing order such that

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r > 0 = \lambda_{r+1} = \cdots = \lambda_m \quad \text{with} \quad r = \text{rank}(K). \tag{2.14}$$

Note that if K is positive definite, then all eigenvalues are positive and K has full rank, hence, $r = m$.
With $s_1^\top, \ldots, s_m^\top$ denoting the rows of S, we define a map ϕ by

$$\phi : \{x_1, \ldots, x_m\} \to \mathbb{R}^r \tag{2.15}$$

$$x_i \mapsto \begin{pmatrix} \sqrt{\lambda_1}\,[s_i]_1 \\ \vdots \\ \sqrt{\lambda_r}\,[s_i]_r \end{pmatrix}. \tag{2.16}$$

Hence,

$$\begin{aligned} \langle \phi(x_i), \phi(x_j) \rangle &= \sum_{l=1}^{r} \sqrt{\lambda_l}\,[s_i]_l \sqrt{\lambda_l}\,[s_j]_l && (2.17)\\ &= \sum_{l=1}^{m} S_{il} \lambda_l (S^\top)_{lj} && (2.18)\\ &= (SDS^\top)_{ij} && (2.19)\\ &= K_{ij}. && (2.20) \end{aligned}$$

Since we have constructed the images $\phi(x_1), \ldots, \phi(x_m)$ such that they span $\mathbb{R}^r$, the proof is complete. ■

It is important to note that in contrast to the Mercer kernel feature space, which contains the entire image of the input space $\mathcal{X}$, the data dependent kernel feature space is restricted to the linear span of the embedded patterns, which in general is only a subset of the former.

Theorems 2.3 and 2.5 state that given a kernel k there exists a (data-dependent) kernel feature space $\mathcal{F}$ and a corresponding kernel feature map $\phi : \mathcal{X} \to \mathcal{F}$ such that k embeds input patterns into a real Euclidean vector space by $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$ where $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ denotes a dot product in $\mathcal{F}$. Thus giving rise to the following generalization of the hypothesis space of linear classifiers:

Definition 2.6 Hypothesis Space of Linear Classifiers in Feature Space
Assume that we are given a nonempty input space $\mathcal{X}$, the binary target space $\mathcal{Y} = \{-1, +1\}$, a kernel feature space $\mathcal{F}$ and the corresponding kernel feature map $\phi : \mathcal{X} \to \mathcal{F}$. Then, we define the weight space $\mathcal{W}$ of linear classifiers in the feature

space $\mathcal{F}$ in an analogous manner to Definition 2.2:

$$\mathcal{W} \overset{def}{=} \{\mathbf{w} \in \mathcal{F} \mid \|\mathbf{w}\|_{\mathcal{F}} = 1\} \tag{2.21}$$

where $\|\cdot\|_{\mathcal{F}}$ denotes the 2-norm in $\mathcal{F}$, i.e., for any $\mathbf{w} \in \mathcal{W}$: $\|\mathbf{w}\|_{\mathcal{F}} \overset{def}{=} \sqrt{\langle \mathbf{w}, \mathbf{w}\rangle_{\mathcal{F}}}$. Furthermore, the hypothesis space of linear classifiers in feature space $\mathcal{F}$ is defined as

$$\mathcal{H} \overset{def}{=} \{h_{\mathrm{w}} : \mathcal{X} \to \mathcal{Y} \mid \mathbf{w} \in \mathcal{W}\} \tag{2.22}$$

where for $\mathbf{w} \in \mathcal{W}$, h_{w} is defined as

$$h_{\mathrm{w}} : \mathcal{X} \to \mathcal{Y} \tag{2.23}$$

$$x \mapsto \mathrm{sign}(\langle \mathbf{w}, \phi(x)\rangle_{\mathcal{F}}). \tag{2.24}$$

Note that in the case of $\mathcal{X} = \mathbb{R}^N$, we can recover our initial Definition 2.2 by considering the identity map $\phi(\mathbf{x}) \overset{\mathrm{def}}{=} \mathbf{x}$ on $\mathcal{F} \overset{\mathrm{def}}{=} \mathbb{R}^N$ endowed with the canonical dot product. The remainder of this thesis will refer to this definition of the weight space and the hypothesis space of linear classifiers. In an analogous manner to the special case of $\mathcal{X} = \mathbb{R}^N$, we will identify both spaces considering the isomorphism $\mathbf{w} \mapsto h_{\mathrm{w}}$.

As we have discussed above, the concept of kernels offers an elegant and efficient way of embedding patterns into real Euclidean vector spaces. However, even if the input space is already endowed with a dot product, using kernels can be useful. Mapping patterns from the input space into a suitable feature space can simplify the task of discriminating examples in machine learning such that, for example, a linearly nonseparable problem in input space becomes linearly separable in feature space. From a different point of view, linear classifiers in the kernel-induced feature space correspond to nonlinear classification boundaries in input space (when using a nonlinear feature map).

Let us consider a simple two-dimensional problem where examples belonging to the negative class are contained in a disk centered at the origin and the remaining area is assigned to the positive class. Obviously, the two classes cannot be discriminated against each other by a hyperplane in input space. However, by lifting the points onto the surface of a paraboloid using the map

$$\begin{pmatrix} [\mathbf{x}]_1 \\ [\mathbf{x}]_2 \end{pmatrix} \mapsto \begin{pmatrix} [\mathbf{x}]_1 \\ [\mathbf{x}]_2 \\ [\mathbf{x}]_1^2 + [\mathbf{x}]_2^2 \end{pmatrix} \tag{2.25}$$

we obtain a problem in the feature space $\mathcal{F} = \mathbb{R}^3$ which can be separated by a hyperplane (see Figure 2.3).[2] In other words, this embedding renders a problem which is linearly separable if we allow for a bias term. More generally, linear classification boundaries in the feature space $\mathbb{R}^3$ correspond to nonlinear ellipsoidal classification boundaries in the input space $\mathbb{R}^2$.

2. In computational geometry, this map is employed to derive a test which determines if a test point lies inside, outside or on the circle defined by three points in the plane.

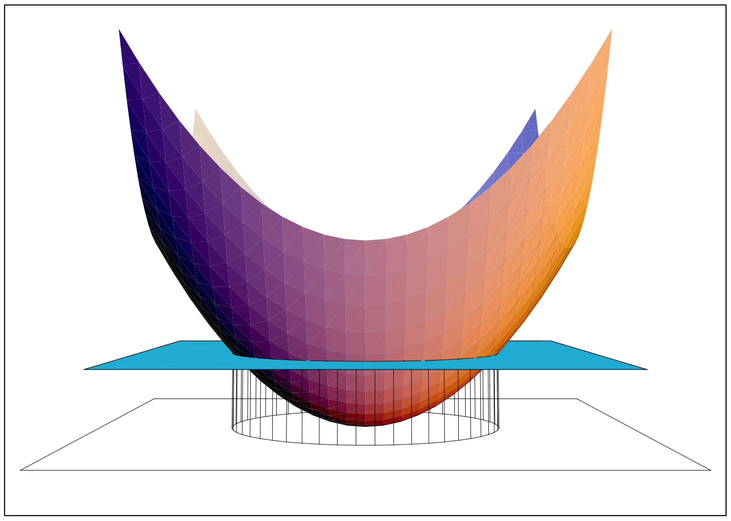


Figure 2.3 *By lifting points onto the surface of a paraboloid the two-dimensional problem of discriminating the interior of a circle against the remaining area becomes separable in three dimensions by a hyperplane.*

Similarly, we could have employed the map

$$\phi : \mathbb{R}^2 \to \mathbb{R}^3 \tag{2.26}$$

$$\begin{pmatrix} [\mathbf{x}]_1 \\ [\mathbf{x}]_2 \end{pmatrix} \mapsto \begin{pmatrix} [\mathbf{x}]_1^2 \\ [\mathbf{x}]_2^2 \\ \sqrt{2}\,[\mathbf{x}]_1[\mathbf{x}]_2 \end{pmatrix}, \tag{2.27}$$

which does not have such an intuitive geometric interpretation, to obtain a linearly separable problem. The corresponding kernel can be simplified to

$$k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle = \langle \mathbf{x}, \mathbf{x}' \rangle^2, \tag{2.28}$$

where the first dot product is computed in $\mathbb{R}^3$ and the second one in $\mathbb{R}^2$. More generally, homogeneous polynomial kernels,

$$k(\mathbf{x}, \mathbf{x}') = \langle \phi_{N,d}(\mathbf{x}), \phi_{N,d}(\mathbf{x}') \rangle^d, \tag{2.29}$$

correspond to feature maps

$$\phi_{N,d} : \mathbb{R}^N \to \mathbb{R}^{M_{N,d}} \tag{2.30}$$

$$\begin{pmatrix} [\mathbf{x}]_1 \\ \vdots \\ [\mathbf{x}]_N \end{pmatrix} \mapsto \left(c_{N,d} \prod_{i=1}^{N} [\mathbf{x}]_i^{[\mathrm{m}]_i} \right)_{\mathrm{m} \in \mathbb{N}^N,\, \sum_{i=1}^N [\mathrm{m}]_i = d} \tag{2.31}$$

with

$$c_{N,d} \overset{\text{def}}{=} \sqrt{\frac{d!}{\prod_{i=1}^{N} [\mathbf{m}]_i!}} \quad \text{and} \quad M_{N,d} \overset{\text{def}}{=} \binom{d+N-1}{d}, \tag{2.32}$$

thus, up to a scaling factor generating all monomial features of order d (Schölkopf and Smola, 2002). The computational advantage of using kernels instead of directly carrying out computations in feature space becomes obvious in this example: In order to compute a kernel value, we have to evaluate the dot product in an N-dimensional input space, whereas a direct stepwise computation involves dot products in a feature space of dimension $\binom{d+N-1}{d}$.

Embedding input patterns into a suitable feature space $\mathcal{F}$ provides a solution to the problem of linearly nonseparable data. In particular, any kernel k can be modified by adding some constant $\nu > 0$ to the diagonal elements of the kernel matrix, $K_{ij} \overset{\text{def}}{=} k(x_i, x_j) + \delta_{ij}\nu$, such that the given training set becomes linearly separable in feature space (Shawe-Taylor and Cristianini, 1999).[3]

Let us focus on the case $\mathcal{X} = \mathbb{R}^N$ for a moment. We have considered linear classifiers

$$h_{\mathrm{w}} : \mathcal{X} \to \mathcal{Y} \tag{2.33}$$

$$\mathbf{x} \mapsto \operatorname{sign}(f_{\mathrm{w}}(\mathbf{x})) = \operatorname{sign}(\langle \mathbf{w}, \mathbf{x} \rangle) \tag{2.34}$$

without bias so far. However, this restriction can be eliminated by embedding input patterns into the space $\mathbb{R}^{N+1}$ of one extra dimension by $\mathbf{x} \mapsto (\mathbf{x}, \tau)$ where $\tau \in \mathbb{R}$ is a fixed constant (Cristianini and Shawe-Taylor, 2000, pp. 130–131). Then, a linear classifier $h_{\mathrm{w},b}$ corresponding to the unit weight vector $\mathbf{w}$ and the bias term b,

$$h_{\mathrm{w},b} : \mathbb{R}^N \to \mathcal{Y} \tag{2.35}$$

$$\mathbf{x} \mapsto \operatorname{sign}(f_{\mathrm{w}}(\mathbf{x}) + b) = \operatorname{sign}(\langle \mathbf{w}, \mathbf{x} \rangle + b), \tag{2.36}$$

Linear Classifiers with Bias

is equivalent to the zero bias classifier $h_{\mathrm{w}'}$ with weight vector $\mathbf{w}' \overset{\text{def}}{=} (\mathbf{w}, b/\tau)$ in the augmented space $\mathbb{R}^{N+1}$ as a direct consequence of

$$f_{\mathrm{w}}(\mathbf{x}) + b = \langle \mathbf{w}, \mathbf{x} \rangle + b = \langle \mathbf{w}', \mathbf{x}' \rangle = f_{\mathrm{w}'}(\mathbf{x}') \tag{2.37}$$

where $\mathbf{x}' \overset{\text{def}}{=} (\mathbf{x}, \tau)$. Note that the first dot product is evaluated in $\mathbb{R}^N$ whereas the second dot product is evaluated in $\mathbb{R}^{N+1}$. Furthermore, it is possible to combine this modification with the kernelization of linear classifiers in a straightforward manner by adding τ^2 to the original kernel k:

$$k'(x, x') \overset{\text{def}}{=} k(x, x') + \tau^2. \tag{2.38}$$

Moreover, details on the appropriate choice of τ are discussed in (Cristianini and Shawe-Taylor, 2000, pp. 130–131).

3. δ_{ij} denotes the Kronecker delta function, which equals 1 if $i = j$ and 0 otherwise.

2.3 Version Space Model

Let us consider a linearly separable, binary classification problem in a kernel-induced feature space $\mathcal{F}$, i.e., we assume that there exists a linear classifier $h_{\mathrm{w}} : \mathcal{X} \to \{-1, +1\}$ such that $h_{\mathrm{w}}(x_i) = y_i$ for each training example (x_i, y_i) with $i = 1, \ldots, m$. As mentioned above, linearly nonseparable sets of examples can be incorporated into this setting in an elegant manner by a suitable modification of the kernel function. Then, the nonempty set

$$\mathcal{V} \overset{\text{def}}{=} \{h_{\mathrm{w}} \in \mathcal{H} \mid h_{\mathrm{w}}(x_i) = y_i \ \text{ for } \ i = 1, \ldots, m\}, \tag{2.39}$$

which consists of all linear classifiers in feature space which separate the training set without errors, is referred to as the version space. While the general notion of version space was introduced by Mitchell (1982), Tong and Koller (2001c) employed this model to study linear classifiers in the context of pool-based active learning. Using the canonical isomorphism between linear classifiers and normalized weight vectors, we redefine the notion of version space as follows:

Definition 2.7 Version Space (Mitchell, 1982)
Suppose we are given a set $\{(x_1, y_1), \ldots, (x_m, y_m)\} \subset \mathcal{X} \times \mathcal{Y}$ of linearly separable binary training examples. Then, we denote the set of normalized weight vectors corresponding to consistent linear classifiers by

$$\mathcal{V} \overset{\text{def}}{=} \{\mathbf{w} \in \mathcal{W} \mid h_{\mathrm{w}}(x_i) = \mathrm{sign}(\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}) = y_i \ \text{ for } \ i = 1, \ldots, m\} \tag{2.40}$$

$$= \{\mathbf{w} \in \mathcal{F} \mid h_{\mathrm{w}}(x_i) = \mathrm{sign}(\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}) = y_i \ \text{ for } \ i = 1, \ldots, m, \ \|\mathbf{w}\|_{\mathcal{F}} = 1\}. \tag{2.41}$$

$\mathcal{V}$ is referred to as the version space with respect to the set of training examples $\{(x_1, y_1), \ldots, (x_m, y_m)\}$.

Let us take a closer look at the geometric shape of the version space in the remainder of this section. Learning can be considered as a search problem within the weight space: Each training example (x_i, y_i) limits the volume of the version space because to correspond to a consistent classifier a weight vector $\mathbf{w} \in \mathcal{W}$ has to satisfy the following condition:

$$h_{\mathrm{w}}(x_i) = \mathrm{sign}(\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}) = y_i \quad \Leftrightarrow \quad y_i \langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}} > 0. \tag{2.42}$$

In other words, consistent solutions are restricted to a halfspace whose boundary is the hyperplane with the normal vector $y_i\phi(x_i)$. For a fixed embedded input pattern $\phi(x_i)$, the class label y_i determines the orientation of the halfspace. Moreover, $\mathcal{V}$ is the intersection of m halfspaces (a convex polyhedral cone) with the unit hypersphere in feature space $\mathcal{F}$ (Rujan and Marchand, 2000).

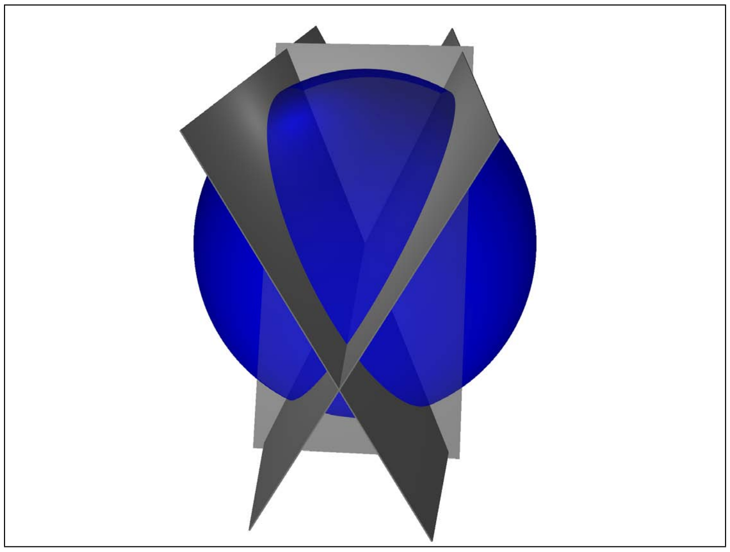


Figure 2.4 *The depicted hyperplanes correspond to the boundaries of the constraints that are imposed on the weight space by three labeled training examples. Thus, the version space is given by the intersection of three halfspaces with the unit hypersphere.*

2.4 Support Vector Machines

Having established appropriate notions and models to the foundation of linear classifiers in general, we turn our attention to learning methods which actually train linear classifiers for the remainder of this chapter.

Research in the field of kernel machines has evolved a comprehensive collection of learning algorithms which both have a solid foundation in statistical learning theory and achieve state-of-the-art accuracy in many significant areas of application. Since learning algorithms and kernels can be considered as separate components, kernel methods support a modular architecture of learning systems. In the following, we will embark on a more detailed discussion of one of the most popular and well-established methods, so-called support vector machines (Boser et al., 1992).

Let us formally define important notions underlying the theory of support vector machines:

Definition 2.8 Functional and Geometric Margin
Denote by $h_{\mathrm{w}} : \mathcal{X} \to \{-1, +1\}$ a linear classifier[4] *in a kernel-induced feature space $\mathcal{F}$ and denote the corresponding linear function by f_{w}. Then, the functional margin of*

4. We take the liberty to refer to h_{w} as a linear classifier in the feature space $\mathcal{F}$, though $\mathcal{F}$ is not considered as being the input space but $\mathcal{X}$.

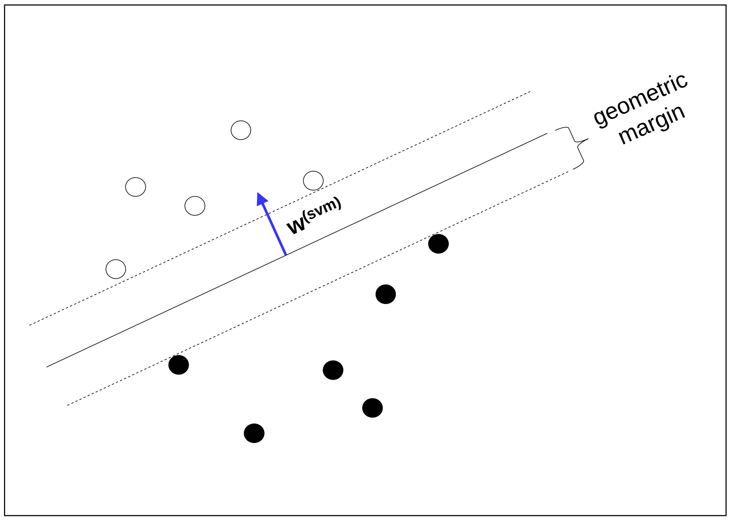


Figure 2.5 *Assume, we are given a linearly separable problem and a classifier corresponding to a separating hyperplane. Then, the geometric margin equals the minimum Euclidean distance within the set of examples to the classification boundary.*

an example (x, y) with respect to h_{w} is defined as

$$\rho_{h_{\mathrm{w}}}(x, y) \overset{def}{=} y f_{\mathrm{w}}(x) = y\langle \mathbf{w}, \phi(x)\rangle_{\mathcal{F}}. \tag{2.43}$$

The functional margin $\rho_{h_{\mathrm{w}}}$ of a set of examples $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ with respect to h_{w} is defined as

$$\rho_{h_{\mathrm{w}}} \overset{def}{=} \min_{i=1,\ldots,m} \rho_{h_{\mathrm{w}}}(x_i, y_i) = \min_{i=1,\ldots,m} y_i\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}. \tag{2.44}$$

Moreover, we refer to normalized functional margins $\frac{\rho_{h_{\mathrm{w}}}(x,y)}{\|\mathrm{w}\|_{\mathcal{F}}}$ and $\frac{\rho_{h_{\mathrm{w}}}}{\|\mathrm{w}\|_{\mathcal{F}}}$, respectively, as geometric margins.

Remark 2.9
The notion of the geometric margin of a set of examples has an intuitive interpretation as the oriented minimum Euclidean distance to the classification boundary (see Figure 2.5).

Given a training set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ which is linearly separable in the kernel-induced feature space, there exists a unique linear classifier $h_{\mathrm{w}^{(\mathrm{svm})}}$ maximizing the functional margin of the set of examples (Boser et al., 1992):

$$h_{\mathrm{w}^{(\mathrm{svm})}} \overset{\mathrm{def}}{=} \operatorname*{argmax}_{h_{\mathrm{w}} \in \mathcal{H}} \rho_{h_{\mathrm{w}}} \tag{2.45}$$

$$= \operatorname*{argmax}_{h_{\mathrm{w}} \in \mathcal{H}} \min_{i=1,\ldots,m} \rho_{h_{\mathrm{w}}}(x_i, y_i). \tag{2.46}$$

Support Vector Machine

The unique maximum margin classifier $h_{\mathrm{w}^{(\mathrm{svm})}}$ is also referred to as (hard margin) support vector machine.

As defined in Section 2.1, the hypothesis space of linear classifiers $\mathcal{H}$ is normalized by restricting weight vectors to unit length. However, this optimization problem becomes more amenable to standard solving techniques if we impose a data-dependent normalization constraint:

Definition 2.10 Canonical Normalization
A linear classifier h_{w} with $\mathbf{w} \in \mathcal{F}$ is in canonical form with respect to a set of examples $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ if the following equation holds:

$$\min_{i=1,\ldots,m} |\rho_{h_{\mathrm{w}}}(x_i, y_i)| = \min_{i=1,\ldots,m} |\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}| = 1. \tag{2.47}$$

If a linear classifier h_{w} is given in canonical form, the geometric margin evaluates to $\frac{1}{\|\mathrm{w}\|_{\mathcal{F}}}$. The optimization problem of calculating the maximum margin classifier can be reformulated as (see (Vapnik, 1998))

$$\underset{\mathrm{w}\in\mathcal{F}}{\text{minimize}} \quad \|\mathbf{w}\|^2_{\mathcal{F}} \tag{2.48}$$

$$\text{subject to} \quad y_i\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}} \geq 1 \quad \text{for } i = 1, \ldots, m. \tag{2.49}$$

Note that the squared norm is considered for mathematical convenience. Furthermore, it can be shown that equality holds in (2.49) for some $i \in \{1, \ldots, m\}$ for the maximum margin classifier, which is therefore given in canonical form. The linearly constraint convex quadratic programming problems associated with support vector machines are amenable to general-purpose solving techniques from optimization theory and can be solved in polynomial time in terms of the number of training examples. More precisely, the theoretical order of complexity was shown to be in $\mathcal{O}(m^3)$ where m denotes the number of examples (Burges, 1998). Moreover, a lot of effort has been spent on the development of more efficient, specialized numerical quadratic programming solvers over the last years, which typically realize an (empirically estimated) quadratic time complexity (Platt, 1999a; Joachims, 1999a).

Time Complexity

Our reasoning so far was restricted to the linearly separable case although nonseparable problems are of particular importance in many areas of application. Therefore, we will now discuss the quadratic soft margin modification of the above-defined hard margin support vector machine formulation which extends the applicability of support vector machines to linearly nonseparable problems. To address this category of problems, we consider the following optimization problem (Cortes and Vapnik, 1995):

Soft Margin Modification

$$\underset{\mathrm{w}\in\mathcal{F},\, \xi_1,\ldots,\xi_m\in\mathbb{R}}{\text{minimize}} \quad \|\mathbf{w}\|^2_{\mathcal{F}} + C\sum_{i=1}^{m} \xi^2 \tag{2.50}$$

$$\text{subject to} \quad y_i\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}} \geq 1 - \xi_i \quad \text{for } i = 1, \ldots, m,$$

$$\xi_i \geq 0 \quad \text{for } i = 1, \ldots, m$$

with $C > 0$ being fixed. This modification allows for examples yielding a margin less than one (including misclassified training examples) and penalizes them by the squared functional difference in the objective function, which is also referred to as quadratic margin loss or L2-loss.[5]

The penalty parameter C controls the weight of two particular objectives, namely achieving a large margin and minimizing the training error. To point up the difference to *hard margin* support vector machines for linearly separable classification problems, the linear classifier corresponding to the weight vector of the solution of the optimization problem (2.50) is referred to as *soft margin* support vector machine. For $C \to \infty$ the soft margin formulation converges to the hard margin formulation.

While there are alternative approaches to generalize the basic support vector machine formulation, such as using different penalty terms, the quadratic soft margin modification has an appealing connection to the hard margin formulation: Modifying the optimization problem as stated in (2.50) is equivalent to modifying the kernel in the original hard margin optimization problem in the following manner (Shawe-Taylor and Cristianini, 1999): The positive constant $\frac{1}{C}$ is added to the diagonal elements of the kernel matrix, $K_{ij} \overset{\text{def}}{=} k(x_i, x_j) + \frac{1}{C}\delta_{ij}$ where δ_{ij} denotes the Kronecker delta function, which equals 1 if $i = j$ and 0 otherwise. Furthermore, we recall that for any given training set there exists some $C > 0$ such that the positive and negative class are linearly separable in the kernel-induced feature space. The connection between quadratic soft margin and hard margin support vector machines and the existence of a kernel of suitable structure that renders a linearly separable learning problem allows us to cast a unified view on learning problems, independently of linear separability in input space.

Now, denote by $\mathbf{w}^{(\text{svm})}$ the weight vector of a quadratic soft margin support vector machine classifier which has been trained on a given binary classification problem. Then, the (potentially modified) kernel yields an embedding of the training set into a feature space where positive and negative examples are linearly separable. Let us recall the definition of the version space introduced in Section 2.3:

$$\mathcal{V} \overset{\text{def}}{=} \{\mathbf{w} \in \mathcal{F} \mid h_{\mathbf{w}}(x_i) = \operatorname{sign}(\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}) = y_i \text{ for } i = 1, \ldots, m, \ \|\mathbf{w}\|_{\mathcal{F}} = 1\}. \tag{2.51}$$

Viewing $\mathbf{w}^{(\text{svm})}$ as the solution of the hard margin support vector machine formulation, we conclude from (2.49) that $\frac{\mathbf{w}^{(\text{svm})}}{\|\mathbf{w}^{(\text{svm})}\|_{\mathcal{F}}} \in \mathcal{V}$.

Duality

Let us assume that all embedded input patterns are normalized to unit length in feature space, i.e., $\|\phi(x_i)\|_{\mathcal{F}} = 1$ for $i = 1, \ldots, m$. We adopt a dual perspective and interpret weight vectors $\mathbf{w} \in \mathcal{W}$ as data points and $\phi(x_i)$ as normal vectors of hyperplanes in feature space. Then, the margin $\rho_{h_{\mathbf{w}}}(x_i, y_i) = y_i\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}$ evaluates to the directed Euclidean distance of $\mathbf{w}$ from the hyperplane corresponding to the normal vector $y_i\phi(x_i)$. Thus, calculating the maximum margin classifier is equivalent to finding the center of the largest ball that is inscribable in version space (see (Tong and Koller, 2001c; Ruján and Marchand, 2000; Herbrich et al., 2001)). Therefore, $\frac{\mathbf{w}^{(\text{svm})}}{\|\mathbf{w}^{(\text{svm})}\|_{\mathcal{F}}}$ can

Geometric Interpretation

5. Note that the arguably more straightforward approach of minimizing the number of misclassified training examples yields an NP-hard optimization problem (Cortes and Vapnik, 1995; Ben-David and Simon, 2001), while this modification does not increase the computational time complexity.

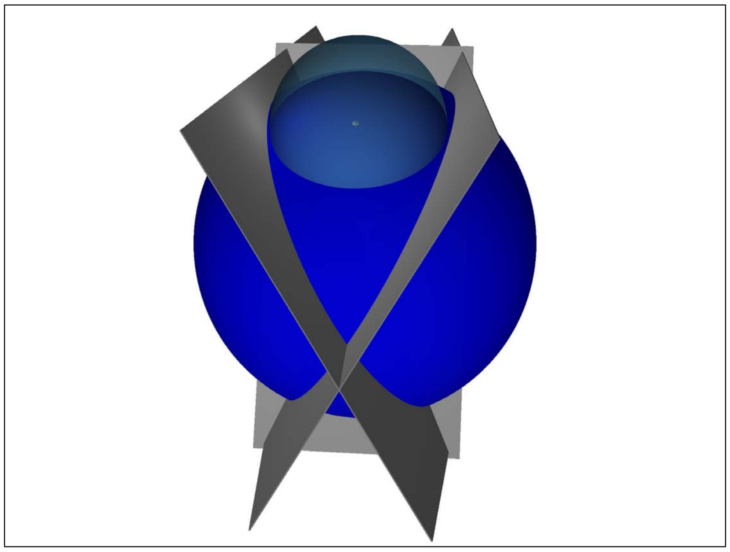


Figure 2.6 *Assume that embedded input patterns are normalized to unit length in feature space. Then, calculating the maximum margin classifier is equivalent to finding the center of the largest ball that is inscribable in version space.*

be viewed as a reasonable approximation of the center of mass of the version space under the assumption of a regularly shaped version space.[6] More generally, the set of all centers of maximum radius balls inside the convex polyhedral cone induced by the constraints corresponding to training examples is given by $\{\lambda \mathbf{w}^{(\mathrm{svm})} \,|\, \lambda > 0\}$ (Ruján and Marchand, 2000). In other words, this set corresponds to the ray in the direction of $\mathbf{w}^{(\mathrm{svm})}$. Figure 2.7 depicts several maximum radius balls which correspond to particular normalizations of the weight vector $\mathbf{w}^{(\mathrm{svm})}$.

Viewing margin maximization as approximating the center of mass depends on the assumption of a normalized set of examples, but can also serve as a justification of the benefit of normalization (Herbrich and Graepel, 2002). In order to normalize a set of example, it is not necessary to directly normalize embedded input patterns. It is more convenient to simply consider the modified kernel $k^*(x, x') \overset{\mathrm{def}}{=} \frac{k(x,x')}{k(x,x)\,k(x',x')}$ which yields unit modulus, $\|\phi(x)\|_{\mathcal{F}} = \sqrt{k^*(x,x)} = 1$, for all patterns. Moreover, the commonly used RBF kernel, $k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$, does not require any modification since $k(\mathbf{x}, \mathbf{x}) = 1$ holds for all $\mathbf{x} \in \mathbb{R}^N$. In other words, all input patterns are mapped onto the unit hypersphere in feature space.

Dual Representation

The presentation of support vector machines has been based on a primal representation in feature space so far. However, a major benefit of incorporating the concept of kernels is a dual representation as finite kernel expansions. The representer theorem (Schölkopf et al., 2001) identifies a broad class of optimization problems which admit

6. See Chapter 8 for a more formal analysis of the accuracy of this approximation.

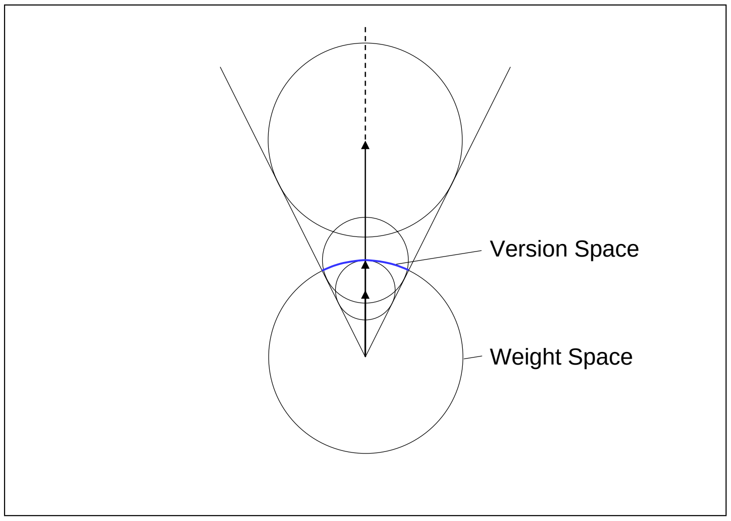


Figure 2.7 *Assume that the embedded input patterns are normalized to unit length in feature space. Then, the set of all centers of maximum radius balls inside the convex polyhedral cone induced by the constraints corresponding to training examples is given by the ray in the direction of* $\mathbf{w}^{(svm)}$. *This figure depicts three different scalings of* $\mathbf{w}^{(svm)}$ *for a two-dimensional feature space: The top circle corresponds to the canonical normalization where the minimum distance of* $\mathbf{w}^{(svm)}$ *to the restricting hyperplanes evaluates to one, the circle in the center corresponds to a unit normalization where* $\mathbf{w}^{(svm)} \in \mathcal{V}$ *and the bottom circle corresponds to a normalization where the maximum radius ball is restricted to the unit ball.*

a dual representation as expansions in terms of the embedded training patterns. In particular, the optimization problem associated with support vector machines satisfies the requirements of the representer theorem and therefore any weight vector $\mathbf{w}^{(\text{svm})}$ corresponding to a support vector machine has a finite expansion in terms of the embedded training patterns $\phi(x_1), \ldots, \phi(x_m)$:

$$\mathbf{w}^{(\text{svm})} = \sum_{i=1}^{m} \alpha_i \phi(x_i) \tag{2.52}$$

where $\alpha_i \in \mathbb{R}$ for $i = 1, \ldots, m$. Hence, the corresponding classifier $h_{\mathbf{w}^{(\text{svm})}} : \mathcal{X} \to \mathcal{Y}$ can be represented as

$$h_{\mathbf{w}^{(\text{svm})}}(\cdot) = \text{sign}(\langle \mathbf{w}^{(\text{svm})}, \phi(\cdot) \rangle_{\mathcal{F}}) \tag{2.53}$$

$$= \text{sign}\left(\left\langle \sum_{i=1}^{m} \alpha_i \phi(x_i), \phi(\cdot) \right\rangle_{\mathcal{F}}\right) \tag{2.54}$$

$$= \text{sign}\left(\sum_{i=1}^{m} \alpha_i k(x_i, \cdot)\right) \tag{2.55}$$

Thus, independently of the possibly high or even infinite dimension of the induced feature space, support vector machines admit a finite dual representation. Furthermore, the embedded input patterns $\phi(x_i)$ which correspond to nonzero coefficients α_i are referred to as support vectors. Moreover, as a consequence of the representation (2.55), the particular choice of feature space and feature map corresponding to a given kernel is not relevant since the solution depends on the training patterns only in terms of the kernel function (see (Schölkopf and Smola, 2002, p. 39) for details).

Support Vector

2.5 Bayes Point Machines

The previous section showed that support vector machines can be viewed as a reasonable approximation of the center of mass of the version space under the assumption of a regularly shaped version space. This section reviews a more sophisticated method based on random walk sampling for approximating the center of mass. Moreover, algorithms approximating the center of mass of the version space are theoretically well-founded because they approximate the optimal projection, the so-called Bayes point, of the Bayes classification strategy to the weight space (Herbrich et al., 2001). Therefore, these algorithms are referred to as Bayes point machines.

Bayes Point

In the following, we will focus on the theoretical reasoning underlying Bayes point machines and review the kernel billiard approach for approximating the Bayes point by the center of mass of the version space. We will point out important modifications of the kernel billiard algorithm presented in (Minka, 2001) to improve numerical stability and accuracy. These improvements had considerable influence on the outcome of the experiments conducted on a novel active learning strategy to be presented in Section 4.2.

In compliance with the standard model in classification learning (see Section 1.1), the Bayesian approach to learning (MacKay, 1991; Herbrich et al., 2001; Minka, 2001) assumes that we are given a training set of examples $(z_1, \ldots, z_m) = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$ drawn independently and identically distributed according to some probability distribution $\mathrm{P}^{\mathrm{Z}} = \mathrm{P}^{(\mathrm{X},\mathrm{Y})}$. As a consequence of the representer theorem, Herbrich et al. (2001) conclude that the center of mass of the version space admits a finite kernel representation. Thus, it is sufficient to consider the linear span of embedded input patterns to express the kernel billiard process to be described subsequently. In the following, we will assume that the given kernel renders a linearly separable learning problem (see Section 2.4 for details) and consider the linear span of the embedded input patterns as the effective feature space $\mathcal{F} = \{\sum_{i=1}^{m} \alpha_i \phi(x_i) \,|\, \alpha_i \in \mathbb{R}\}$. Furthermore, the Bayesian approach assumes that we are given a prior probability distribution (prior belief) P^{H} where H denotes a random variable on the hypothesis space $\mathcal{H}$. Moreover, we consider the 0-1-loss function l_{0-1} as stated in Section 1.1, thus, the expected risk of a classifier $h \in \mathcal{H}$ evaluates to the misclassification probability on examples (x, y) drawn according to $\mathrm{P}^{(\mathrm{X},\mathrm{Y})}$:

$$\mathrm{R}(h) = \int_{\mathcal{X}\times\mathcal{Y}} l_{0-1}(y, h(x))\, d\mathrm{P}^{(\mathrm{X},\mathrm{Y})} = \mathrm{P}^{(\mathrm{X},\mathrm{Y})}(h(x) \neq y). \tag{2.56}$$

Bayes Classification Strategy

An appealing property of the Bayesian model of learning is the existence of a classification strategy h_{Bayes} which is optimal in the sense that it has minimum expected misclassification probability with respect to the random choice of h (see (Herbrich et al., 2001) for a more detailed presentation). This classification strategy is referred to as Bayes classification strategy. The predicted class label of a pattern x is given by

$$h_{\text{Bayes}}(x) \overset{\text{def}}{=} \operatorname*{argmin}_{y \in \mathcal{Y}} \mathsf{E}(l_{0-1}(y, \mathsf{H}(\mathsf{X})) \mid \mathsf{Z}^m = (z_1, \ldots, z_m)). \tag{2.57}$$

While exhibiting this appealing optimality property, the Bayes classification strategy suffers from two major deficiencies (Gilad-Bachrach et al., 2004): In general, the Bayes classification strategy is not an element of the hypothesis space and therefore faces the problem of possibly inconsistent classifications with respect to the considered hypothesis space. Furthermore, it is computationally demanding in the case of linear classifiers to calculate the predicted class label y for a given pattern x. To circumvent these drawbacks, the Bayes point classifier approximates the Bayes classification strategy by a single classifier $h_{\text{bp}} \in \mathcal{H}$ from the hypothesis space which is chosen according to minimum expected difference in misclassification probability, i.e.,

Bayes Point Classifier

$$h_{\text{bp}} \overset{\text{def}}{=} \operatorname*{argmin}_{h \in \mathcal{H}} \mathsf{E}(l_{0-1}(h(\mathsf{X}), \mathsf{H}(\mathsf{X})) \mid \mathsf{Z}^m = (z_1, \ldots, z_m)). \tag{2.58}$$

Herbrich et al. (2001) showed that under the assumption of a uniform prior distribution over the weight space $\mathcal{W}$ and a spherical Gaussian distribution of embedded input patterns[7], the weight vector $\mathbf{w}^{(\text{bp})}$ corresponding to the Bayes point classifier h_{bp} can be approximated with high accuracy by the center of mass of the version space $\mathbf{w}^{(\text{center})}$, which is formally defined as

$$\mathbf{w}^{(\text{center})} \overset{\text{def}}{=} \frac{\mathsf{E}(\mathsf{W} \mid \mathsf{Z}^m = (z_1, \ldots, z_m))}{\|\mathsf{E}(\mathsf{W} \mid \mathsf{Z}^m = (z_1, \ldots, z_m))\|_{\mathcal{F}}} \tag{2.59}$$

where W is a random variable having a uniform distribution over the hypothesis space of unit weight vectors $\mathcal{W}$. Note that the posterior distribution of weight vectors $\mathsf{P}^{\mathsf{W}|\mathsf{Z}^m=(z_1,\ldots,z_m)}$ is the restriction of the prior distribution P^{W} to the version space. Thus, giving rise to the following definition:

$$h_{\text{bpm}} : \mathcal{X} \to \{-1, +1\} \tag{2.60}$$
$$x \mapsto \operatorname{sign}(\langle \mathbf{w}^{(\text{center})}, \phi(x) \rangle_{\mathcal{F}}). \tag{2.61}$$

Bayes Point Machine

The kernel classifier h_{bpm} as well as classifiers which are based on some approximation $\mathbf{w}^{(\text{bpm})}$ of the exact center of mass of the version space $\mathbf{w}^{(\text{center})}$ are referred to as Bayes point machines.

Gilad-Bachrach et al. (2004) proved that the misclassification probability of a Bayes point machine is greater than the misclassification probability of the corresponding

7. The probability distribution of $\phi(\mathsf{X})$ in feature space is governed by the probability density function $f^{\phi(\mathsf{X})} : \mathcal{F} \to \mathbb{R}$, $\bar{x} \mapsto \frac{2}{\sqrt{\pi}} e^{-\|\bar{x}\|^2}$.

Bayes classification strategy by a constant factor of $(e-1) \approx 1.71$ at most, i.e.,

$$\mathrm{R}(h_{\mathrm{bpm}}) \leq (e-1)\,\mathrm{R}(h_{\mathrm{Bayes}}). \tag{2.62}$$

A straightforward Monte Carlo approach to estimate the center of mass based on the average of a finite number of samples drawn independently and identically distributed from a uniform distribution over the version space is computationally very demanding (Fine et al., 2002). Moreover, the currently best known algorithm (Kannan et al., 1997) for sampling from convex bodies has a complexity of $\mathcal{O}^*(N^3)$ for each sample call and a preprocessing complexity of $\mathcal{O}^*(N^5)$ where N denotes the dimension of the convex body, which is impractical for our area of application.[8]

Kernel Billiard

In order to reduce the computational complexity, kernel billiard algorithms (Ruján and Marchand, 2000; Herbrich et al., 2001; Minka, 2001) approximate the center of mass by averaging over the trajectory of a billiard that is bounced within the version space. The version space $\mathcal{V}$ is the intersection of m halfspaces (a convex polyhedral cone) with the unit hypersphere in the feature space $\mathcal{F}$. The polyhedral cone is bounded by hyperplanes having normal vectors $y_1\phi(x_1), \ldots, y_m\phi(x_m)$ as can be seen from Definition 2.7. Both (Ruján and Marchand, 2000) and (Herbrich et al., 2001) face the problem that the billiard ball can escape from the version space to which these algorithms recover from by initiating a restart procedure which introduces a severe bias to the estimate (Minka, 2001). To reduce this bias, Minka (2001) proposed a modified billiard algorithm which operates within the entire convex polyhedral cone (rather than projecting the billiard onto the unit hypersphere) and considers the unit hypersphere as an additional boundary. In other words, the hypothesis space of weight vectors is redefined by imposing the normalization constraint $\|\mathbf{w}\|_{\mathcal{F}} \leq 1$ where the corresponding prior distribution P^{W} is the uniform distribution over the unit ball in $\mathcal{F}$. Moreover, this modification gives rise to a redefined convex version space, which is more amenable to theoretical analysis and will be referred to later on (see Definition 2.7):

Definition 2.11 Convex Version Space

Suppose we are given a set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ of linearly separable binary training examples. Then, we denote the bounded set of weight vectors corresponding to consistent linear classifiers by

$$\mathcal{V} \overset{def}{=} \{\mathbf{w} \in \mathcal{F} \mid h_{\mathrm{w}}(x_i) = \mathrm{sign}(\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}) = y_i \ \ \textit{for} \ \ i = 1, \ldots, m, \ \|\mathbf{w}\|_{\mathcal{F}} \leq 1\}. \tag{2.63}$$

$\mathcal{V}$ is referred to as the convex version space with respect to the set of training examples $\{(x_1, y_1), \ldots, (x_m, y_m)\}$.

Though this approach prevents the billiard ball from escaping the version space, Minka (2001) suggests to employ the restart procedure every 100 bounces since the sampling process may not be perfectly ergodic such that a typical trajectory does not homogenously cover the version space (Ruján and Marchand, 2000). In our

8. The $\mathcal{O}^*$-notation neglects logarithmic factors as well as other factors depending on parameters which control the accuracy.

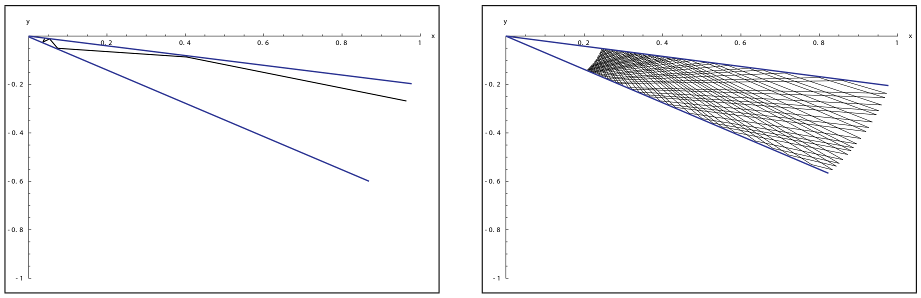

(a) (b)

Figure 2.8 *Trajectories close to the origin, such as in (a), can lead to severe numerical instabilities. By adding the surface of a ball with radius* $r < 1$ *as an additional boundary, these problems can be avoided. Figure (b) depicts a trajectory which was generated using the modified billiard algorithm with a ball of radius* $r = 0.25$.

experiments, we achieved slightly better results by decreasing the number of bounces to 10. As noted by Herbrich et al. (2001), current research in the field of Markov Chain Monte Carlo methods includes the question of ergodicity of methods such as billiard sampling.

The active learning strategy that will be proposed in Section 4.2 is based on kernel billiard sampling. Since the problem of an escaping billiard frequently occurred during an initial run of low-dimensional experiments, we decided to base our implementation on the algorithm presented in (Minka, 2001). However, we faced the problem of severe numerical instabilities due to trajectories close to the origin, i.e., the norm $\|\cdot\|_{\mathcal{F}}$ of the position of the billiard is close to zero (see Figure 2.8). As noted in (Minka, 2001), we can sample from any prior distribution that assigns equal probability to all $\mathbf{w} \in \mathcal{V}$ of a given length. Therefore, we propose the following modification: We consider the surface of a ball of radius $0 < r < 1$ as an additional boundary and redefine the sampling pool as the intersection of the polyhedral cone with the unit ball and the complement of the additional ball. In an analogous manner to the aforementioned redefinition of the prior distribution, we have to adapt the notions of the hypothesis space of weight vectors and of the version space by imposing the constraint $r \leq \|\mathbf{w}\|_{\mathcal{F}} \leq 1$. A choice of $r = 0.1$ reliably solved the above-mentioned numerical problems in our experiments.

Numerical Instabilities

Restart Procedure

The restart procedure essentially calculates a new random velocity $\mathbf{v} = \sum_{i=1}^{m} \alpha_i \phi(x_i)$ of the billiard. While available implementations generate a random velocity by uniformly sampling weights α_i from $[-1, +1]$, followed by a normalization to unit length, this method does not generate a new velocity uniformly drawn from the unit hypersphere since $\phi(x_1), \ldots, \phi(x_m)$ are not guarantied to form an orthonormal basis. We incorporated a more sophisticated method from (Gilad-Bachrach et al., 2003): Using the standard Gram-Schmidt-algorithm (Lorenz, 1996), we calculate an orthonormal basis

$$b_1 = \sum_{i=1}^{m} \beta_{1,i}\phi(x_i), \ldots, b_r = \sum_{i=1}^{m} \beta_{r,i}\phi(x_i) \qquad (2.64)$$

of span $\{\phi(x_1), \ldots, \phi(x_m)\}$ where $\beta_{r,i} = 0$ for $i > r$. A more detailed description of the kernelized Gram-Schmidt-algorithm is given in Algorithm 1. Then, we independently sample γ_i (for $i = 1, \ldots, r$) from a standard normal distribution and calculate the new velocity by

$$\mathbf{v} = \sum\nolimits_{j=1}^{r} \gamma_j b_j = \sum\nolimits_{i=1}^{m} \left(\sum\nolimits_{j=1}^{r} \beta_{j,i} \gamma_j \right) \phi(x_i). \tag{2.65}$$

and normalize $\mathbf{v}$ to unit length. Besides the theoretical justification, experimental results will demonstrate the benefits of this modification.

Computational Complexity

The computational complexity of the presented kernel billiard algorithm is of order $\mathcal{O}(Mm^2)$ where M denotes the number of billiard bounces (typically $M = 1000$ in our experiments) and m denotes the number of training examples. In practice, this algorithm scales to moderately sized learning problems. While theoretical bounds on the accuracy of the approximation have not been established so far, empirical results indicate that kernel billiard algorithms are able to approximate the center of mass of the version space with an accuracy of about 10^{-2} to 10^{-3} in terms of Euclidean distance (Minka, 2001).

Algorithm 1 *Gram-Schmidt-Algorithm (in primal representation)*

```
input:
$x_1, \ldots, x_m$                                   (input patterns)

output:
$b_1, \ldots, b_r$        (orthonormal basis of span $\{\phi(x_1), \ldots, \phi(x_m)\}$)

for $i = 1, \ldots, m$ do
    $b_i \leftarrow \phi(x_i)$
    for $j = 1, \ldots, i-1$ do
        $b_i \leftarrow b_i - \frac{\langle \phi(x_i), b_j \rangle_{\mathcal{F}}}{\|b_j\|_{\mathcal{F}}} b_j$
    end for
    if $\|b_i\|_{\mathcal{F}} = 0$ then
        return $b_1, \ldots, b_{i-1}$
    end if
    $b_i \leftarrow b_i / \|b_i\|_{\mathcal{F}}$
end for

return $b_1, \ldots, b_m$
```

II ACTIVE LEARNING

3 Active Learning Models

Overview

The introductory chapter provided a broad overview of the general concept and terminology of active learning. In this chapter, we will embark on a more detailed presentation of principal active learning models and categorize various approaches to active learning. The pool-based active learning model, which forms the theoretical basis for both binary classification learning and subsequent generalizations to multiclass and label ranking learning in this thesis, will be discussed in a more formal manner.

3.1 General Models in Active Learning

The superordinate concept of active learning refers to a collection of approaches which aim at reducing the labeling effort in supervised machine learning. We can distinguish between two principal categories of active learning algorithms, viz. *query learning* and *selective sampling*, which are based on different learning models.

Query Learning

Query learning was introduced by Valiant (1984) as a communication protocol between the learner and a so-called membership oracle in the theory of concept learning. It refers to a class of learning algorithms which are provided with the ability to sequentially select new patterns from the entire input space and to request the corresponding correct class labels (positive or negative membership) from a membership oracle. Alternatively, in compliance with standard terminology in related research, we refer to the process of selecting new patterns from the entire input space as constructing new examples. Query learning has been theoretically studied in detail using the probably approximately correct (PAC) learning model. Valiant (1984); Angluin (1988) showed that some target concepts which are not PAC-learnable using random samples, i.e., the number of training examples cannot be bounded in terms of a polynomial function which depends on certain parameters controlling the accuracy, are efficiently learnable in the PAC model by query learning. However, Eisenberg and Rivest (1990) analyzed certain classes of concepts which are known to be efficiently learnable in the PAC learning model already by random samples and proved that the capability to actively construct new examples does not substantially increase the efficiency of learning in terms of the necessary number of labeled examples in general.

Cohn et al. (1994) introduced an approach to concept learning via membership queries based on sampling from the region of uncertainty, which forms the set of all input patterns $x \in \mathcal{X}$ such that there are two concepts which are consistent with the set of labeled examples and yet disagree on the classification of x. As a consequence of the practical difficulty and computational complexity of representing the region of uncertainty even for simple concepts, Cohn et al. (1994) considered an approximate strategy which maintains both a superset and a subset of the region of uncertainty and described an implementation using feedforward artificial neural networks. Improvements over random sampling were demonstrated on a number of simple learning problems, while it was noted that this approach exhibits both theoretical and practical limitations on more complex problem domains.

Furthermore, Baum (1991) introduced a query learning algorithm based on feed-forward artificial neural networks with one hidden layer. Employing this query learning algorithm on an optical character recognition (OCR) task, Baum (1991) faced the problem of constructing patterns which were not classifiable by human beings, i.e., the constructed patterns did not correspond to valid characters. Thus, along with the ability to construct new examples from the entire input space in active learning, the existence of a powerful membership oracle is required. This assumption may not be reasonable for several practically relevant learning problems. Therefore, most applications of active learning are based on the alternative *selective sampling learning model*.

Selective Sampling

In contrast to query learning, in the *selective sampling model*, the learner is restricted to request target objects corresponding to input patterns which are contained in a given finite set (*pool-based model*) or the learning algorithm is provided with the capability to determine whether or not to request the associated target objects of sequentially presented patterns (*stream-based model*).

Stream-based Model

The seminal Bayesian *query-by-committee* (Seung et al., 1992) approach considers the stream-based model and determines whether to request particular class labels based on the disagreement of the predictions within a set of Gibbs classifiers[1]. The generality and simplicity of the query-by-committee approach initiated a series of follow-up publications which studied theoretical properties and considered its application to special categories of classifiers. Freund et al. (1997) proved that certain classes of concepts which had been identified by Eisenberg and Rivest (1990) not to be efficiently learnable in the PAC query learning model are efficiently learnable in the stream-based model using the query-by-committee approach. The essential difference between these models is that in the latter model the learning algorithm is granted access to unlabeled examples. Moreover, by accessing a stream of unlabeled examples, the learner is provided with information about the input distribution of patterns and may avoid the problem of unclassifiable patterns encountered by Baum (1991). Fine et al. (2002) gave a detailed analysis of the query-by-committee strategy in the case of linear classifiers and presented a feasible way to simulate the Gibbs-based criterion under the assumption of an almost uniform distribution on the hypothesis space. A generalization to linear classifiers in kernel feature spaces and its implementation were discussed in (Gilad-Bachrach et al., 2003). Variants of the query-by-committee approach have been applied to text classification learning (Liere and Tadepalli, 1997; McCallum and Nigam, 1998) and to part-of-speech tagging problems (Argamon-

1. The Gibbs algorithm randomly draws a classifier from the given hypothesis space according to the posterior distribution.

Engelson and Dagan, 1999), among others. Furthermore, Argamon-Engelson and Dagan (1999); McCallum and Nigam (1998) adapted the query-by-committee approach to the pool-based selective sampling model and proposed alternative measures for quantifying the disagreement within a committee of probabilistic classifiers.

Lewis and Gale (1994); Lewis and Catlett (1994) introduced the pool-based selective sampling model in the context of text classification learning and proposed selection strategies based on the uncertainty of the prediction of a single classifier. Roy and McCallum (2001) proposed an approach to active learning which aims at directly optimizing the expected risk in the pool-based model. To this means, the expected risks for all unlabeled examples and assignments of class labels are estimated on the pool of unlabeled examples by evaluating posterior class probabilities according to a probabilistic classifier trained on the given set of labeled examples. Moreover, Roy and McCallum (2001) suggested several approximations, such as subsampling, and algorithmic optimizations to reduce the computational complexity. Abe and Mamitsuka (1998) proposed an approach to binary active learning which is based on query-by-committee and employs a committee of classifiers generated by bagging or boosting.

Several approaches to binary active learning in the field of kernel machines depend on some approximation of the center of mass of the version space (Tong, 2001; Campbell et al., 2000; Schohn and Cohn, 2000; Warmuth et al., 2002). Tong (2001) focused on text classification problems while Warmuth et al. (2002) investigated the efficiency of active learning in the process of chemical drug discovery. Moreover, the domain of image retrieval was successfully demonstrated as an area of application for this category of active learning approaches (Tong and Chang, 2001). We will provide a more detailed presentation of this class of approaches in Section 4.1.

Semi-supervised Learning

Being situated between two extreme settings in the spectrum of machine learning, namely supervised and unsupervised learning, the field of semi-supervised learning has received increasing attention.[2] Semi-supervised learning assumes both a labeled and an unlabeled set of examples being available, where typically the set of labeled examples is relatively small while there is a large amount of unlabeled examples.[3] As in active learning, the underlying objective is to exploit the availability of unlabeled examples, however, semi-supervised learning does not consider a sequential labeling process, but aims at improving the prediction function in terms of expected risk based on the given sets of examples. As a special instance of semi-supervised learning, the transductive inference setting assumes that the prediction function will be evaluated on the given set of unlabeled examples and thus the expected risk should be minimized with respect to these examples.[4]

Moreover, both McCallum and Nigam (1998) and Muslea et al. (2002) inter-

2. See (Seeger, 2001) for an exhaustive review on learning with both labeled and unlabeled examples.
3. Bennett and Demiriz (1999); Fung and Mangasarian (2001) proposed generalizations of support vector learning to the semi-supervised setting.
4. See (Vapnik, 1998; Joachims, 1999b) for transductive inference approaches in the field of support vector learning.

leaved active learning and semi-supervised learning in order to further expedite the learning process. While McCallum and Nigam (1998) considered a combination of a committee-based approach with expectation maximization based on a naive Bayes model, Muslea et al. (2002) combined a multi-view active learning algorithm and a probabilistic version of co-training.

Beyond Binary Classification

Most of the aforementioned research in active learning has focused on learning a binary-valued prediction function. Beyond the binary classification setting, Cohn et al. (1996) examined active learning for statistical learning models in the context of regression learning. A related active learning approach to class probability estimation and regression, which was developed along a similar line of reasoning, was proposed by Saar-Tsechansky and Provost (2004). Both approaches focus on the underlying objective of reducing the expected variance of the prediction function. While (Cohn et al., 1996) is based on closed-form variance models over the entire input space, Saar-Tsechansky and Provost (2004) considered a more general bootstrap sampling technique for estimating the local variance which extends to arbitrary hypothesis spaces. Furthermore, active learning of both the structure of a causal Bayesian network (Tong and Koller, 2001b) and the parameters of a fixed Bayesian network (Tong and Koller, 2001a) were investigated.

We will discuss extensions to multiclass classification learning in more detail in Chapter 6 and propose a novel extension based in the context of kernel machines. Furthermore, using a similar line of reasoning, we will extend active learning to so-called label ranking problems, a category of learning problems which suffers tremendously from the necessary labeling effort, but has not been considered in active learning research so far.

3.2 Pool-based Active Learning

Fundamental notions of the pool-based active learning model have already been defined in an informal manner in the introductory chapter and in the preceding section of this chapter. In the remaining part, we will embark on a more formal presentation of the pool-based selective sampling model. We will consider this active learning model in the remainder of this thesis and, unless otherwise noted, refer to this model as (pool-based) active learning in the following to simplify our presentation.

Let us proceed to some basic definitions for the formalization of the pool-based active learning model:

Definition 3.1 Pool/Set of Unlabeled Examples
Suppose we are given a nonempty input space $\mathcal{X}$. Then, we refer to a finite set of input patterns $\mathcal{U} = \{x_1, \dots, x_n\} \subset \mathcal{X}$ as a pool or set of unlabeled examples.

Definition 3.2 Set of Labeled Examples
Denote by $\mathcal{X}$ and $\mathcal{Y}$ nonempty input and target spaces, respectively. A finite set $\mathcal{L} = \{(x_1, y_1), \dots, (x_m, y_m)\} \subset \mathcal{X} \times \mathcal{Y}$ consisting of pairs of input patterns and associated target objects is referred to as a set of labeled examples.

The essential idea behind active learning is to select promising patterns from a given pool of unlabeled examples $\mathcal{U}$ in the sense that the corresponding target objects contribute to a more accurate prediction function. To guide the selection process, the active learner is provided with the pool of unlabeled examples $\mathcal{U}$ and the set of already labeled examples $\mathcal{L}$. In a straightforward manner, we define the notion of a selection strategy as follows:

Definition 3.3 Selection Strategy
Denoting the power set of a given set Ω by $\mathcal{P}(\Omega)$, we refer to a probabilistic mapping

$$\sigma : \mathcal{P}(\mathcal{X}) \times \mathcal{P}(\mathcal{X} \times \mathcal{Y}) \to \mathcal{P}(\mathcal{X}) \tag{3.1}$$

as selection strategy if the following holds: For any nonempty pool of unlabeled examples $\mathcal{U} \subset \mathcal{X}$ and set of labeled examples $\mathcal{L} \subset \mathcal{X} \times \mathcal{Y}$,

$$\sigma(\mathcal{U}, \mathcal{L}) \neq \emptyset \quad \textit{and} \quad \sigma(\mathcal{U}, \mathcal{L}) \subsetneq \mathcal{U} \quad \textit{if } |\mathcal{U}| \geq 2, \tag{3.2}$$

$$\sigma(\mathcal{U}, \mathcal{L}) = \mathcal{U} \quad \textit{if } |\mathcal{U}| = 1. \tag{3.3}$$

Remark 3.4
We consider both deterministic and probabilistic selection strategies, which depend on some source of randomness, thus given the same input possibly selecting different patterns from the pool of unlabeled examples.

Endowed with these basic notions, we can proceed to a formal description of the process of active learning. Assume we are given a nonempty pool of unlabeled examples $\mathcal{U}_0 \subset \mathcal{X}$ of cardinality $|\mathcal{U}_0| = n_0$ and an initial set of labeled examples $\mathcal{L}_0 \subset \mathcal{X} \times \mathcal{Y}$ of cardinality $|\mathcal{L}_0| = m_0$. If not otherwise noted, we assume these sets to be sampled according to some underlying probability distributions P^{X} and $\mathrm{P}^{(\mathrm{X},\mathrm{Y})}$, respectively, as stated in the introductory chapter. Suppose a given selection strategy σ is evaluated on $(\mathcal{U}_0, \mathcal{L}_0)$, thus selecting a finite set of patterns $\sigma(\mathcal{U}_0, \mathcal{L}_0) = \{x_{i_1}, \dots, x_{i_j}\}$. The corresponding target objects $y_{i_1}, \dots, y_{i_j}$ are assumed to be drawn according to the conditional probability distribution $\mathrm{P}^{\mathrm{Y}|\mathrm{X}}$. The updated pool of unlabeled examples and the augmented set of labeled examples are given by

$$\mathcal{U}_1 \overset{\text{def}}{=} \mathcal{U}_0 \backslash \sigma(\mathcal{U}_0, \mathcal{L}_0) \quad \text{and} \tag{3.4}$$

$$\mathcal{L}_1 \overset{\text{def}}{=} \mathcal{L}_0 \cup \{(x_{i_1}, y_{i_1}), \dots, (x_{i_j}, y_{i_j})\}. \tag{3.5}$$

In an inductive manner, the active learning process generates two finite sequences of sets $(\mathcal{U}_i)_i$ and $(\mathcal{L}_i)_i$ of strictly monotonously decreasing and increasing cardinality, respectively. The maximum length of the sequences is bounded by the cardinality n_0 of the initial pool of unlabeled examples $\mathcal{U}_0$. Moreover, we have to establish a stopping criterion for the sequential active learning process which can either be of dynamic nature and depend on a measure of the learning progress or be of static nature such as a fixed number of requested target objects.

For the definition of active learning strategies, it suffices to specify the selection mapping given a nonempty pool $\mathcal{U}$ and a labeled set $\mathcal{L}$. Let us introduce the Random selection strategy, which will be considered as a baseline strategy in various places.

Random Selection Strategy

Given some pool of unlabeled examples $\mathcal{U} = \{x_1, \ldots, x_n\}$ and a set of labeled examples $\mathcal{L}$, the Random strategy draws patterns from $\mathcal{U}$ according to a uniform distribution, thus effectively disregarding all information contained in the labeled set $\mathcal{L}$. More formally,

$$\sigma(\mathcal{U}) = \sigma(\{x_1, \ldots, x_n\}) = \{x_{\mathsf{S}}\} \tag{3.6}$$

where the random variable S is uniformly distributed on $\{1, \ldots, n\}$. Moreover, we will consider a simple generalization of the basic Random selection strategy which selects a fixed number of patterns by iteratively running the one-step Random selection strategy on the resulting unlabeled and labeled sets and returning the union of the selected patterns.

Evaluation Measure

In our experiments, we will empirically compare selection strategies which are compatible in the sense that they select an a priori fixed and equal number of patterns in every step. In the case of classification learning, the efficiency of a strategy is measured by estimating the classification accuracy (see Section 1.1) of a classifier trained on the resulting labeled datasets. In general, two given strategies are compared based on the sequence of estimated risks (or corresponding measures of accuracy).

Typical Experimental Setup

Typically, the experimental setup for real-world data is as follows: The given dataset of patterns and associated target objects is randomly split into a training and a test set of fixed ratio. Then, an initial set $\mathcal{L}_0$ of fixed cardinality is randomly drawn from the training set while the remaining patterns are submitted to the pool of unlabeled patterns $\mathcal{U}_0$. The target objects associated with the patterns in the pool are masked out for the selection strategies. Then, the above-defined active learning process starts and the accuracy of the given selection strategy is estimated by the average accuracy on the test set after every selection step. The selection process stops after a predefined number of rounds. To acquire stable estimates and compensate for effects based on the random choice of the initially labeled set and the split into training and test set, the aforementioned procedure is repeated and the results are averaged.

We will employ a slightly modified experimental setup on datasets which are supplied with a predefined split into a training and a test set. On these datasets, the supplied test set is used to estimate the accuracy in all runs while we randomly generate initial sets of labeled examples from the training set for each run.

4 Binary Classification

Overview

The previous chapter presented and categorized various approaches to active learning and defined the pool-based active learning model from a formal point of view. In this chapter, we will embark on a more detailed presentation of active learning selection strategies which are based on the version space model. The subsequent section reviews the class of selection strategies which aim at reducing the volume of the version space by means of approximating the center of mass and discusses the underlying theoretical reasoning. In Section 4.2, we analyze a linear learning setting where these approaches exhibit some potential shortcomings and propose an improved binary selection strategy to address this problem. However, this selection strategy is computationally very demanding and therefore we introduce a sophisticated subsampling technique for the reduction of the computational complexity, which preselects a subset of unlabeled examples according to a less demanding volume-based strategy as candidates for our novel selection strategy. From a theoretical point of view, the minimization of the volume of the version space is a *necessary* precondition for the minimization of the improved selection criterion. As we anticipated on account of our theoretical analysis, experimental results demonstrate that the two-layered subsampling approach substantially decreases the computational complexity without a concomitant loss of classification accuracy.

While the following chapter is restricted to binary classification learning, it provides the basis for subsequent generalizations to both multiclass and label ranking learning.

4.1 Volume-based Selection Strategies

In this section, we will review the principal reasoning behind a class of selection strategies which is based on the version space model for binary classification problems.[1] We group together several approaches which consider different approximations to the same underlying principle into this category of selection strategies. We start with a presentation of the general model and then proceed to a discussion of different approximate strategies.

Let us consider a linearly separable, binary classification problem in a kernel-induced feature space, i.e., we assume that there exists a linear classifier $h_{\mathbf{w}^*} : \mathcal{X} \rightarrow \{-1, +1\}$

1. See Section 2.3 for an introduction to the version space model.

such that $h_{w^*}(x_i) = y_i$ for every training example (x_i, y_i) with $i = 1, \ldots, m$. Linearly nonseparable sets of examples can be incorporated into this setting in an elegant manner by a suitable modification of the kernel function as discussed in Chapter 2. We recall the definition of the version space in the case of linear classifiers in a kernel-induced feature space $\mathcal{F}$:

$$\mathcal{V} \overset{\text{def}}{=} \{\mathbf{w} \in \mathcal{W} \mid h_{\mathrm{w}}(x_i) = \operatorname{sign}(\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}) = y_i \text{ for } i = 1, \ldots, m\} \tag{4.1}$$

where the weight space $\mathcal{W}$ is defined as the unit hypersphere in $\mathcal{F}$. Thus, $\mathcal{V}$ forms the set of (normalized) weight vectors corresponding to linear classifiers in feature space which are consistent with the given training set.

Learning can be viewed as a search problem within the weight space: Each labeled training example (x_i, y_i) limits the volume of the version space because to correspond to a consistent classifier h_{w} a weight vector $\mathbf{w}$ has to satisfy

$$h_{\mathrm{w}}(x_i) = \operatorname{sign}(\langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}}) = y_i \quad \Leftrightarrow \quad y_i \langle \mathbf{w}, \phi(x_i)\rangle_{\mathcal{F}} > 0. \tag{4.2}$$

In other words, consistent classifiers are restricted to the intersection of the weight space $\mathcal{W}$ with the halfspace defined by the normal vector $y_i\phi(x_i)$. The hyperplane $\{\mathbf{w} \in \mathcal{F} \mid \langle \mathbf{w}, y_i\phi(x_i)\rangle_{\mathcal{F}} = 0\}$ which corresponds to the normal vector $y_i\phi(x_i)$ forms the boundary of this halfspace. Moreover, for a fixed embedded input pattern $\phi(x_i)$, the class label y_i determines the orientation of the halfspace. From a geometric perspective, we can view the version space $\mathcal{V}$ as the intersection of m halfspaces (a convex polyhedral cone) with the unit hypersphere in the feature space $\mathcal{F}$ (see Figure 2.4).

Since it is computationally very demanding to calculate high-dimensional volumes, selection strategies based on exact volume calculations are not feasible in most areas of application. More precisely, the problem of computing the volume of a convex body is NP-hard (Elekes, 1986). However, the volume can be approximated in polynomial time to any finite precision with arbitrary low probability of failure by randomized algorithms (Dyer et al., 1991). The currently best algorithm (Lovász and Vempala, 2003) for approximating the volume of a convex body in $\mathbb{R}^N$ has a computational complexity of $\mathcal{O}^*(N^4)$ calls to a separation oracle where the asterisk indicates that the dependence on logarithmic factors in N and on parameters controlling the accuracy of approximation is not shown. Gilad-Bachrach et al. (2003) proposed an efficient kernelization where for the computational complexity the input dimension N has to be substituted by the number of labeled examples.

Computing Volumes of Convex Bodies

In the following, we will review selection strategies which focus on the objective of reducing the volume of the version space by means of approximate reduction measures. These approximate measures can be evaluated efficiently and have been successfully tested on many real-world problems.

A classical result from the theory of convex sets states that any halfspace containing the center of mass of a convex set also contains at least $1/e$ of the overall volume (Grünbaum, 1960; Bertsimas and Vempala, 2002). Denoting the version space corresponding to the set of labeled examples $\mathcal{L}_i$ by $\mathcal{V}_i$ and the associated center of mass by $\mathbf{w}_i^{(\text{center})}$, we assume that in step i the labeled set $\mathcal{L}_i$ is augmented by an example (x, y) which corresponds to a restricting hyperplane passing exactly through the current center of mass $\mathbf{w}_i^{(\text{center})}$ of the version space $\mathcal{V}_i$, i.e.,

$$0 = y\langle \mathbf{w}_i^{(\text{center})}, \phi(x)\rangle_{\mathcal{F}} = \langle \mathbf{w}_i^{(\text{center})}, \phi(x)\rangle_{\mathcal{F}}. \tag{4.3}$$

Then, independently of the actual class label y, the volume of the version space is reduced exponentially in terms of the number of labeled examples.[2] More formally, the following bounds hold for the sequence of volumes:

$$\left(\frac{1}{e}\right)^i \operatorname{vol}(\mathcal{V}_0) \leq \operatorname{vol}(\mathcal{V}_i) \leq \left(1 - \frac{1}{e}\right)^i \operatorname{vol}(\mathcal{V}_0) \quad \text{for } i \geq 1. \tag{4.4}$$

In order to derive practical selection strategies, we have to make a number of approximations: It is computationally expensive to calculate the center of mass in high-dimensional spaces to some specified level of accuracy. For this reason, we consider two approaches for the approximation of the center of mass of the version space.

In Section 2.5, we have already discussed a kernel billiard algorithm which calculates an approximation $\mathbf{w}^{(\text{bpm})}$ of the center of mass by means of a random walk in the version space. The computational complexity of this kernel billiard algorithm is of order $\mathcal{O}(Mm^2)$ where M denotes the number of billiard bounces (typically $M = 1000$ in our experiments) and m denotes the number of training examples. In practice, this algorithm scales to moderately sized learning problems. As mentioned in Section 2.5, empirical results indicate that kernel billiard algorithms are able to approximate the center of mass of the version space with an accuracy of about 10^{-2} to 10^{-3} in terms of Euclidean distance (Minka, 2001). However, theoretical bounds on the accuracy of the approximation have not been established so far.

The center $\mathbf{w}^{(\text{ball})}$ of the maximum radius ball inscribable in version space is a reasonable approximation of the center of mass under the assumption of a regular shape of the version space in the sense that it is roughly symmetric and not elongated into some direction (see (Herbrich et al., 2001) and Section 2.4 for more details). We will analyze convergence properties of this approximation in Chapter 8. Under the assumption of a normalized set of patterns, $\mathbf{w}^{(\text{ball})}$ evaluates to $\mathbf{w}^{(\text{svm})}$ (see Section 2.4 for details) and can be calculated efficiently, thus, scaling to large learning problems.

Moreover, since we are only given a finite set of unlabeled examples to choose from, in general we will not be able to find an example which exactly satisfies the above-defined selection criterion based on some approximation of the center of mass, i.e., there is no unlabeled example $x \in \mathcal{U}$ such that $\langle \mathbf{w}^{(\text{bpm})}, \phi(x)\rangle_{\mathcal{F}} = 0$ or $\langle \mathbf{w}^{(\text{svm})}, \phi(x)\rangle_{\mathcal{F}} = 0$.

Given a *labeled* example (x, y), the directed distance of the hyperplane parameterized by $\mathbf{w}^{(\text{center})}$ evaluates to $y\langle \mathbf{w}^{(\text{center})}, \phi(x)\rangle_{\mathcal{F}}$, i.e., it evaluates to the margin of (x, y) with respect to $h_{\mathbf{w}^{(\text{center})}}$. It can be viewed as an approximate measure of the

2. More precisely, for this argument to hold, we have to consider an augmented *convex* version space as the intersection with the unit ball instead of the unit hypersphere. We will provide a more detailed convergence analysis and present a survey of related results in Chapter 8.

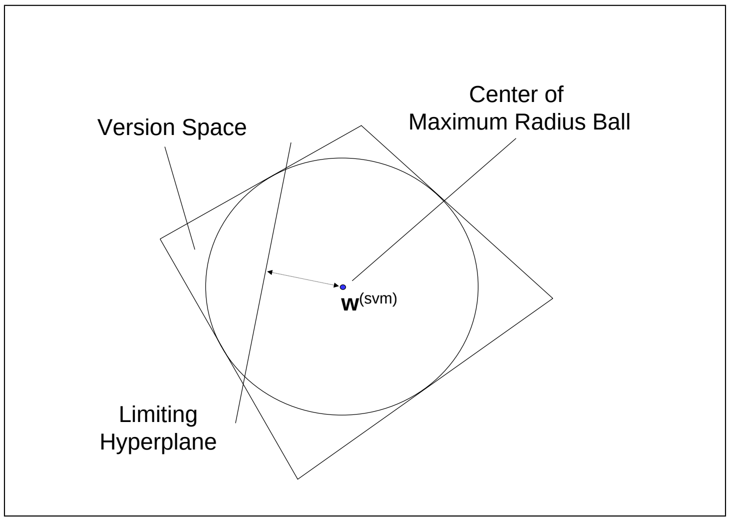


Figure 4.1 *Schematic view of the* Simple *selection strategy, which selects examples with minimum distance of the corresponding hyperplanes to the (approximate) center of mass of the version space. In this figure, the center of mass is approximated by the center of the maximum radius ball inscribable in version space.*

reduction of the volume, where lower values correspond to a higher reduction and, in particular, negative values correspond to the case where the smaller part of the version space remains (Tong and Koller, 2001c). Considering a best worst-case approach, we obtain minimization of $\max_{y \in \{-1,+1\}} y\langle \mathbf{w}^{(\mathrm{center})}, \phi(x)\rangle_{\mathcal{F}}$ as selection criterion, thus, the corresponding selection strategy is given by

$$(\mathcal{U},\mathcal{L}) \mapsto \operatorname*{argmin}_{x \in \mathcal{U}} \max_{y \in \{-1,+1\}} y\langle \mathbf{w}^{(\mathrm{center})}, \phi(x)\rangle_{\mathcal{F}} \quad (4.5)$$

$$= \operatorname*{argmin}_{x \in \mathcal{U}} |\langle \mathbf{w}^{(\mathrm{center})}, \phi(x)\rangle_{\mathcal{F}}|. \quad (4.6)$$

Note that the weight vector $\mathbf{w}^{(\mathrm{center})}$ on the right-hand side depends on the set of labeled examples $\mathcal{L}$. We will substitute the exact center of mass $\mathbf{w}^{(\mathrm{center})}$ by the approximation $\mathbf{w}^{(\mathrm{svm})}$ and $\mathbf{w}^{(\mathrm{bpm})}$, respectively, and refer to this selection strategy as Simple strategy (Tong and Koller, 2001c).

Simple Strategy

The computational complexity of the Simple selection strategy for approximating the center of mass of the version space is equivalent to the complexity of training a support vector machine and a Bayes point machine, respectively. Additionally, absolute distances to the approximation of the center of mass have to be computed for all unlabeled examples, which is equivalent to predicting class labels from a computational point of view. Apart from considerations of the theoretical order of complexity, the Simple selection strategy scales to large learning problems in practice and has been successfully evaluated on various benchmark problems, e.g., in the field of character recognition (Campbell et al., 2000), document classification (Tong and Koller, 2001c; Schohn and Cohn, 2000) and computational chemistry (Warmuth et al., 2002).

However, in time-critical applications, further modifications may be necessary for the reduction of the computational complexity. As the evaluation of the SIMPLE selection criterion is essentially dominated by distance calculations in the case of large pool sizes, a preceding step of random subsampling from the pool of unlabeled examples forms an efficient technique, which also applies to arbitrary selection strategies, for decreasing the demand for computational time. On the other hand, the size of the subsample influences the progress of the learning process, i.e., in general a larger pool size contributes to the performance of active learning as demonstrated in (Tong and Koller, 2001c). Thus, this parameter has to be chosen with a view to the considered setting as a trade-off between computational time and accuracy.

Subsampling

A more sophisticated means for reducing the computational complexity is to incorporate a technique proposed in (DeCoste, 2002). In order to speed up the classification process for kernel machines, DeCoste (2002) introduced a computational geometry method for which classification time becomes roughly proportional to inherent difficulty, i.e., classification time is inverse proportional to the distance from the classification hyperplane. More precisely, the suggested classification procedure generates monotonically convergent sequences of lower and upper bounds on the real-valued prediction and terminates if both of the bounds take either positive or negative values. In an analogous manner, we can expedite the evaluation process of the SIMPLE selection criterion by terminating the evaluation for a given unlabeled example once the bounds guarantee that the distance exceeds the current minimum, and we calculate exact distances only for those unlabeled examples which are guaranteed to achieve new minima. In contrast to the subsampling technique, the incremental bounding technique provides an exact, yet more efficient method which does not influence the result of the selection strategy.

Incremental Bounding

The Bayesian query-by-committee approach (Seung et al., 1992) considers the stream-based active learning model and determines whether to request class labels based on the disagreement within a set of Gibbs classifiers. The underlying objective is an efficient reduction of the volume of the version space. Under the assumption of a deterministic noise-free linear learning setting, the query-by-committee approach achieves an exponential reduction of the misclassification probability in the number of *labeled* training examples in the stream-based selective sampling setting (Freund et al., 1997; Fine et al., 2002).

In addition to the version space model, which has been considered in (Tong and Koller, 2001c) to justify the SIMPLE selection strategy, additional theoretical justifications for this approach are provided in (Campbell et al., 2000; Schohn and Cohn, 2000). Moreover, the theory of convex sets is an active field of research and recent results are promising to provide a more formal theoretical derivation of the margin as an approximate measure for the volume reduction. More concretely, Bertsimas and Vempala (2002) proved that for a convex set in isotropic position[3], any halfspace at distance t from the center of mass contains at least $\frac{1}{e} - t$ of its volume.

3. See Definition A.1.

4.2 Kernel Billiard Approach to Active Learning

In this section, we analyze potential shortcomings of (approximate) volume-based selection strategies in a particular linear learning setting and propose an improved binary selection strategy to address this problem. Since our approach is computationally demanding, we introduce a sophisticated subsampling technique which preselects a subset of unlabeled examples based on the less demanding volume-based SIMPLE strategy as candidates for our novel selection strategy in order to reduce the computational complexity. While the derivation of the proposed selection strategy is based on strong assumptions about the underlying probability distributions, one of the main contributions of this section is a detailed analysis on the dependence of version space volume and misclassification probability in the considered setting.

Let us assume that we are given a fixed feature map $\phi : \mathcal{X} \to \mathcal{F} = \mathbb{R}^N$ with $N \in \mathbb{N}$ which embeds the data into the real Euclidean feature space $\mathcal{F}$ (endowed with the canonical dot product). Moreover, we assume that examples (x, y) are labeled according to a teacher kernel classifier $h_{\mathrm{w}^*} : \mathcal{X} \to \{-1, +1\}$ such that $y \overset{\text{def}}{=} h_{\mathrm{w}^*}(x)$ for all $x \in \mathcal{X}$. Under the assumption of a spherical Gaussian distribution of embedded input patterns, i.e., the probability distribution of $\mathrm{P}^{\phi(\mathrm{X})}$ is governed by the probability density function $f^{\phi(\mathrm{X})} : \mathcal{F} \to \mathbb{R}$, $\bar{x} \mapsto \frac{2}{\sqrt{\pi}} e^{-\|\bar{x}\|^2}$, where P^{X} denotes the probability distribution of patterns in input space $\mathcal{X}$, the misclassification probability of the linear classifier h_{w} corresponding to the weight vector $\mathbf{w} \in \mathcal{W}$ is given by (see (Herbrich et al., 2001))

$$\begin{aligned}
\mathrm{P}\big(h_{\mathrm{w}^*}(\mathrm{X}) \neq h_{\mathrm{w}}(\mathrm{X})\big) &= \mathrm{E}\big(l_{0-1}(h_{\mathrm{w}^*}(\mathrm{X}), h_{\mathrm{w}}(\mathrm{X}))\big) && (4.7)\\
&= \mathrm{E}\big(l_{0-1}(\mathrm{sign}(\langle \mathbf{w}^*, \phi(\mathrm{X})\rangle), \mathrm{sign}(\langle \mathbf{w}, \phi(\mathrm{X})\rangle))\big) && (4.8)\\
&= \frac{\arccos(\langle \mathbf{w}^*, \mathbf{w}\rangle)}{\pi} && (4.9)
\end{aligned}$$

where l_{0-1} denotes the 0-1-loss as defined in (1.3). Let us consider the following three-dimensional example which illustrates that the reduction of the volume of the version space does not guarantee a reduction of the misclassification probability in this model in general (see Figure 4.2): Assume we are given a strictly increasing sequence of sets of labeled examples $(\mathcal{L}_i)_{i \geq 0}$, such that $\mathcal{L}_0 = \emptyset$ and $\mathcal{L}_i = \{(x_1, y_1), \ldots, (x_i, y_i)\}$ where

Example

$$y_i \phi(x_i) = \begin{pmatrix} 0 \\ \cos \varphi_i \\ \sin \varphi_i \end{pmatrix} \quad \text{with} \quad \varphi_i \overset{\text{def}}{=} \pi - \frac{\pi}{2^{i-1}} \quad \text{for } i \geq 1. \qquad (4.10)$$

Note that there exist at least two alternative options for every example (x_i, y_i) depending on the choice of the class label y_i and the corresponding sign of the embedded input pattern which are equivalent in this context. Additional options may result from the underlying embedding of the input space via ϕ.

In an analogous manner to the previous section, we define $\mathcal{V}_i$ as the version space corresponding to the set of labeled examples $\mathcal{L}_i$. It is readily verified that each labeled

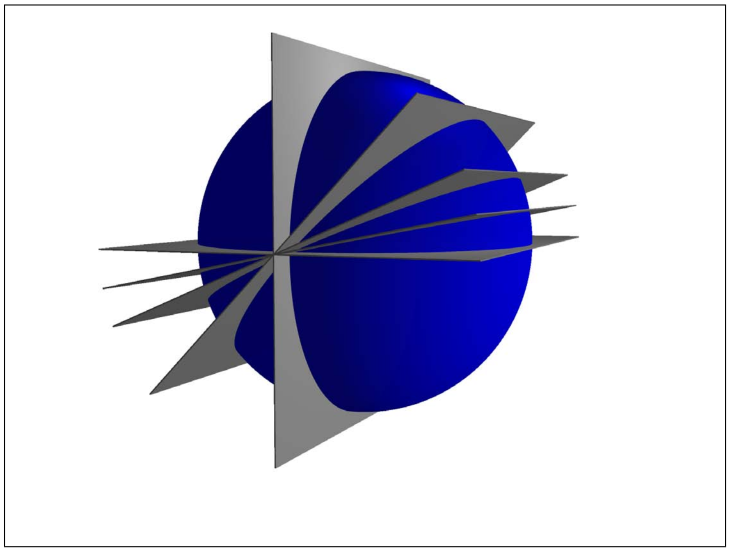


Figure 4.2 *Despite the exponential reduction of the volume of the version space, the misclassification probability of a classifier in the version space can be arbitrarily close to* 1.

example imposes a constraint on the weight space such that the volume of the preceding version space is halved, i.e.,

$$\mathrm{vol}(\mathcal{V}_i) = \frac{1}{2^i}\,\mathrm{vol}(\mathcal{W}) = \frac{\pi}{2^{i-2}} \quad \text{for } i \geq 0 \tag{4.11}$$

where $\mathrm{vol}(\mathcal{W}) = 4\pi$. Furthermore, the normalized center of mass is given by

$$\mathbf{w}_i^{(\text{center})} = \begin{pmatrix} 0 \\ \sin\varphi_{i+1} \\ -\cos\varphi_{i+1} \end{pmatrix} \tag{4.12}$$

and it holds that

$$y_{i+1}\langle \mathbf{w}_i^{(\text{center})}, \phi(x_{i+1})\rangle = 0 \quad \text{for } i \geq 1. \tag{4.13}$$

Thus, the elements of the sequence of patterns minimize the SIMPLE selection criterion based on the exact center of mass and indeed achieve an exponential reduction of the volume of the version space in the number of labeled examples.

We consider the weight vectors $\mathbf{v} = (1,0,0)^\top$ and $\mathbf{v}' = (-1,0,0)^\top$ which are contained in the compactification of $\mathcal{V}_i$ for all $i \in \mathbb{N}$ and conclude that

$$\sup_{\mathbf{w},\mathbf{w}' \in \mathcal{V}_i} \frac{\arccos\langle \mathbf{w}, \mathbf{w}'\rangle}{\pi} = 1 \tag{4.14}$$

for all $i \in \mathbb{N}$. As a direct consequence of this observation, it is straightforward to construct sequences of examples of arbitrary length n for any $\varepsilon > 0$ in a query learning setting, such that there exists a classifier in $\mathcal{V}_i$ for all $1 \leq i \leq n$ with misclassification probability greater than $1 - \varepsilon$, despite that the SIMPLE selection criterion evaluates to its global optimum in every step and the sequence of examples achieves an exponential reduction of the volume of the version space. Hence, reduction of the volume of the version space is not a sufficient criterion to guarantee the reduction of the misclassification probability in this setting.[4] Intuitively, exclusively requesting class labels associated with the constructed category of examples does not provide any information about the first component of the teacher weight vector and therefore the supremum of the angles between weight vectors in version space, which determines the worst-case misclassification probability, does not decrease. Note, that this construction considers the fixed feature space dimension $N = 3$ and therefore does not vary the dimension of the problem in dependence of the length of the sequence. In the case of increasing dimension in the length of the sequence, the aforementioned result is trivial.

In the absence of further knowledge, it is reasonable to assume that the weight vector $\mathbf{w}^*$ of the teacher kernel machine is drawn from a uniform distribution over $\mathcal{W}$ (Herbrich et al., 2001). If we denote the teacher kernel machine by the random variable W^*, then for any $\mathbf{w} \in \mathcal{W}$ the expected misclassification probability with respect to the random variables X and W^* evaluates to

$$\mathrm{E}\big(l_{0-1}(\mathrm{sign}(\langle \mathrm{W}^*, \phi(\mathrm{X})\rangle), \mathrm{sign}(\langle \mathbf{w}, \phi(\mathrm{X})\rangle))\big) = \int \frac{\arccos(\langle \mathbf{w}^*, \mathbf{w}\rangle)}{\pi} \, d\mathrm{P}^{\mathrm{W}^*}(\mathbf{w}^*). \tag{4.15}$$

A similar line of reasoning as for the worst-case scenario reveals that volume reduction does not necessarily yield a reduction of the expected misclassification probability for every classifier in the version space. We will sketch the formal proof by showing that the expected misclassification probability (with respect to the random variables X and W^*) of the classifier corresponding to the weight vector $\mathbf{v} = (1, 0, 0)^\top$, which is contained in the compactification of $\mathcal{V}_i$ for all $i \in \mathbb{N}$, evaluates to the constant $\frac{1}{2}$.

Consider the sequence of labeled sets $\mathcal{L}_1, \ldots, \mathcal{L}_n$ as defined above and assume without loss of generality that $\mathbf{w}^* \in \mathcal{V}_i$ for $1 \leq i \leq n$, in other words, $\mathbf{w}^*$ is sampled from a uniform distribution over $\mathcal{V}_n$. Furthermore, the posterior distribution of W^* in selection step i, which is denoted by $\mathrm{P}^{\mathrm{W}_i^*}$, is the restriction of the uniform distribution over $\mathcal{W}$ to the version space $\mathcal{V}_i$. Therefore, in step i the expected misclassification probability of a classifier $\mathbf{w} \in \mathcal{W}$ is given by

$$\mathrm{E}\big(l_{0-1}(\mathrm{sign}(\langle \mathrm{W}_i^*, \phi(\mathrm{X})\rangle), \mathrm{sign}(\langle \mathbf{w}, \phi(\mathrm{X})\rangle))\big) = \int \frac{\arccos(\langle \mathbf{w}^*, \mathbf{w}\rangle)}{\pi} \, d\mathrm{P}^{\mathrm{W}_i^*}(\mathbf{w}^*) \tag{4.16}$$

$$= \frac{1}{\mathrm{vol}(\mathcal{V}_i)} \int_{\mathcal{V}_i} \frac{\arccos(\langle \mathbf{w}^*, \mathbf{w}\rangle)}{\pi} \, d\mathbf{w}^*. \tag{4.17}$$

4. Freund et al. (1997) considered a related setting where volume reduction is not a sufficient criterion to guarantee the reduction of the misclassification probability.

It holds for the expected misclassification probability of the classifier h_{v} that

$$\begin{aligned}
&\mathrm{E}\big(l_{0-1}(\mathrm{sign}(\langle \mathrm{W}_i^*, \phi(\mathrm{X})\rangle), \mathrm{sign}(\langle \mathbf{v}, \phi(\mathrm{X})\rangle))\big) && (4.18)\\
&=\frac{1}{\mathrm{vol}(\mathcal{V}_i)} \int_{\mathcal{V}_i} \frac{\arccos(\langle \mathbf{w}^*, \mathbf{v}\rangle)}{\pi}\, d\mathbf{w}^* && (4.19)\\
&=\frac{1}{\mathrm{vol}(\mathcal{V}_i)} \int_0^{\pi} \int_{\frac{\pi}{2}-\frac{\pi}{2^{i-1}}}^{\frac{\pi}{2}} \sin(\varphi_1) \frac{\arccos(\cos(\varphi_1))}{\pi}\, d\varphi_2 d\varphi_1 && (4.20)\\
&=\frac{1}{2^{i-1}\,\mathrm{vol}(\mathcal{V}_i)} \int_0^{\pi} \sin(\varphi_1)\, \varphi_1\, d\varphi_1 && (4.21)\\
&=\frac{\pi}{2^{i-1}\,\mathrm{vol}(\mathcal{V}_i)} = \frac{\pi}{2^{i-1}\frac{\pi}{2^{i-2}}} = \frac{1}{2}. && (4.22)
\end{aligned}$$

Note that (4.20) is a transformation by means of polar coordinates in $\mathbb{R}^3$ (Elstrodt, 1996, p. 206),

$$\Phi : (0, \pi) \times (0, 2\pi) \to \mathcal{W} \backslash \{\mathbf{w} \in \mathcal{W} \mid [\mathbf{w}]_2 \geq 0, [\mathbf{w}]_3 = 0\} \qquad (4.23)$$

$$\begin{pmatrix} \varphi_1 \\ \varphi_2 \end{pmatrix} \mapsto \begin{pmatrix} \cos\varphi_1 \\ \sin\varphi_1 \cos\varphi_2 \\ \sin\varphi_1 \sin\varphi_2 \end{pmatrix}. \qquad (4.24)$$

We have argued that in the given setting, volume reduction is not a sufficient criterion to guarantee both the reduction of the worst-case and the expected misclassification probability. In the following, we will present a novel selection strategy which incorporates (4.17) as the crucial quantity in this setting for the reduction of the expected misclassification probability. Moreover, we will discuss the dependence between the proposed selection criterion and volume reduction in detail.

We consider the following approach to approximate this quantity: Using the kernel billiard algorithm described in Section 2.5, we calculate the M points $\mathbf{w}_1, \ldots, \mathbf{w}_M$ defining the trajectory of the billiard (the collision points) and project them onto the unit hypersphere by $\mathbf{w}'_j \overset{\mathrm{def}}{=} \frac{\mathrm{w}_j}{\|\mathrm{w}_j\|} \in \mathcal{V}_i$ for $j = 1, \ldots, M$. We assume a constant probability density on the projected trajectory and average the misclassification probability over the trajectory on the unit hypersphere:

$$\begin{aligned}
&\mathrm{E}\big(l_{0-1}(\mathrm{sign}(\langle \mathrm{W}_i^*, \phi(\mathrm{X})\rangle), \mathrm{sign}(\langle \mathbf{w}, \phi(\mathrm{X})\rangle))\big) && (4.25)\\
&\approx \frac{1}{\pi} \sum_{j=1}^{M-1} \left[\frac{\arccos(\langle \mathbf{w}'_j, \mathbf{w}'_{j+1}\rangle)}{\sum_{l=1}^{M-1} \arccos(\langle \mathbf{w}'_l, \mathbf{w}'_{l+1}\rangle)} \cdot \frac{\arccos(\langle \mathbf{w}'_j, \mathbf{w}\rangle) + \arccos(\langle \mathbf{w}'_{j+1}, \mathbf{w}\rangle)}{2} \right]. && (4.26)
\end{aligned}$$

As mentioned in Section 2.5, a direct Monte Carlo sampling approach is impractical in this setting due to the computational complexity.

For the selection of new examples, we consider a best worst-case scenario for each unlabeled example $x \in \mathcal{U}$ in the following manner: The currently labeled set of patterns is augmented by the negative example $(x, -1)$ and the kernel billiard algorithm is employed to calculate an approximation $\mathbf{w}_-^{(\mathrm{bpm})}$ of the Bayes point. In an analogous manner, an approximation $\mathbf{w}_+^{(\mathrm{bpm})}$ is calculated based on the currently labeled dataset augmented by the positive example $(x, +1)$. Each augmented training set corresponds to a new version space. The new expected misclassification probabilities e^- and e^+ corresponding to $\mathbf{w}_-^{(\mathrm{bpm})}$ and $\mathbf{w}_+^{(\mathrm{bpm})}$, respectively, in the case of a negative and a positive class label are estimated by (4.26), i.e.,

$$e^- \overset{\mathrm{def}}{=} \frac{1}{\pi} \sum_{j=1}^{M-1} \left[\frac{\arccos(\langle \mathbf{w}'_j, \mathbf{w}'_{j+1} \rangle)}{\sum_{l=1}^{M-1} \arccos(\langle \mathbf{w}'_l, \mathbf{w}'_{l+1} \rangle)} \right. \tag{4.27}$$

$$\left. \cdot \frac{\arccos(\langle \mathbf{w}'_j, \mathbf{w}_-^{(\mathrm{bpm})} \rangle) + \arccos(\langle \mathbf{w}'_{j+1}, \mathbf{w}_-^{(\mathrm{bpm})} \rangle)}{2} \right] \tag{4.28}$$

and e^+ is defined in an analogous manner with respect to $\mathbf{w}_+^{(\mathrm{bpm})}$. The next pattern to be labeled is selected according to minimum estimated worst-case expected misclassification probability with respect to the unknown binary class label. Best worst-case approximations of expected misclassification probabilities have been successfully employed in active learning research to derive heuristic selection strategies (Tong and Koller, 2001c). Hence, this selection strategy is given by

KB Strategy

$$(\mathcal{U}, \mathcal{L}) \mapsto \underset{x \in \mathcal{U}}{\operatorname{argmin}} \max \{e^-, e^+\} \tag{4.29}$$

where the constants in the definition of e^- and e^+ can be omitted. We refer to this selection strategy as KB(M) where the optional parameter M denotes the number of bounces used to estimate expected misclassification probabilities.

The KB selection strategy is computationally very demanding for sufficiently sized pools of unlabeled examples since for each unlabeled example the kernel billiard algorithm is executed twice or four times depending on whether or not the trajectory points for the estimation of the Bayes point and for the estimation of the expected misclassification probabilities are calculated in separate runs. To reduce the computational effort, we propose a two-layered approach where a subsample is selected from the complete pool of unlabeled examples by the fast volume reduction heuristic SIMPLE in stage one: We calculate the Bayes point estimate $\mathbf{w}^{(\mathrm{bpm})}$ based on the set of labeled examples and sort all unlabeled patterns x in ascending order with respect to $|\langle \mathbf{w}^{(\mathrm{bpm})}, \phi(x) \rangle|$. Then, the subsample is selected as a predefined number of patterns from the initial part of the ordered set. This subsampling approach is based on the observation that low version space volume is a *necessary* precondition for low expected misclassification probability, however, it is not a *sufficient* condition in the given setting as discussed above. We will provide a formal derivation in the following.

KBSUB Strategy

Let us define a ball for $0 \leq \gamma \leq \pi$ in the weight space $\mathcal{W} = \{\mathbf{w}' \in \mathcal{F} \mid \|\mathbf{w}'\| = 1\}$ which is centered at $\mathbf{w}$ by

$$\mathcal{Q}_\gamma(\mathbf{w}) \overset{\mathrm{def}}{=} \{\mathbf{w}' \in \mathcal{W} \mid \langle \mathbf{w}', \mathbf{w} \rangle > \cos(\gamma)\}. \tag{4.30}$$

It was shown in (Herbrich and Graepel, 2002) that the ratio between the volumes of

$\mathcal{Q}_\gamma(\mathbf{w})$ and $\mathcal{W}$ can be expressed as

$$\frac{\mathrm{vol}(\mathcal{Q}_\gamma(\mathbf{w}))}{\mathrm{vol}(\mathcal{W})} = \frac{\int_0^\gamma \sin^{N-2}(\theta)\, d\theta}{\int_0^\pi \sin^{N-2}(\theta)\, d\theta} \quad \text{for } 0 \le \gamma \le \pi. \tag{4.31}$$

For $N \ge 2$ and $0 \le \gamma \le \frac{\pi}{2}$, the volume ratio can be upper bounded by

$$\frac{\mathrm{vol}(\mathcal{Q}_\gamma(\mathbf{w}))}{\mathrm{vol}(\mathcal{W})} = \frac{\int_0^\gamma \sin^{N-2}(\theta)\, d\theta}{\int_0^\pi \sin^{N-2}(\theta)\, d\theta} \tag{4.32}$$

$$= \underbrace{\frac{\Gamma(\frac{N}{2})}{\sqrt{\pi}\,\Gamma(\frac{N-1}{2})}}_{=:c_1^{(N)}} \int_0^\gamma \sin^{N-2}(\theta)\, d\theta \tag{4.33}$$

$$\le c_1^{(N)}\, \gamma \sin^{N-2}(\gamma). \tag{4.34}$$

Hence,

$$\mathrm{vol}(\mathcal{V}_i) = \mathrm{vol}(\mathcal{W}) \cdot \frac{\mathrm{vol}(\mathcal{V}_i \cap \mathcal{Q}_\gamma(\mathbf{w})^C) + \mathrm{vol}(\mathcal{V}_i \cap \mathcal{Q}_\gamma(\mathbf{w}))}{\mathrm{vol}(\mathcal{W})} \tag{4.35}$$

$$\le \mathrm{vol}(\mathcal{W}) \left(\frac{\mathrm{vol}(\mathcal{V}_i \cap \mathcal{Q}_\gamma(\mathbf{w})^C)}{\mathrm{vol}(\mathcal{W})} + c_1^{(N)}\, \gamma \sin^{N-2}(\gamma) \right) \tag{4.36}$$

$$\le \mathrm{vol}(\mathcal{W}) \left(\frac{1}{\mathrm{vol}(\mathcal{V}_i)} \mathrm{vol}(\mathcal{V}_i \cap \mathcal{Q}_\gamma(\mathbf{w})^C) + c_1^{(N)}\, \gamma \sin^{N-2}(\gamma) \right) \tag{4.37}$$

$$\le \mathrm{vol}(\mathcal{W}) \left(\frac{\pi}{\gamma} \cdot \frac{1}{\mathrm{vol}(\mathcal{V}_i)} \int_{\mathcal{V}_i \cap \mathcal{Q}_\gamma(\mathrm{w})^C} \frac{\arccos(\langle \mathbf{w}^*, \mathbf{w} \rangle)}{\pi}\, d\mathbf{w}^* + c_1^{(N)}\, \gamma \sin^{N-2}(\gamma) \right) \tag{4.38}$$

$$\le \mathrm{vol}(\mathcal{W}) \left(\frac{\pi}{\gamma} \cdot \frac{1}{\mathrm{vol}(\mathcal{V}_i)} \int_{\mathcal{V}_i} \frac{\arccos(\langle \mathbf{w}^*, \mathbf{w} \rangle)}{\pi}\, d\mathbf{w}^* + c_1^{(N)}\, \gamma \sin^{N-2}(\gamma) \right) \tag{4.39}$$

Note that (4.38) holds since $\arccos(\langle \mathbf{w}^*, \mathbf{w} \rangle) \ge \gamma$ for $\mathbf{w}, \mathbf{w}^* \in \mathcal{V}_i \cap \mathcal{Q}_\gamma(\mathbf{w})^C$ as a direct consequence of the definition of $\mathcal{Q}_\gamma(\mathbf{w})$. Now, we define $e(\mathbf{w})$ as the expected misclassification probability of the linear classifier h_w with respect to the posterior distribution $\mathrm{P}^{\mathrm{W}_i^*}$,

$$e(\mathbf{w}) \overset{\mathrm{def}}{=} \int \frac{\arccos(\langle \mathbf{w}^*, \mathbf{w} \rangle)}{\pi}\, d\mathrm{P}^{\mathrm{W}_i^*}(\mathbf{w}^*) = \frac{1}{\mathrm{vol}(\mathcal{V}_i)} \int_{\mathcal{V}_i} \frac{\arccos(\langle \mathbf{w}^*, \mathbf{w} \rangle)}{\pi}\, d\mathbf{w}^*, \tag{4.40}$$

and fix $\gamma \overset{\mathrm{def}}{=} \sqrt{e(\mathbf{w})}$. Then, it follows that $\gamma \le 1 < \frac{\pi}{2}$ and we conclude:

$$\mathrm{vol}(\mathcal{V}_i) \le \mathrm{vol}(\mathcal{W}) \left(\pi \sqrt{e(\mathbf{w})} + c_1^{(N)} \sqrt{e(\mathbf{w})} \sin^{N-2}(\sqrt{e(\mathbf{w})}) \right). \tag{4.41}$$

Thus, the volume of the version space can be bounded in terms of the expected misclassification probability of any classifier h_w with respect to the posterior distribution. From a different point of view, low version space volume is a *necessary* precondition for low expected misclassification probability. As a consequence, this motivates the choice of a volume-based selection strategy for preselecting a promising subset of unlabeled examples. Subsequent to the selection of a subsample according to the volume reduction criterion, the novel strategy $\textsc{Kb}(M)$ selects the next pattern from the subsample. This two-layered selection strategy is referred to as $\textsc{KbSub}(S, M)$ and parameterized by the size of the subsample S and the number of bounces M used to estimate the expected misclassification probabilities. The computational effort for the selection process can be controlled by the size of the subsample. More precisely, as the dominating factor of the two-layered subsampling approach is the $\textsc{Kb}(M)$ strategy, the overall computational time depends on S in an approximately proportional manner.

Computational Time

4.3 Experiments

We have conducted a set of active learning experiments on both artificial data and real-world data to investigate the efficiency of the proposed selection strategies. While the number of bounces in the kernel billiard algorithm was fixed to 1000 whenever the center of mass estimate was used as a classifier, we evaluated the $\textsc{Kb}(M)$ and the $\textsc{KbSub}(S, M)$ selection strategies based on $M = 1000, 5000$ collision points. Random selection ($\textsc{Random}$) of new examples was considered as a baseline strategy and random selection of examples from the preselected subsample (denoted by $\textsc{RndSub}(S)$) was included in the experimental setup as an additional selection strategy.

4.3.1 Artificial Data

The first set of experiments was conducted on data which was generated according to the examined learning setting. More precisely, we considered a noise-free scenario where the teacher weight vectors $\mathbf{w}^*$ were sampled from a uniform distribution over the $N-$dimensional unit hypersphere. Then, a set of patterns was sampled uniformly from the $N-$dimensional unit hypersphere and labeled according to $\mathbf{x} \mapsto \text{sign}(\langle \mathbf{w}^*, \mathbf{x} \rangle)$. Setting $N = 10$, we generated a dataset consisting of 2000 examples. The dataset was randomly split 100 times into a training and a test set of equal size. The initially labeled sets of examples consisted of one positive and one negative example randomly drawn from the training sets. While new training examples were selected from the remaining training sets (associated class labels were masked out prior to selection), the classification accuracy was estimated on the test sets after every selection step. We fixed the final number of labeled examples to 20 and averaged the results over all 100 runs of random initialization and separation into training and test sets.

Detailed experimental results of our novel strategies with various parameter settings are presented in Tables 4.1 and 4.2. Our experiments demonstrate that both the $\textsc{Kb}(1000)$ and the $\textsc{KbSub}(S, M)$ for $S = 100, 250$ and $M = 1000, 5000$ significantly outperform the $\textsc{Simple}$, the $\textsc{Random}$ and the $\textsc{RndSub}(S)$ selection strategies. In particular, for all choices of S and M the $\textsc{KbSub}(S, M)$ strategy improves on the $\textsc{RndSub}(S)$ selection strategy, thus, demonstrating that the observed increase in accuracy is not a consequence of a randomization effect. Furthermore, pathological sequences of selected examples as constructed in the previous section do not occur frequently as indicated by the empirical results, nevertheless, it is possible to improve on the Simple strategy in this setting.

Moreover, Table 4.2 provides experimental results of the KbSub(100,1000) strategy which were obtained by running the kernel billiard algorithm without computing an orthonormal basis. These results indicate that this modification, which was proposed in Section 2.5, is an essential component of the envisaged selection strategy.

The differences in classification accuracy between the Kb(M) and the KbSub(S, M) selection strategy based on the different parameter settings are negligible. Qualitatively similar results were obtained for different choices of the input dimension N. As the initial size of the pool of unlabeled examples was 998, the computational time when using the KbSub(S, M) strategy with $S = 100, 250$ for the size of the subsample was reduced by a factor of approximately $\frac{1}{10}$ and $\frac{1}{4}$, respectively, in comparison to the Kb(M) strategy. Thus, as we anticipated on account of our theoretical analysis, the experimental results demonstrate that the two-layered subsampling approach substantially decreases the computational complexity without a concomitant loss of classification accuracy.

4.3.2 Real-World Data

The high computational complexity, even for the KbSub(S, M) strategy, prohibits a comprehensive study of the proposed selection strategies on real-world data. Nevertheless, we conducted a preliminary set of experiments on the well-studied *letter* dataset from the UCI repository of machine learning databases (Blake and Merz, 1998) in addition to the experiments on artificial data to investigate if the KbSub(S, M) selection strategy might be employed as a general-purpose strategy beyond the considered linear setting. We considered the binary problem *class* 0 *-versus-all* on this multiclass dataset. This dataset consists of 20000 examples with $N = 16$ input features and was split into training and test sets in an analogous manner to the artificial setting. The initial training sets consisted of one randomly chosen positive and one negative example. The classification accuracy was evaluated after each selection step. Using a kernelized billiard algorithm, we chose a linear kernel with $C = 10000$ and L2-loss, i.e., the soft margin modification of the kernel described in Section 2.4. Furthermore, the patterns were normalized to unit length in feature space by $k^*(x_i, x_j) \overset{\text{def}}{=} k(x_i, x_j)/\sqrt{k(x_i, x_i)\, k(x_j, x_j)}$. As a consequence of the large pool size of 9998 examples, experimental results are given only for the KbSub(S, M) strategy, which reduced the computational time by a factor of approximately $\frac{1}{100}$ and $\frac{1}{40}$, respectively, depending on the choice of S, in comparison to the Kb(M) strategy.

Although the assumption of uniformly distributed patterns does not hold for this dataset, Table 4.3 demonstrates that there is a significant increase in the attained level of classification accuracy for the KbSub(S, M) (with $S = 100$ and $M = 1000, 5000$) in comparison to the Simple selection strategy in the initial part of the learning process, while all active selection strategies approach approximately the same level of accuracy at the end of the experiment. Furthermore, with the number of bounces M increasing, we obtained slightly more stable results.

4.4 Discussion

We reviewed a class of selection strategies which aim at reducing the volume of the version space by means of approximating the center of mass and analyzed a particular linear learning setting where these approaches exhibit some potential shortcomings. To address this problem, we proposed an improved binary selection strategy which achieves a significant increase in classification accuracy in comparison to the SIMPLE volume-based selection strategy. However, this selection strategy is computationally demanding and therefore we introduced a sophisticated subsampling technique for the reduction of the computational complexity which preselects a subset of unlabeled examples based on a less demanding volume-based strategy as candidates for our novel selection strategy. From a theoretical point of view, the minimization of the volume of the version space is a *necessary* precondition for the minimization of the improved selection criterion. As we anticipated on account of our theoretical analysis, the experimental results demonstrate that the two-layered subsampling approach substantially decreases the computational complexity without a concomitant loss of classification accuracy. Beyond the considered linear setting, experimental results on a real-world dataset demonstrate an increase in the attained level of accuracy in the initial part of the learning process, even though this dataset does not match the theoretical model. Besides a violation of the assumptions about the underlying probability model, we note that in pool-based active learning it is only possible to approximately satisfy a selection criterion in general due to the finite number of candidate patterns. Therefore, we hypothesize that this property contributes to the increased efficiency in the initial part with the difference decreasing later on.

Since the computational complexity of the subsampling approach is still very high, we had to restrict our experimental setting to a small number of labeled examples. With more computational power being available, we plan to extend our experiments to a more realistic scale in the future in order to investigate, if the proposed selection strategies, although being derived under strong assumptions about the underlying probability distribution, can be employed as general-purpose selection strategies in practice.

Moreover, a promising direction for developing a computationally more efficient implementation is to incorporate an idea proposed by Graepel et al. (2000) where computationally expensive resampling of the version space is avoided by reusing one precomputed trajectory (and the set of collision points, respectively) for the evaluation of the selection criterion. The basic idea is to calculate a set of line segments defined by the precomputed trajectory and the halfspace corresponding to a given example which, from a computational point of view, is less expensive than computing a new trajectory.

Apart from practical issues, the considered setting sheds light on potential shortcomings of volume-based strategies from a theoretical perspective. We provided a detailed analysis of these drawbacks and proposed a promising direction of further research.

m	Random	Simple	RndSub(100)	KbSub(100, 1000)	KbSub(100, 5000)
2	0.6180 ±0.0068	0.6213 ±0.0062	0.6086 ±0.0062	0.6248 (0.3445) ±0.0061	0.6335 (0.0943) ±0.0068
3	0.6509 ±0.0070	0.6498 ±0.0061	0.6386 ±0.0060	0.6548 (0.2690) ±0.0055	0.6574 (0.2029) ±0.0068
4	0.6787 ±0.0068	0.6707 ±0.0063	0.6664 ±0.0061	0.6804 (0.1187) ±0.0053	0.6815 (0.1154) ±0.0065
5	0.6938 ±0.0066	0.6937 ±0.0066	0.6889 ±0.0059	0.7003 (0.2225) ±0.0054	0.6991 (0.2761) ±0.0061
6	0.7061 ±0.0062	0.7131 ±0.0068	0.7102 ±0.0059	0.7255 (0.0768) ±0.0054	0.7218 (0.1640) ±0.0059
7	0.7198 ±0.0060	0.7336 ±0.0066	0.7343 ±0.0055	0.7453 (0.0795) ±0.0049	0.7417 (0.1790) ±0.0057
8	0.7349 ±0.0060	0.7505 ±0.0066	0.7499 ±0.0053	0.7661 (0.0288) ±0.0048	0.7634 (0.0677) ±0.0055
9	0.7485 ±0.0057	0.7661 ±0.0065	0.7706 ±0.0047	0.7842 (0.0120) ±0.0046	0.7896 (0.0023) ±0.0050
10	0.7597 ±0.0054	0.7834 ±0.0064	0.7877 ±0.0042	0.8020 (0.0093) ±0.0045	0.8043 0.0029) ±0.0039
11	0.7761 ±0.0058	0.8016 ±0.0059	0.8067 ±0.0039	0.8200 (0.0062) ±0.0042	0.8194 (0.0059) ±0.0036
12	0.7878 ±0.0057	0.8159 ±0.0056	0.8239 ±0.0043	0.8323 (0.0093) ±0.0041	0.8337 (0.0034) ±0.0034
13	0.7972 ±0.0056	0.8263 ±0.0051	0.8331 ±0.0039	0.8466 (0.0008) ±0.0038	0.8459 (0.0007) ±0.0032
14	0.8073 ±0.0053	0.8417 ±0.0045	0.8413 ±0.0043	0.8603 (0.0006) ±0.0034	0.8570 (0.0036) ±0.0033
15	0.8191 ±0.0051	0.8507 ±0.0041	0.8510 ±0.0042	0.8713 (0.0000) ±0.0031	0.8712 (0.0000) ±0.0030
16	0.8277 ±0.0051	0.8601 ±0.0038	0.8611 ±0.0038	0.8817 (0.0000) ±0.0033	0.8795 (0.0000) ±0.0030
17	0.8344 ±0.0050	0.8742 ±0.0035	0.8746 ±0.0033	0.8899 (0.0003) ±0.0028	0.8887 (0.0006) ±0.0028
18	0.8406 ±0.0047	0.8839 ±0.0031	0.8841 ±0.0032	0.8955 (0.0027) ±0.0027	0.8994 (0.0001) ±0.0027
19	0.8447 ±0.0047	0.8895 ±0.0030	0.8942 ±0.0030	0.9030 (0.0004) ±0.0026	0.9048 (0.0000) ±0.0023
20	0.8476 ±0.0049	0.8985 ±0.0028	0.9028 ±0.0027	0.9111 (0.0004) ±0.0024	0.9101 (0.0008) ±0.0022

Table 4.1 *This table shows the averaged classification accuracy results on the artificial dataset with $N = 10$ and the corresponding standard errors of the mean. p-values of a $t-$test for the null hypothesis that the given strategy for the given number of labeled examples m achieves accuracy less than or equal to the* Simple *strategy are stated in brackets.*

m	KbSub(250, 1000)	KbSub(250, 5000)	Kb(1000)	KbSub(100, 1000)*
2	0.6227 (0.4368) ±0.0063	0.6268 (0.2808) ±0.0072	0.6201 (0.5496) ±0.0072	0.6283 (0.2166) ±0.0064
3	0.6547 (0.2770) ±0.0057	0.6594 (0.1410) ±0.0066	0.6516 (0.4191) ±0.0064	0.6552 (0.2735) ±0.0066
4	0.6763 (0.2538) ±0.0056	0.6855 (0.0519) ±0.0065	0.6756 (0.2905) ±0.0064	0.6750 (0.3185) ±0.0067
5	0.6978 (0.3213) ±0.0057	0.7036 (0.1394) ±0.0062	0.6977 (0.3332) ±0.0062	0.6962 (0.3893) ±0.0059
6	0.7211 (0.1803) ±0.0055	0.7295 (0.0355) ±0.0060	0.7240 (0.1128) ±0.0060	0.7120 (0.5476) ±0.0057
7	0.7460 (0.0756) ±0.0055	0.7471 (0.0660) ±0.0059	0.7493 (0.0346) ±0.0055	0.7251 (0.8369) ±0.0055
8	0.7664 (0.0345) ±0.0056	0.7662 (0.0302) ±0.0050	0.7663 (0.0280) ±0.0049	0.7417 (0.8377) ±0.0059
9	0.7927 (0.0006) ±0.0049	0.7948 (0.0002) ±0.0044	0.7868 (0.0050) ±0.0046	0.7585 (0.7990) ±0.0064
10	0.8102 (0.0003) ±0.0043	0.8077 (0.0007) ±0.0040	0.8057 (0.0026) ±0.0046	0.7647 (0.9826) ±0.0061
11	0.8227 (0.0018) ±0.0040	0.8235 (0.0014) ±0.0041	0.8224 (0.0024) ±0.0042	0.7718 (0.9997) ±0.0062
12	0.8361 (0.0016) ±0.0038	0.8338 (0.0033) ±0.0034	0.8349 (0.0031) ±0.0040	0.7858 (0.9999) ±0.0059
13	0.8470 (0.0005) ±0.0035	0.8463 (0.0006) ±0.0033	0.8462 (0.0010) ±0.0037	0.7972 (0.9998) ±0.0062
14	0.8596 (0.0008) ±0.0033	0.8570 (0.0037) ±0.0034	0.8576 (0.0028) ±0.0034	0.8057 (1.0000) ±0.0060
15	0.8705 (0.0001) ±0.0029	0.8681 (0.0007) ±0.0035	0.8683 (0.0005) ±0.0034	0.8106 (1.0000) ±0.0060
16	0.8779 (0.0001) ±0.0030	0.8779 (0.0003) ±0.0034	0.8764 (0.0005) ±0.0032	0.8225 (1.0000) ±0.0060
17	0.8891 (0.0006) ±0.0029	0.8877 (0.0013) ±0.0028	0.8854 (0.0086) ±0.0031	0.8344 (1.0000) ±0.0059
18	0.8978 (0.0003) ±0.0025	0.8959 (0.0024) ±0.0028	0.8933 (0.0137) ±0.0029	0.8414 (1.0000) ±0.0059
19	0.9039 (0.0001) ±0.0024	0.9013 (0.0019) ±0.0027	0.9013 (0.0015) ±0.0025	0.8495 (1.0000) ±0.0056
20	0.9092 (0.0031) ±0.0026	0.9080 (0.0072) ±0.0026	0.9112 (0.0003) ±0.0023	0.8617 (1.0000) ±0.0050

Table 4.2 *This table shows the averaged classification accuracy results on the artificial dataset with $N = 10$ and the corresponding standard errors of the mean. p-values of a $t-$test for the null hypothesis that the given strategy for the given number of labeled examples m achieves accuracy less than or equal to the* Simple *strategy are stated in brackets. The asterisk indicates that the kernel billiard was run without computing an orthonormal basis (see Section 2.5).*

m	Random	Simple	RndSub(100)	KbSub(100, 1000)	KbSub(100, 5000)
2	0.4895 ±0.0199	0.4731 ±0.0214	0.4600 ±0.0213	0.4960 ±0.0211 (0.2233)	0.4974 ±0.0195 (0.2010)
3	0.7568 ±0.0113	0.6992 ±0.0184	0.6786 ±0.0196	0.6854 ±0.0198 (0.6956)	0.6620 ±0.0195 (0.9167)
4	0.8449 ±0.0074	0.8067 ±0.0158	0.7875 ±0.0151	0.8006 ±0.0145 (0.6118)	0.8017 ±0.0130 (0.5957)
5	0.8744 ±0.0066	0.8735 ±0.0107	0.8608 ±0.0106	0.8788 ±0.0086 (0.3512)	0.8807 ±0.0075 (0.2898)
6	0.8950 ±0.0054	0.9002 ±0.0088	0.9016 ±0.0071	0.9240 ±0.0046 (0.0089)	0.9174 ±0.0053 (0.0478)
7	0.9084 ±0.0047	0.9216 ±0.0062	0.9261 ±0.0044	0.9445 ±0.0031 (0.0006)	0.9398 ±0.0034 (0.0054)
8	0.9170 ±0.0040	0.9353 ±0.0038	0.9325 ±0.0040	0.9497 ±0.0027 (0.0011)	0.9486 ±0.0028 (0.0027)
9	0.9248 ±0.0036	0.9439 ±0.0030	0.9417 ±0.0026	0.9514 ±0.0024 (0.0261)	0.9569 ±0.0013 (0.0001)
10	0.9268 ±0.0035	0.9507 ±0.0022	0.9435 ±0.0025	0.9573 ±0.0014 (0.0056)	0.9591 ±0.0014 (0.0006)
11	0.9301 ±0.0032	0.9547 ±0.0019	0.9474 ±0.0024	0.9583 ±0.0013 (0.0645)	0.9584 ±0.0011 (0.0485)
12	0.9350 ±0.0026	0.9567 ±0.0017	0.9514 ±0.0020	0.9594 ±0.0012 (0.0879)	0.9607 ±0.0011 (0.0220)
13	0.9377 ±0.0025	0.9596 ±0.0016	0.9551 ±0.0018	0.9604 ±0.0010 (0.3482)	0.9615 ±0.0010 (0.1613)
14	0.9381 ±0.0027	0.9610 ±0.0015	0.9575 ±0.0017	0.9616 ±0.0011 (0.3746)	0.9627 ±0.0012 (0.1761)
15	0.9388 ±0.0026	0.9622 ±0.0015	0.9612 ±0.0016	0.9637 ±0.0011 (0.2073)	0.9639 ±0.0012 (0.1944)
16	0.9405 ±0.0026	0.9644 ±0.0012	0.9632 ±0.0013	0.9646 ±0.0011 (0.4671)	0.9650 ±0.0010 (0.3580)
17	0.9410 ±0.0025	0.9646 ±0.0012	0.9654 ±0.0012	0.9659 ±0.0010 (0.1971)	0.9652 ±0.0011 (0.3529)
18	0.9425 ±0.0023	0.9656 ±0.0011	0.9646 ±0.0013	0.9656 ±0.0011 (0.5025)	0.9660 ±0.0009 (0.3989)
19	0.9443 ±0.0021	0.9656 ±0.0010	0.9674 ±0.0011	0.9672 ±0.0012 (0.1495)	0.9675 ±0.0009 (0.0826)
20	0.9464 ±0.0019	0.9662 ±0.0011	0.9688 ±0.0010	0.9685 ±0.0010 (0.0577)	0.9686 ±0.0009 (0.0379)

Table 4.3 *This table shows the averaged classification accuracy results on the* letter *dataset and the corresponding standard errors of the mean. p-values of a $t-$test for the null hypothesis that the given strategy for the given number of labeled examples m achieves accuracy less than or equal to the* Simple *strategy are stated in brackets.*

5 Selection of Multiple Examples

Overview

In the previous chapter, we elaborated on active learning strategies which sequentially select *individual* examples from a set of unlabeled examples. However, both from a computational point of view and, more significantly, with regard to common characteristics of learning problems in practice, generalizing this scheme such that *sets* of unlabeled examples are selected yields beneficial effects. The present chapter proposes a generalized selection strategy for the active selection of batches of multiple patterns and presents experimental results indicating that this novel approach provides a more efficient method in terms of the number of labeled examples necessary to attain a level of classification accuracy in comparison to a commonly used extension of the SIMPLE selection strategy. As a fundamental component of the proposed generalization, we introduce a measure of diversity which enforces a heterogenous composition of selected batches.
The present chapter is based on (Brinker, 2003c).

5.1 Introduction

The previous chapter elaborated on active learning strategies which sequentially select individual examples from a set of unlabeled examples. The general course of action can be summarized as follows: Being initialized by only a small number of labeled examples, the learning algorithm sequentially selects individual examples from a finite set of unlabeled examples and requests the corresponding class labels. For evaluating the selection criterion of the SIMPLE strategy, it is necessary to retrain an intermediate classifier on the set of labeled examples and to evaluate functional distances of all unlabeled examples. While this setting is convenient from an analytical point of view, it does not necessarily match given problem characteristics and, furthermore, may cause computational problems in time-critical applications.

Since it is time-consuming to retrain the classifier and evaluate functional distances whenever a new example is submitted to the training set, it is more efficient from a computational point of view to select and label a set of examples before repeatedly executing the training and evaluation procedure. More significantly, if a parallel labeling component is available, the active learning setting should be modified such as to utilize this valuable resource. For instance, Warmuth et al. (2002) describe a typical setting in the domain of chemical drug discovery where multiple labels can be determined simultaneously by a biochemical experimental testing procedure.

The well-studied SIMPLE strategy for individual examples has been extended to the selection of batches of multiple patterns in a straightforward manner by choosing the top-ranked patterns with respect to the individual selection criterion.

We present a novel approach which is especially designed to select batches of multiple patterns and incorporates a measure of diversity which enforces a heterogenous composition of the set of selected patterns in order to circumvent theoretical drawbacks of the generalized SIMPLE strategy. The proposed approach has moderate computational requirements, thus, providing a feasible selection strategy for large-scale problems with several thousands of examples. Compared to previous research, the experimental results presented in Section 5.5 indicate that this approach provides a more efficient method for attaining a particular level of classification accuracy in terms of the number of labeled examples.

The remainder of this chapter is structured as follows: In the subsequent section, we recapitulate fundamental properties of the SIMPLE selection strategy and discuss its natural generalization to the selection of batches of multiple patterns. In Section 5.3, we introduce a novel approach for generalizing pool-based active learning which aims at improving on theoretical drawbacks of the straightforward extension of the SIMPLE strategy. The computational complexity of our approach is analyzed in Section 5.4 in detail. Section 5.5 presents experimental results demonstrating the benefits of our selection strategy. Finally, we summarize our results and discuss open research topics.

5.2 Generalization of the Simple Selection Strategy

In Section 4.1, the well-studied SIMPLE selection strategy was presented in detail and theoretically justified in the version space model. To provide a more self-contained presentation in this chapter, we briefly review this approach and discuss a straightforward generalization to the batch selection setting.

The SIMPLE selection strategy can be viewed as an approximate strategy for the greedy reduction of the volume of the version space, which forms the set of classifiers that are consistent with the set of labeled examples. In the following, we consider the SIMPLE strategy based on support vector machines since computational time is one of the key issues in the active learning setting examined in this chapter: Highly specialized numerical quadratic programming solvers allow for fast training and the sparsity of support vector machines reduces the computational effort of evaluating the selection criterion in comparison to the alternative choice of Bayes point machines.

In the following, we assume $\mathbf{w}^{(\mathrm{svm})}$ to be given in canonical form with respect to the training set. Under the assumption that all support vectors are normalized to a fixed length in feature space[1], $\mathbf{w}^{(\mathrm{svm})}/\|\mathbf{w}^{(\mathrm{svm})}\|_{\mathcal{F}}$ is a reasonable approximation of the center of mass of the version space $\mathcal{V}$. Furthermore, the center of mass approximates the Bayes point, which is the center of the region of intersection of

1. This assumption is satisfied if all patterns are normalized to unit length in feature space by the modified kernel $k^*(x, x') \overset{\mathrm{def}}{=} \frac{k(x,x')}{k(x,x)\,k(x',x')}$ as stated in Section 2.4.

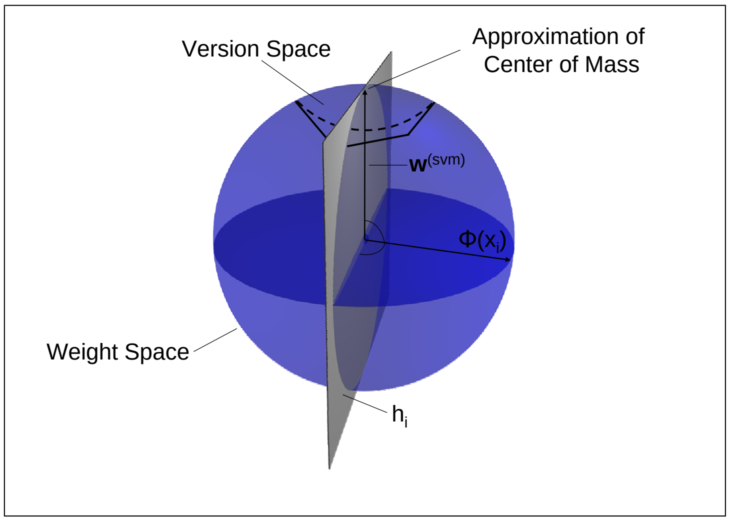


Figure 5.1 *$\phi(x_i)$ is perpendicular to $\mathbf{w}^{(svm)}$, in other words, its distance to the classification hyperplane in feature space is zero. The corresponding hyperplane h_i approximately bisects the version space into two halves of equal volume. Note that only the boundaries of the current version space are visualized, while the hyperplanes defining the version space are omitted to increase visibility.*

all hyperplanes bisecting the version space into two halves of equal volume. It was shown in Section 4.1 that if we select an unlabeled example according to minimum distance to the classification hyperplane, the corresponding constraint on the weight space approximately bisects the version space into two halves of equal volume (see Figure 5.1). Therefore, the subsequent version space based on the augmented set of labeled examples is reduced to approximately half the former volume, independently of the actual class label of the selected pattern. Formally, the SIMPLE selection strategy based on the maximum radius ball approximation is given by

$$(\mathcal{U},\mathcal{L}) \mapsto \operatorname*{argmin}_{x\in\mathcal{U}} \max_{y\in\{-1,+1\}} y\langle \mathbf{w}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}} \tag{5.1}$$

$$= \operatorname*{argmin}_{x\in\mathcal{U}} |\langle \mathbf{w}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}}|. \tag{5.2}$$

In a straightforward manner, Warmuth et al. (2002); Schohn and Cohn (2000); Campbell et al. (2000) proposed to extend the well-studied SIMPLE strategy for individual patterns to the selection of batches of multiple patterns by choosing the $H \in \mathbb{N}$ top-ranked patterns with respect to minimum absolute functional distance to the classification hyperplane given in (5.2). Formally,

$$(\mathcal{U},\mathcal{L}) \mapsto \operatorname*{argmin}_{x\in\mathcal{U}}{}^{(H)} |\langle \mathbf{w}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}}| \tag{5.3}$$

where $\text{argmin}^{(H)}$ returns the H arguments associated with minimum values and ties are broken arbitrarily.

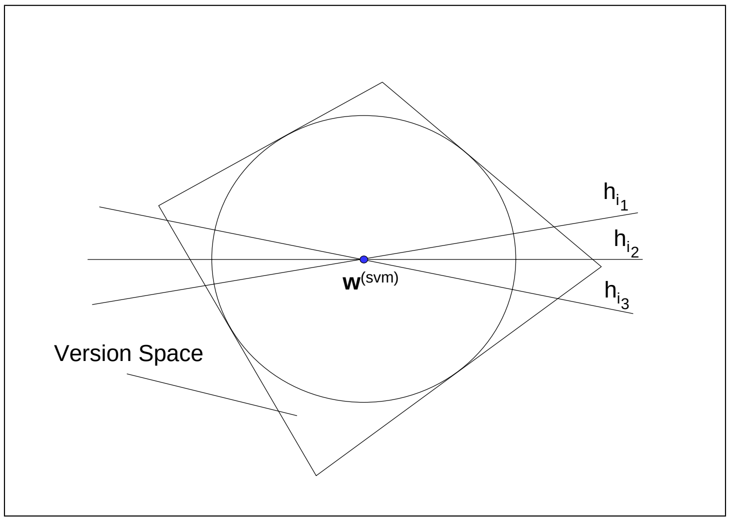


Figure 5.2 *Schematic projection of a three-dimensional version space to the plane: The dihedral angles between the hyperplanes which are induced by the three given patterns are small. Therefore, the additional reduction of the volume of the version space resulting from examples two and three can be arbitrarily small, despite the fact that their distances to the classification hyperplane evaluate to zero.*

5.3 Incorporating Diversity into Batch Selection

Drawback of Generalization of SIMPLE Strategy

Sequentially selecting unlabeled examples with respect to minimum absolute functional distance to the classification boundary can be theoretically well justified for batch size $H = 1$ in the version space model as discussed above. However, the theoretical justification for each selected example to impose a constraint on the weight space such that the version space is approximately bisected into two halves of equal volume does not hold for the generalization to simultaneous selection of multiple examples, i.e., for $H > 1$. Augmenting the labeled set by a batch of new examples that is selected exclusively based on minimum functional distances to the classification hyperplane does not necessarily impose a set of constraints on the weight space inducing a reduction of the volume of the version space substantially greater than by an individual example. As illustrated in Figure 5.2, where the dihedral angles between the restricting hyperplanes corresponding to the selected examples are small, the additional reduction of volume induced by the second and third examples can be arbitrarily small. From a more abstract point of view, a homogenous composition of the set of selected examples potentially results in a minor increase in knowledge about the learning problem. Interestingly, the binary classification problem discussed in Section 4.2 illustrates a related problem in the case of individual selection of examples where the similarity between selected examples impedes the learner from acquiring information about the first component of the generating weight vector.

A straightforward approach to circumvent this potential problem is to select batches of patterns yielding minimum worst-case version space volume. However, the associated combinatorial optimization requires an exhaustive search in the space of all possible label assignments for all batches of size H and expensive volume estimation techniques, thus, resulting in unfeasible computational complexity for practical application.

Diversity Measure

Our novel heuristic selection strategy tries to overcome this theoretical drawback by incorporating a measure of diversity that considers the dihedral angles between the restricting hyperplanes corresponding to unlabeled examples. The calculation of the (undirected) dihedral angle between two hyperplanes h_i and h_j which correspond to patterns x_i and x_j (with normal vectors $\phi(x_i)$ and $\phi(x_j)$) can be expressed in terms of the kernel function:[2]

$$|\cos(\angle(h_i, h_j))| = \frac{|\langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{F}}|}{\|\phi(x_i)\|_{\mathcal{F}} \|\phi(x_j)\|_{\mathcal{F}}} \tag{5.4}$$

$$= \frac{|k(x_i, x_j)|}{\sqrt{k(x_i, x_i) k(x_j, x_j)}}. \tag{5.5}$$

To maximize the dihedral angles within a set of hyperplanes, we employ the following incremental algorithm: Let $\mathcal{U}$ denote the set of unlabeled examples. Starting with an initial pattern $\mathcal{D} \leftarrow \{x_{i_1}\}$ which corresponds to the hyperplane h_{i_1}, we proceed to adding that unlabeled example x_{i_2} to the set $\mathcal{D} \leftarrow \mathcal{D} \cup \{x_{i_2}\}$ whose corresponding hyperplane h_{i_2} maximizes the dihedral angle to h_{i_1}. We continue to add further patterns $x_{i_j} \in \mathcal{U} \setminus \mathcal{D}$ which minimize

$$\max_{l \in \{1,\ldots,j-1\}} \frac{|k(x_{i_l}, x_{i_j})|}{\sqrt{k(x_{i_l}, x_{i_l}) k(x_{i_j}, x_{i_j})}} = \max_{l \in \{1,\ldots,j-1\}} |k^*(x_{i_l}, x_{i_j})|. \tag{5.6}$$

Figure 5.3 illustrates this strategy for the three-dimensional case by schematically projecting the version space, which is a subset of the unit hypersphere, to the plane. From a more abstract point of view, this strategy ensures a heterogenous composition of the selected batch by choosing unlabeled examples which are diverse in terms of their angles to each other in feature space.

Finally, in order to combine both requirements, viz. minimum distance to the classification hyperplane and diversity of dihedral angles, we consider the convex combination of both measures and proceed in the following way to select a batch of multiple patterns: Let $\mathcal{U}^*$ denote the set of unlabeled examples which have a distance to the classification hyperplane that is less than one. The additional distance restriction ensures that hyperplanes corresponding to (normalized) examples in $\mathcal{U}^*$ intersect with the version space. We select a batch of patterns $\mathcal{D}$ in an incremental manner as follows:

2. Note that while linear classifiers $h_{\mathbf{w}}$ take weight vectors $\mathbf{w}$ as subscripts, we denote hyperplanes corresponding to input patterns using the particular index as subscript in this chapter.

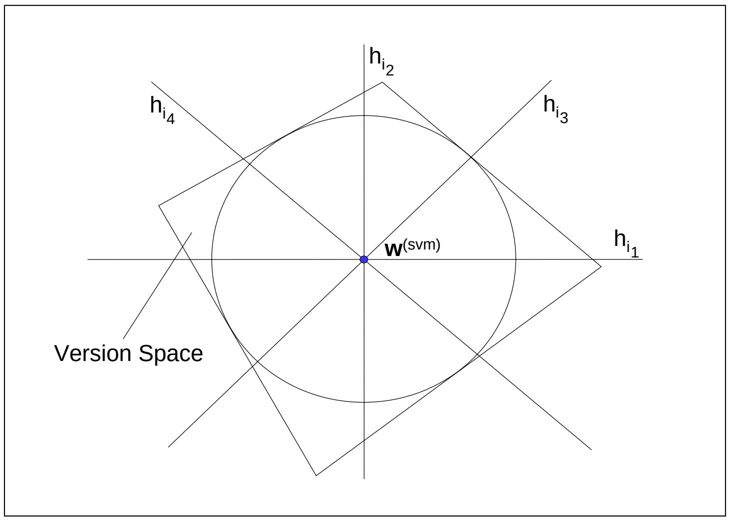


Figure 5.3 *We assume that there exist unlabeled examples $x_{i_1}, \ldots, x_{i_4}$ (corresponding to limiting hyperplanes $h_{i_1}, \ldots, h_{i_4}$) which evaluate to the global minimum of (5.6). Therefore, each selected example corresponds to a hyperplane which maximizes the minimum angle to previous hyperplanes.*

```
1: $\mathcal{D} \leftarrow \emptyset$
2: **repeat**
3:     $x \leftarrow \operatorname*{argmin}_{x' \in \mathcal{U}^* \setminus \mathcal{D}} \left( \lambda \, |f_{\mathrm{w}^{(\mathrm{svm})}}(x')| + (1-\lambda) \max_{x'' \in \mathcal{D}} |k^*(x', x'')| \right)$
4:     $\mathcal{D} \leftarrow \mathcal{D} \cup \{x\}$
5: **until** $|\mathcal{D}| = H$
```

Remember that $f_{\mathrm{w}^{(\mathrm{svm})}} : \mathcal{X} \to \mathbb{R}, x \mapsto \langle \mathbf{w}^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}}$, denotes the real-valued output of the classifier $h_{\mathrm{w}^{(\mathrm{svm})}} : \mathcal{X} \to \{-1, +1\}, x \mapsto \mathrm{sign}(f_{\mathrm{w}^{(\mathrm{svm})}}(x))$, before thresholding as defined in Definition 2.1. The trade-off parameter λ controls the individual influence of each requirement. Setting $\lambda = 1$ restores the SIMPLE strategy, whereas for $\lambda = 0$ the algorithm focuses exclusively on maximizing the dihedral angle diversity. We will refer to this selection strategy as DIVERSITY strategy.

DIVERSITY Strategy

The DIVERSITY strategy can be implemented efficiently and is almost as fast as the SIMPLE strategy (see next section for details on the computational complexity): Reevaluating the second term of the sum in line 3 in a naive manner for every single example which is submitted to the training batch $\mathcal{D}$ results in a quadratic dependence of computational time on the size of the new batch H. It is computationally more efficient to cache the values of the second term for all unlabeled examples in $\mathcal{U}^* \setminus \mathcal{D}$ and perform an update only if the absolute value of the cosine between an unlabeled example and a newly added example is greater than the stored maximum. Hence, for each example submitted to the training set, this procedure requires three kernel evaluations for all $|\mathcal{U}^* \setminus \mathcal{D}|$ unlabeled examples. The complete pseudo code of the efficient implementation of the DIVERSITY strategy is given in Algorithm 2.

Efficient Implementation

Algorithm 2 DIVERSITY *Selection Strategy*

```
input:
$\mathcal{U} = \{x_{i_1}, \ldots, x_{i_n}\}$                                  (unlabeled set)
$\mathcal{L}$                                                                  (labeled set)

$H$                                                                            (batch size)
$\lambda$                                          (trade-off between distance and diversity)
--------------------------------------------------------------------------------
output:
$\mathcal{D}$                                                  (selected batch of patterns)
--------------------------------------------------------------------------------

$\mathcal{D} \leftarrow \emptyset$
$\mathcal{I} \leftarrow \{i_1, \ldots, i_n\}$

train support vector machine $h_{\mathrm{w}^{(svm)}}(\cdot) = \mathrm{sign}\left(f_{\mathrm{w}^{(svm)}}(\cdot)\right)$ on $\mathcal{L}$

for all $j \in \mathcal{I}$ do
    $distance[j] \leftarrow |f_{\mathrm{w}^{(svm)}}(x_j)|$
    $maxCos[j] \leftarrow 0$
end for

repeat
    $minIndex \leftarrow 1$
    $minValue \leftarrow +\infty$
    for all $j \in \mathcal{I}$ do
        $measure \leftarrow \lambda\, distance[j] + (1 - \lambda)\, maxCos[j]$
        if ($measure < minValue$) and ($distance[j] < 1$) then
            $minIndex \leftarrow j$
            $minValue \leftarrow measure$
        end if
    end for

    $\mathcal{D} \leftarrow \mathcal{D} \cup \{x_{minIndex}\}$
    $\mathcal{I} \leftarrow \mathcal{I} \backslash \{minIndex\}$

    for all $j \in \mathcal{I}$ do
        $nextCos \leftarrow |k^*(x_j, x_{minIndex})|$
        if $nextCos > maxCos[j]$ then
            $maxCos[j] \leftarrow nextCos$
        end if
    end for
until $|\mathcal{D}| = H$

return $\mathcal{D}$
```

H	1	2	4	8	16	32	64
DIVERSITY	1691.9	935.8	439.8	204.2	105.9	66.7	50.7
SIMPLE	1691.9	821.0	359.1	159.0	72.8	29.6	15.5

Table 5.1 *Experimentally measured running time (in seconds) for the* DIVERSITY *and the* SIMPLE *selection strategy on the shuttle dataset for different choices of the batch size H. The final number of labeled examples was* 456 *from a pool of size* 21750. *With the batch size increasing, the difference in computational time levels off at the constant computational time necessary to perform the dihedral angle calculations in compliance with the theoretical analysis. For small H, the results are slightly distorted by procedural overhead.*

5.4 Computational Complexity

The linearly constraint quadratic programming problems associated with support vector machines are solvable in polynomial time in terms of the number of training examples. More precisely, the theoretical order of complexity was shown to be in $\mathcal{O}(m^3)$ where m denotes the number of examples (Burges, 1998). In practice, highly specialized numerical quadratic programming solvers require empirically estimated computational time of $\mathcal{O}(m^2)$ for training support vector machines (Joachims, 1999a; Platt, 1999a). We consider the latter order of complexity to provide a more realistic analysis.

The process of incrementally learning m examples by adding batches consisting of H examples (with $1 \leq H < m$ and $H|m$ for notational convenience) to the training set before retraining with a standard batch learning algorithm requires an accumulated computational time of order

$$\mathcal{O}\left(\sum_{i=1}^{\frac{m}{H}}(Hi)^2\right) = \mathcal{O}\left(H^2 \frac{\frac{m}{H}\left(\frac{m}{H}+1\right)\left(\frac{2m}{H}+1\right)}{6}\right) = \mathcal{O}\left(\frac{m^3}{H}\right), \tag{5.7}$$

excluding the selection steps. It is possible to increase computational efficiency by using an incremental support vector machine algorithm (Cauwenberghs and Poggio, 2001). However, this incremental algorithm does not provide an upper bound on the number of operations per iteration and therefore does not allow us to derive a lower order of complexity from a theoretical point of view. Furthermore, the time complexity of the pure training step does not vary for different selection strategies and thus is of minor importance when comparing the SIMPLE and the DIVERSITY selection strategy.

In order to select new patterns, we need to calculate the functional distances of all unlabeled examples to the classification hyperplane once within each iteration, i.e., after having selected H examples. Denoting the initial number of unlabeled examples by $n_0 \overset{\text{def}}{=} |\mathcal{U}_0|$, we obtain an overall time complexity for distance calculations of order

$$\mathcal{O}\left(\sum_{i=1}^{\frac{m}{H}-1} Hi(n_0 - Hi)\right) \subset \mathcal{O}\left(\frac{n_0 m^2}{H}\right) \tag{5.8}$$

taking into account that the computational time for one distance calculation is proportional to the number of support vectors, which is upper bounded by the number of labeled examples. In addition to this, for every example which is submitted to the labeled set, three kernel evaluations for all unlabeled examples are necessary to update the dihedral angle values. This amounts to $\mathcal{O}(n_0\, m)$ time complexity in total.

Summing up training time and selection time, actively learning m examples from a pool of n_0 examples and batch size H with the DIVERSITY strategy requires computational time of order

$$\underbrace{\mathcal{O}\left(\frac{m^3}{H}\right)}_{\text{training}} + \underbrace{\mathcal{O}\left(\frac{n_0 m^2}{H}\right)}_{\substack{\text{distance}\\\text{calculations}}} + \underbrace{\mathcal{O}\Big(n_0\, m\Big)}_{\substack{\text{dihedral angle}\\\text{calculations}}}. \tag{5.9}$$

In comparison to the SIMPLE strategy, the DIVERSITY strategy requires additional computational time of order $\mathcal{O}(n_0 m)$ to perform the dihedral angle calculations.

5.5 Experiments

5.5.1 Experimental Setting

We have conducted several experiments on datasets that are publicly available from the UCI repository of machine learning databases (Blake and Merz, 1998) and from the Statlog collection (Michie et al., 1994) to evaluate the DIVERSITY selection strategy. Our experimental setting included the SIMPLE and the DIVERSITY strategy for various choices of values for the control parameters H and λ. Additionally, the RANDOM selection strategy of new training batches of multiple examples served as a baseline strategy.

Each of the datasets was randomly split 100 times into a training set and a test set of equal size. The initial training batches consisted of 8 examples which were randomly drawn from the training sets and contained at least one positive and one negative example. While new training examples were selected from the remaining training sets according to the different strategies (associated class labels were masked out prior to selection), the classification accuracy was evaluated on the test sets after every selection step and the results were averaged over all 100 experimental runs. In our experiments, we employed a modified version of BSVM[3] (Hsu and Lin, 2002b) which trains support vector machines without bias in compliance with our theoretical line of reasoning. All experiments were conducted on a single processor pentium 4 with

3. We have to thank Chih-Len Lin for valuable comments on the implementation.

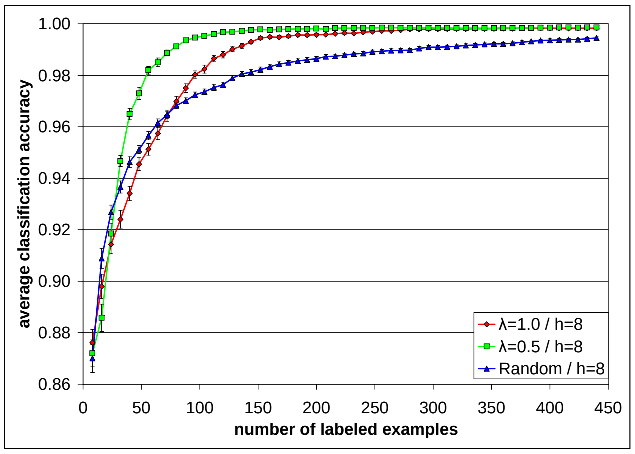


Figure 5.4 *Learning curves for the* Simple*,* Diversity *and* Random *selection strategy on the shuttle dataset. The classification accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration $H = 8$ new examples were submitted to the training set. The* Diversity *strategy outperforms both the* Simple *and the* Random *selection strategy.*

1.8 GHz and 1 GB of memory.

We chose the *shuttle* dataset from the Statlog collection as the first benchmark problem. It contains 43500 examples[4] with 9 continuous features. Approximately 80% of the examples belong to class 1 of the 7 classes. We tested all strategies on the binary classification problem class 1-versus-all and used an RBF kernel with $\sigma = 2$. The second benchmark dataset was the *waveform-5000* problem from the UCI repository, which contains 5000 examples with 21 continuous features and 3 target classes. The class distribution is $\frac{1}{3}$ for each of the 3 classes. We conducted our experiments on the class 0-versus-all problem using an inhomogeneous polynomial kernel of degree 5. The *krkpa7* dataset from the UCI repository contains 3196 examples with 36 discrete features taking either two or three different values. The number of target classes is 2 where approximately 52% of the examples belong to class 1 and 48% belong to class 2. On the *krkpa7* problem we employed an inhomogeneous polynomial kernel of degree 3.

The first part of our experiments has been set up to investigate the influence of the batch size on the efficiency of the selection strategies. For all datasets, we fixed $\lambda = 0.5, 1.0$ and tested the Diversity strategy for batch sizes $H = 8, 16, 32, 64$. For $\lambda = 1.0$, the Diversity strategy corresponds to the Simple strategy. In a second series of experiments, we examined the influence of the diversity control parameter λ. Therefore, setting the batch size to $H = 16$, we compared the Diversity strategy for $\lambda = 0, 0.25, 0.50, 0.75, 1.0$ to the Random selection strategy.

4. The original dataset contains 58000 examples that are usually split into 43500 training and 14500 test examples. For computational reasons, we only used the first 43500 examples.

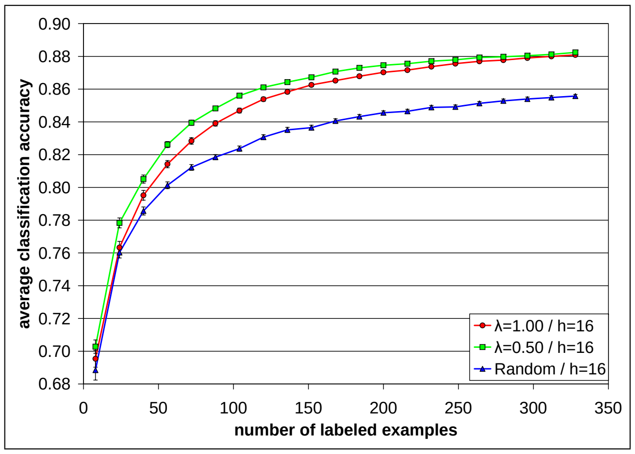


Figure 5.5 *Learning curves for the* Simple, Diversity *and* Random *selection strategy on the waveform-5000 dataset. The classification accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration* $H = 16$ *new examples were submitted to the training set. The* Diversity *strategy outperforms both the* Simple *and the* Random *selection strategy.*

5.5.2 Experimental Results

For $\lambda = 0.5$, the estimated classification accuracy of the Diversity strategy is consistently superior to the Simple strategy ($\lambda = 1.00$), independently of the batch size (see Figure 5.4 for a typical learning curve for batch size $H = 8$ on the *shuttle* dataset and Figure 5.5 for batch size $H = 16$ on the *waveform-5000* dataset). Furthermore, the efficiency of both the Diversity and the Simple strategy in terms of the attained level of accuracy decreases if the batch size H increases (Figure 5.6 and Figure 5.7 show learning curves for the *krkpa7* dataset and the *waveform-5000* dataset for different batch sizes). With the number of training examples increasing, we observed a higher level of classification accuracy for the Diversity and the Simple strategy in comparison to the Random strategy in all our experiments. However, at the beginning of the learning process the Random strategy tends to perform better with the turning point increasing with the batch size H. For batch sizes $H = 8$ and $H = 16$, the turning point is typically reached after less than 50 training examples, whereas for $H = 64$ it can take several hundred examples. Therefore, our experiments suggest that with respect to classification accuracy it is preferable to choose H as small as possible, while from a computational point of view increasing H allows us to control the computational effort. Hence, H has to be chosen carefully as a trade-off between accuracy and computational complexity.

In our experiments, the learning curves for $\lambda = 0.25, 0.50, 0.75$ were very similar (see Figure 5.8 for the *shuttle* dataset and Figure 5.9 for the *waveform-5000* dataset) and consistently superior to the Simple strategy ($\lambda = 1.00$). Contrary to this, the behavior of the Diversity strategy for $\lambda = 0.00$ is rather unstable, ranging from the best (for the *shuttle* dataset) to the worst strategy (for the *waveform-5000* dataset) apart from the RANDOM strategy. Thus, except for the extreme choice of $\lambda = 0.00$, our experiments indicate that the DIVERSITY strategy is fairly robust with respect to the choice λ.

5.6 Discussion

Our experiments indicate that the DIVERSITY batch selection strategy is an efficient method in terms of the number of labeled examples necessary to attain a certain level of classification accuracy and consistently outperforms a common generalization of the SIMPLE strategy. Requiring only a limited amount of additional computational time, the proposed selection strategy is applicable to large-scale learning problems. Although the DIVERSITY selection strategy was fairly robust with respect to the trade-off parameter λ in our experiments, the question of how to determine optimal values for λ, which may depend on the progress of the training process (dynamic adaptation), will be investigated in future research. As a promising first step in this direction, Goh et al. (2004) considered a fixed time-varying scheme with decaying λ for our DIVERSITY selection strategy and demonstrated improvements over the basic strategy in the domain of image retrieval.

Interestingly, Baram et al. (2004) proposed a multi-armed bandit meta-algorithm for switching online among an ensemble of different active learning strategies and suggested to include a so-called kernel farthest-first strategy into the committee. This selection strategy is similar to the DIVERSITY strategy with $\lambda = 0$ in the case of normalized patterns as the underlying selection criterion is based on dihedral angle maximization with respect to the set of labeled examples. In compliance with our experimental results, the kernel farthest-first strategy was shown to be rather unstable as an individual selection strategy, however, contributed to the overall performance of the committee. The essential difference to the DIVERSITY strategy is the fact that we combine the SIMPLE strategy and the dihedral angle diversity strategy using the convex combination of the individual selection criteria, while Baram et al. (2004) consider a sophisticated switching technique for integrating the kernel farthest-first strategy.

In the field of computational linguistics, the proposed notion of diversity was integrated into a multi-criteria active learning approach for the reduction of the annotation effort in named entity recognition (Shen et al., 2004). As fundamental components, this approach combines measures of informativeness, representativeness and diversity.

Beyond active learning with support vector machines, the proposed batch selection strategy can be adapted to other base strategies which aim at selecting examples halving the volume of the version space, such as query-by-committee (Seung et al., 1992). In particular, it is possible to apply our DIVERSITY selection strategy without modifications to the SIMPLE strategy based on Bayes point machines, which are able to more accurately approximate the center of mass if the version space is not symmetrical (Herbrich et al., 2001).

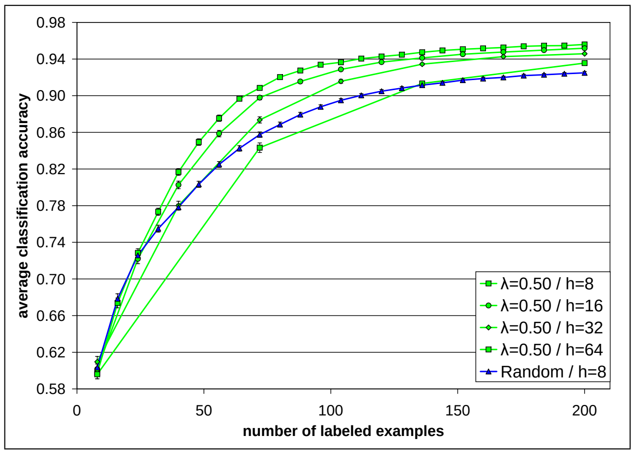


Figure 5.6 *Learning curves for the* DIVERSITY *selection strategy* ($\lambda = 0.50$) *and for the* RANDOM *strategy on the krkpa7 dataset. The classification accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration* $H = 8, 16, 32, 64$ *new examples were submitted to the training set. For smaller batch sizes* H*, the* DIVERSITY *strategy is more efficient in terms of the number of labeled examples necessary to attain a certain level of accuracy.*

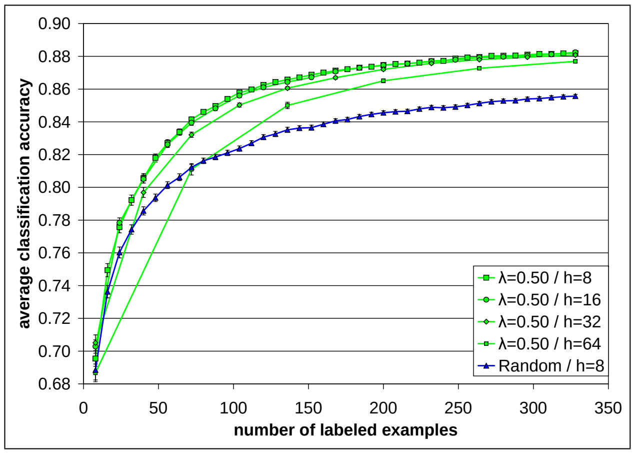


Figure 5.7 *Learning curves for the* DIVERSITY *selection strategy* ($\lambda = 0.50$) *and for the* RANDOM *strategy on the waveform-5000 dataset. The classification accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration* $H = 8, 16, 32, 64$ *new examples were submitted to the training set. For smaller batch sizes* H*, the* DIVERSITY *strategy is more efficient in terms of the number of labeled examples necessary to attain a certain level of accuracy.*

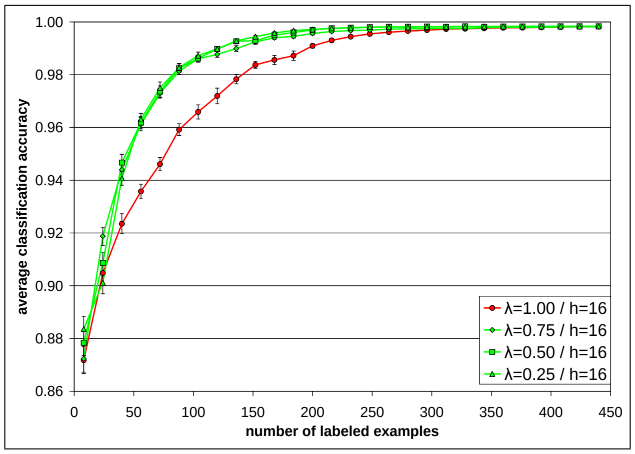


Figure 5.8 *Learning curves for the* SIMPLE *and* DIVERSITY *selection strategy for* $\lambda = 0.25, 0.50, 0.75$ *on the shuttle dataset. The classification accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration* $H = 16$ *new examples were submitted to the training set. For all choices of* λ, *the* DIVERSITY *strategy outperforms the* SIMPLE *selection strategy.*

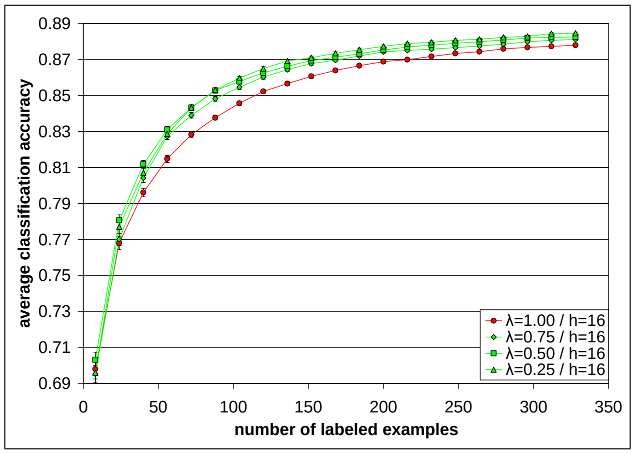


Figure 5.9 *Learning curves for the* SIMPLE *and* DIVERSITY *selection strategy for* $\lambda = 0.25, 0.50, 0.75$ *on the waveform-5000 dataset. The classification accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration* $H = 16$ *new examples were submitted to the training set. For all choices of* λ, *the* DIVERSITY *strategy outperforms the* SIMPLE *selection strategy.*

6 Multiclass Classification

Overview

In supervised classification learning, the active learning framework has been considered to reduce the labeling effort. While most research on active learning in the field of kernel machines has focused on binary problems, less attention has been paid to the problem of learning classifiers in the case of multiple classes. In the present chapter, we consider three common decomposition methods for expressing multiclass problems in terms of sets of binary classification problems and propose novel active learning selection strategies in order to reduce of the labeling effort. Various experiments conducted on real-world datasets demonstrate the merits of our approach in comparison to previous research.
The present chapter extends the results presented in (Brinker, 2004b).

6.1 Introduction

Multiclass Setting

As a natural generalization of binary classification learning, we consider the problem of learning a classifier in the case of an arbitrary number of target class labels in a supervised learning scenario. More formally, the fundamental objective is to learn a mapping h from a given input space $\mathcal{X}$ to a finite and a priori fixed set of class labels $\mathcal{Y} = \{1, \dots, d\}$ based on a finite training set of labeled examples $\{(x_1, y_1), \dots, (x_m, y_m)\} \subset \mathcal{X} \times \mathcal{Y}$. Various real-world learning problems are compatible with the multiclass learning setting, e.g., optical character recognition (OCR), which considers the objective of predicting digits and letters depicted on digitalized images. Multiclass classification learning assumes the target classes to be disjunct, i.e., each pattern belongs exclusively to a specific class. In contrast to the multiclass setting, in multilabel learning each pattern is not assigned to a particular class label in general, but possibly belongs to several different classes, i.e., the target space is given by the power set of $\{1, \dots, d\}$.[1]

From Binary to Multiclass Learning

The original formulation of support vector machines (Boser et al., 1992) is inherently restricted to binary classification learning, a property which holds for many other machine learning algorithms as well. We can differentiate two principal categories of approaches which generalize binary support vector machines to the multiclass setting.

1. Har-Peled et al. (2002) give precise definitions for various categories of learning problems and propose a unified approach to the reduction of more complex learning problems to binary classification learning.

One principal means for the generalization of binary support vector machines is to construct and combine several binary classifiers in a suitable manner. The second category of approaches considers a direct reformulation of the original optimization problem into one more general optimization problem. While the latter category is arguably the more natural and elegant one, the underlying optimization problems contain more variables, which, in consequence, leads to an increased overall computational complexity in comparison to the first category (Crammer and Singer, 2001; Weston and Watkins, 1999). Moreover, there is no empirical evidence that all-in-one approaches, such as (Crammer and Singer, 2001; Weston and Watkins, 1999), compare favorably to binary-based approaches in terms of classification accuracy in general.[2] Therefore, binary-based approaches are more suitable for practical use and widely employed to solve real-world problems. We will consider different binary decomposition approaches to generalize pool-based active learning to the multiclass setting in the following.

Overview: Binary-based Approaches

The one-versus-one approach (Knerr et al., 1990) decomposes multiclass problems into sets of binary classification problems by considering all pairwise decisions between two class labels. Each pairwise decision problem is treated independently as a binary classification problem. For predicting class labels, *all* binary classifiers are evaluated and the results are summarized by means of a combination procedure. In DAG multiclass classification (Platt, 1999b), multiclass problems are decomposed in an analogous manner. However, in contrast to one-versus-one classification, binary classifiers are considered as nodes in a decision directed acyclic graph (DDAG). Class labels are predicted by evaluating the corresponding decision paths in the DDAG and assigning the class labels associated with the leaf nodes. An alternative approach to the expression of multiclass problem is the one-versus-all approach (Cortes and Vapnik, 1995), which trains a separate classifier for each possible class against the rest of classes and predicts class labels according to the maximum output[3] among all d binary classifiers.

In the field of kernel machines, active learning has been successfully applied to classification problems to reduce the labeling effort: Areas of application range from character recognition (Campbell et al., 2000) to text classification (Tong, 2001; Schohn and Cohn, 2000) and drug discovery (Warmuth et al., 2002). While the approaches presented in (Campbell et al., 2000; Schohn and Cohn, 2000; Warmuth et al., 2002) are restricted to binary classification, Tong (2001) additionally considered active learning of multiclass classifiers using the one-versus-all approach.

We propose a novel extension of pool-based active learning to multiclass problems. Employing one-versus-one decomposition, DAG multiclass classification and the one-versus-all technique, we propose heuristic strategies for the selection of new training examples. These strategies are compared to (Tong, 2001) in the case of the one-versus-all decomposition and a straightforward generalization of this approach to both pairwise decomposition techniques. Additionally, we consider random selection of new

2. Hsu and Lin (2002a) presented an extensive empirical comparison of different approaches to the generalization of binary support vector machines.

3. In the following, we consider real-valued classifiers which are thresholded at zero to make binary predictions $\{-1, +1\}$.

examples as a baseline strategy. Various experiments conducted on real-world datasets demonstrate that the proposed selection strategies are more efficient in the majority of cases.

The present chapter is organized as follows: The subsequent section discusses the aforementioned techniques for solving multiclass problems. In Section 6.3, we investigate active learning in the case of multiclass classification and introduce our novel generalization. Section 6.4 presents experimental results conducted on several real-world datasets and demonstrates the benefits of our approach. In Section 6.5, we provide a summary of our results and point out references to related research.

6.2 Reducing Multiclass to Binary Classification Learning

This section recapitulates different techniques for solving multiclass classification problems by means of constructing and combining sets of binary problems. Formally, the learning problem that we are investigating can be stated as follows: Based on a given training set of labeled examples

$$\mathcal{L} = \{(x_1, y_1), \dots, (x_m, y_m)\} \subset \mathcal{X} \times \mathcal{Y} \tag{6.1}$$

with $\mathcal{X}$ denoting a nonempty input space and $\mathcal{Y} = \{1, \dots, d\}$ being a finite space of class labels, we seek to induce a mapping $h : \mathcal{X} \to \mathcal{Y}$ for the accurate prediction of class labels for unseen examples. The most common measure for evaluating classifiers in multiclass learning is classification accuracy, which is a straightforward generalization of the binary definition. In Definition 1.3, we have already defined the 0-1-loss in a manner such that it generalizes to multiclass classification. The corresponding risk evaluates to the misclassification probability.

While the following decomposition techniques are applicable to a wider range of binary learning algorithms, we focus on the hypothesis class of linear classifiers in kernel feature space (see Section 2.2),

$$\begin{aligned} h_{\mathrm{w}} : \mathcal{X} &\to \{-1, +1\} && (6.2)\\ x &\mapsto \operatorname{sign}(\langle \mathbf{w}, \phi(x) \rangle_{\mathcal{F}}) && (6.3) \end{aligned}$$

with weight vector $\mathbf{w} \in \mathcal{F}$ and the sign-function thresholding the real-valued output of h at zero. Support vector machines form a category of linear classifiers which admit a dual representation as an expansion in terms of the training examples (Vapnik, 1998):

$$\begin{aligned} h_{\mathrm{w}^{(\mathrm{svm})}}(x) &= \operatorname{sign}(\langle \mathbf{w}^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}}) && (6.4)\\ &= \operatorname{sign}\left(\left\langle \sum_{i=1}^{m} \alpha_i \phi(x_i), \phi(x) \right\rangle_{\mathcal{F}} \right) && (6.5)\\ &= \operatorname{sign}\left(\sum_{i=1}^{m} \alpha_i \, k(x_i, x) \right) && (6.6) \end{aligned}$$

where $\alpha = (\alpha_1, \dots, \alpha_m)^\top \in \mathbb{R}^m$ denotes the vector of Lagrange multipliers. If the training set is linearly separable[4] in feature space, hard margin support vector machines calculate the vector of Lagrange multipliers α such that the minimum margin $\min_{i=1,\ldots,m} y_i \langle \mathbf{w}^{(\text{svm})}, \phi(x_i) \rangle_{\mathcal{F}}$ between training examples and the classification boundary in feature space is maximized. Note that scaling with a positive constant does not have an effect on the classification outcome. Therefore, we have to impose a normalization constraint on $\mathbf{w}^{(\text{svm})}$, e.g., unit length in feature space, in order to get a unique solution.

6.2.1 One-versus-all Decomposition

In the field of kernel machines, a common method for solving multiclass problems is to train a separate binary classifier

$$h_{\mathrm{w}_i} : \mathcal{X} \to \{-1, +1\} \tag{6.7}$$
$$x \mapsto \operatorname{sign}(f_{\mathrm{w}_i}(x)) = \operatorname{sign}(\langle \mathbf{w}_i, \phi(x) \rangle_{\mathcal{F}}) \tag{6.8}$$

for each of the d target classes against the remaining set of classes (Cortes and Vapnik, 1995). More precisely, all examples belonging to class i are considered as positive examples in the process of training the binary classifier h_{w_i}, whereas the remaining examples are considered as negative examples. Class labels for unseen examples are predicted according to the maximum real-valued output of the binary classifiers before thresholding:

$$h : \mathcal{X} \to \{1, \ldots, d\} \tag{6.9}$$
$$x \mapsto \operatorname*{argmax}_{i=1,\ldots,d} f_{\mathrm{w}_i}(x) = \operatorname*{argmax}_{i=1,\ldots,d} \langle \mathbf{w}_i, \phi(x) \rangle_{\mathcal{F}}. \tag{6.10}$$

If we assume that the computational time for training a binary classifier h_{w_i} scales as $\mathcal{O}(m^2)$ with respect to the number of labeled training examples, which is a reasonable assumption in support vector learning as stated in Section 2.4, then the overall training complexity for the one-versus-all decomposition techniques evaluates to $\mathcal{O}(dm^2)$.

6.2.2 One-versus-one Decomposition

One-versus-one multiclass classification (Knerr et al., 1990) learns a separate binary classifier for each of the $d(d-1)/2$ pairs of different class labels. Each binary classifier $h_{\mathrm{w}_{ij}}$ (with $1 \leq i < j \leq d$) determines for a given pattern x whether or not class label i is preferred over class label j. The training set for $h_{\mathrm{w}_{ij}}$ consists of all the examples (x, y) having class labels $y \in \{i, j\}$ with class i being considered as the positive class and class j being considered as the negative class, while the remaining examples with $y \notin \{i, j\}$ are disregarded.

Knerr et al. (1990) suggested to combine the binary classifications $h_{\mathrm{w}_{ij}}(x)$ using an AND-gate, while Friedman (1996) proposed to interpret binary classifications as

4. See Section 2.4 for details on how to incorporate the nonseparable case.

votes for class i and j, respectively, and to determine the class label which receives the maximum number of aggregated votes for the prediction of multiclass labels. Moreover, Friedman (1996) proved that under certain assumptions the MAX wins voting strategy is Bayes-optimal.

The MAX wins strategy is the most common technique in one-versus-one multiclass learning with kernel machines. We will consider this method in the following to combine the votes of the underlying binary classifiers. In related research, this technique is also referred to as pairwise or round robin classification (see (Fürnkranz, 2002)). Formally, the MAX wins prediction strategy in one-versus-one multiclass learning is given by

$$h : \mathcal{X} \to \{1, \dots, d\} \tag{6.11}$$

$$x \mapsto \operatorname*{argmax}_{i=1,\dots,d} \Big(\sum_{j=i+1}^{d} \max(h_{ij}(x), 0) + \sum_{j=1}^{i-1} \max(-h_{ji}(x), 0) \Big). \tag{6.12}$$

In the case of ties, though it might be suboptimal, we prioritize lower class labels as suggested in a recent study on multiclass classification learning with support vector machines (Hsu and Lin, 2002a).

Pairwise one-versus-one classification provides a framework that is applicable in a more general setting to a wide range of underlying binary classifiers. As noted by Fürnkranz (2002), the above-defined decomposition scheme implicitly assumes that that the base classifiers are *class-symmetric*, i.e., the problem of discriminating class i against class j is identical to discriminating class j against class i. It is readily verified that this assumption holds for both support vector machines and Bayes point machines. Furthermore, it was shown in (Fürnkranz, 2002) that the one-versus-one technique is more efficient than the one-versus-all technique in terms of training time if the binary base learner has superlinear time complexity.

6.2.3 DAG Multiclass Classification

In an analogous manner to one-versus-one pairwise decomposition, the directed acyclic graph (DAG) multiclass approach (Platt, 1999b) trains a separate classifier for each pair of different classes. While one-versus-one classification predicts class labels by means of the MAX wins voting technique, which requires all $d(d-1)/2$ classifiers to be evaluated, DAG multiclass classification considers each binary classifier as a decision node in a DAG learning architecture: Each classifier is assigned to one of the $d(d-1)/2$ internal nodes of a binary rooted DAG with d leaves where the ith internal node in layer j is connected to the ith and $(i+1)$th node in layer $j+1$ and each leaf is assigned to a particular class label. To illustrate the DAG architecture, an exemplary decision DAG corresponding to a four-class problem is depicted in Figure 6.1.

Given a new pattern x, the classifier assigned to the root node is evaluated and, depending on the classification outcome, the node is exited via the left or right edge to the next layer. Then, the subsequent node is evaluated analogously. Finally, x is assigned to the class label associated with the leaf node of the decision path. Since all decision paths have depth $d-1$, we have to evaluate $d-1$ classifiers in order to predict a class label.

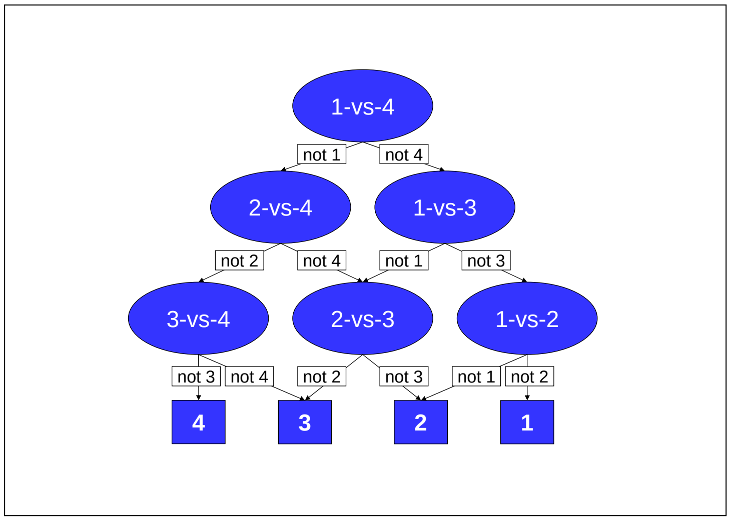


Figure 6.1 *This figure illustrates the DAG multiclass to binary decomposition architecture on a four-class problem (adapted from (Platt et al., 2000)).*

6.3 Active Multiclass Learning

This section introduces novel heuristic active learning selection strategies for the one-versus-one, DAG and the one-versus-all decomposition techniques. We derive selection criteria for multiclass classification based on the version space model for binary classification, which was discussed in Section 2.3 in detail.

All the aforementioned decomposition techniques, i.e., the one-versus-all, one-versus-one and the DAG approach, depend on sets of binary classification problems to solve multiclass problems. The key idea for generalizing binary pool-based active learning is to consider well-studied binary selection strategies on each of the corresponding binary problems and to combine all binary selection scores into a single multiclass selection criterion.

With a view to the version space model, we are given a *set* of search problems. Based on this model, binary active learning is generalized in (Tong, 2001) considering the one-versus-all decomposition method: Under the assumption that the base support vector machines are normalized such that the closest labeled example has unit functional distance to the classification hyperplane, i.e., $\min_{i=1,\dots,m} |\langle \mathbf{w}^{(\mathrm{svm})}, \phi(x_i)\rangle_{\mathcal{F}}| = 1$ (canonical form), the ratio of volume reduction on a *single* version space when augmenting the binary training set by an example (x, y) is approximated by $\frac{1+y\langle \mathrm{w}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}}}{2}$. According to this simple approximation, the volume of the version space is reduced to approximately half the former size if the margin of a binary example evaluates to zero, whereas if the margin is 1, there is no reduction at all, and if the margin evaluates to -1, the volume is reduced to zero. Furthermore, to quantify the volume reduction of the set of d version spaces, Tong (2001) proposed to consider the product of all individual reduction ratios:

$$\prod_{i=1}^{d} \frac{1 + y^{(i)} \langle \mathbf{w}_i^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}}}{2} \tag{6.13}$$

where $\mathbf{w}_i^{(\text{svm})}$ denotes the weight vector of the support vector machine and $y^{(i)} \in \{-1, +1\}$ denotes the binary label of the multiclass example $(x, y) \in \mathcal{X} \times \{1, \dots, d\}$ in the binary classification problem i-versus-all ($i \in \{1, \dots, d\}$). Considering a best worst-case scenario, for each unlabeled example x the maximum product of reduction ratios over all possible multiclass class labels is calculated. As stated in Section 6.2.1, assuming that i is the correct class label, x is submitted to the binary problem i-versus-all as a positive and to all binary problems j-versus-all with $j \neq i$ as a negative example. Hence, unlabeled examples are selected according to the following best worst-case selection criterion:

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x \in \mathcal{U}} \max_{y \in \{1,\dots,d\}} \left(\prod_{i=1}^{d} \frac{1 + y^{(i)} \langle \mathbf{w}_i^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}}}{2} \right) \tag{6.14}$$

$$= \operatorname*{argmin}_{x \in \mathcal{U}} \max_{i=1,\dots,d} \left(\frac{1 + \langle \mathbf{w}_i^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}}}{2} \prod_{\substack{j=1,\dots,d \\ j \neq i}} \frac{1 - \langle \mathbf{w}_j^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}}}{2} \right) \tag{6.15}$$

$$= \operatorname*{argmin}_{x \in \mathcal{U}} \max_{i=1,\dots,d} \left(\left(1 + \langle \mathbf{w}_i^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}}\right) \prod_{\substack{j=1,\dots,d \\ j \neq i}} 1 - \langle \mathbf{w}_j^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}} \right). \tag{6.16}$$

GLOBAL Strategy

We refer to this multiclass selection strategy as GLOBAL strategy. In the binary case $d = 2$, the GLOBAL strategy is equivalent to the SIMPLE selection strategy:

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x \in \mathcal{U}} \max_{i=1,2} \left(\left(1 + \langle \mathbf{w}_i^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}}\right) \prod_{\substack{j=1,2 \\ j \neq i}} 1 - \langle \mathbf{w}_j^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}} \right) \tag{6.17}$$

$$= \operatorname*{argmin}_{x \in \mathcal{U}} \max \left\{ \left(1 + \langle \mathbf{w}_1^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}}\right)^2, \left(1 - \langle \mathbf{w}_1^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}}\right)^2 \right\} \tag{6.18}$$

$$= \operatorname*{argmin}_{x \in \mathcal{U}} \left| \langle \mathbf{w}_1^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}} \right| \tag{6.19}$$

Note that equality holds in (6.18) as a direct consequence of the symmetry of the binary classifiers, i.e., $\mathbf{w}_1^{(\text{svm})} = -\mathbf{w}_2^{(\text{svm})}$. In (6.19), the max-term is substituted by an order-preserving transformation.

The GLOBAL selection strategy can be generalized to pairwise decomposition, which subsumes the one-versus-one and the DAG techniques, in a straightforward manner. In contrast to one-versus-all decomposition where we are given a set of d version spaces, we have to consider $d(d-1)/2$ binary problems as stated in Section 6.2.2: Each labeled multiclass example (x, y) is submitted to the $d - 1$ binary classification problems i-versus-j (with $i < j$ and $y \in \{i, j\}$), where (x, y) is relabeled as a positive example $(x, +1)$ if $y = i$ and as a negative example $(x, -1)$ otherwise. Since the remaining problems are unaffected and hence the ratio of volume reduction evaluates to 1, the adapted Global selection strategy is given by

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x\in\mathcal{U}} \max_{i=1,\dots,d} \left(\prod_{j=i+1}^{d} \frac{1+\langle \mathbf{w}_{ij}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}}}{2} \prod_{j=1}^{i-1} \frac{1-\langle \mathbf{w}_{ji}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}}}{2} \right) \tag{6.20}$$

$$= \operatorname*{argmin}_{x\in\mathcal{U}} \max_{i=1,\dots,d} \left(\prod_{j=i+1}^{d} 1+\langle \mathbf{w}_{ij}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}} \prod_{j=1}^{i-1} 1-\langle \mathbf{w}_{ji}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}} \right). \tag{6.21}$$

In addition to the generalized Global strategy, we propose a novel approach which considers a best worst-case scenario with respect to binary reduction ratios. More precisely, we aim at selecting unlabeled examples which yield maximum worst-case volume reduction for any of the given binary version spaces. In an analogous manner as for the Global strategy, the volume reduction for each of the binary problems is estimated by the normalized margin where lower values correspond to higher ratios of volume reduction. Given an unlabeled example, we consider all possible class labels and calculate the minimum margins within the corresponding set of binary examples. Each example is assigned to the maximum among all minimum margins, i.e., the maximum estimated volume reduction on any of the binary problems for the worst-case class label is quantified by this measure. Finally, we request the class label of that unlabeled example with minimum score, i.e., with maximum estimation of guaranteed volume reduction on a single binary problem. More formally, this strategy can be stated in the case of one-versus-all decomposition as follows:

BinMin Strategy

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x\in\mathcal{U}} \max_{i=1,\dots,d} \left(\min \left\{ \frac{1+\langle \mathbf{w}_{i}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}}}{2} \right\} \cup \bigcup_{\substack{j=1,\dots,d\\ j\neq i}} \left\{ \frac{1-\langle \mathbf{w}_{j}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}}}{2} \right\} \right)$$
$$= \operatorname*{argmin}_{x\in\mathcal{U}} \max_{i=1,\dots,d} \left(\min \left\{ \langle \mathbf{w}_{i}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}} \right\} \cup \bigcup_{\substack{j=1,\dots,d\\ j\neq i}} \left\{ -\langle \mathbf{w}_{j}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}} \right\} \right). \tag{6.22}$$

For the pairwise decomposition techniques, the analogous multiclass selection strategy is given by

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x\in\mathcal{U}} \max_{i=1,\dots,d} \left(\min \bigcup_{j=i+1,\dots,d} \left\{ \frac{1+\langle \mathbf{w}_{ij}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}}}{2} \right\} \cup \bigcup_{j=1,\dots,i-1} \left\{ \frac{1-\langle \mathbf{w}_{ji}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}}}{2} \right\} \right) \tag{6.23}$$

$$= \operatorname*{argmin}_{x\in\mathcal{U}} \max_{i=1,\dots,d} \left(\min \bigcup_{j=i+1,\dots,d} \left\{ \langle \mathbf{w}_{ij}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}} \right\} \cup \bigcup_{j=1,\dots,i-1} \left\{ -\langle \mathbf{w}_{ji}^{(\mathrm{svm})}, \phi(x)\rangle_{\mathcal{F}} \right\} \right). \tag{6.24}$$

We refer to this multiclass selection strategy as BinMin strategy.

In contrast to the GLOBAL strategy, the BINMIN selection strategy focuses more aggressively on improving a single classifier among the set of classifiers rather than selecting examples which more uniformly reduce the volumes of the version spaces in a balanced manner.

It is readily verified that the BINMIN strategy reduces to the SIMPLE selection strategy for binary classification problems ($d = 2$) for both the pairwise and the one-versus-all decomposition techniques. In the following, we will only derive this property for the pairwise techniques since the one-versus-all decomposition technique requires an analogous line of reasoning:

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x \in \mathcal{U}} \max_{i=1,2} \left(\min \left\{ \langle \mathbf{w}_i^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}} \right\} \cup \bigcup_{\substack{j=1,2 \\ j \neq i}} \left\{ -\langle \mathbf{w}_j^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}} \right\} \right) \tag{6.25}$$

$$= \operatorname*{argmin}_{x \in \mathcal{U}} \max \left\{ \langle \mathbf{w}_1^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}}, -\langle \mathbf{w}_1^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}} \right\} \tag{6.26}$$

$$= \operatorname*{argmin}_{x \in \mathcal{U}} \left| \langle \mathbf{w}_1^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}} \right| . \tag{6.27}$$

Hence, all of the selection strategies which have been reviewed or newly proposed in this section for the generalization of pool-based active learning to multiclass learning can be considered as direct generalizations of the binary SIMPLE selection strategy.

6.4 Experiments

6.4.1 Experimental Setting

To compare the efficiency of all the considered selection strategies, we have conducted several experiments on multiclass benchmark datasets[5] that are publicly available from the UCI repository of machine learning databases (Blake and Merz, 1998) and from the Statlog collection (Michie et al., 1994). We considered both the GLOBAL and the BINMIN strategy based on the one-versus-all and the pairwise decomposition techniques (one-versus-one and DAG) for the selection of new examples. Furthermore, we investigated random selection (RANDOM) of new examples as a baseline strategy for each of the decomposition techniques.

For all problems which include a standard partition into training and test sets (*dna*, *satimage*, *letter*, *shuttle*), the selection strategies were initialized using a randomly drawn set of 20 examples (30 for the *letter* dataset, which contains 26 classes) from the training set with at least one example from each class. While new training examples were selected from the remaining training sets, the classification accuracy was estimated on the test sets after every 10 selection steps. We fixed the final number of labeled examples to 200 and averaged the results over 20 runs of random initialization. Each of the remaining benchmark datasets, which do not provide a

5. We selected all datasets with cardinality $>$ 500 from a recent study on multiclass classification (Hsu and Lin, 2002a).

	Strategy	*vowel*	*vehicle*	*segment*	*dna*	*satimage*	*letter*	*shuttle*
One vs All	Random	0.670 ±0.004	0.741 ±0.002	0.893 ±0.001	0.861 ±0.003	0.828 ±0.002	**0.590** ±0.003	**0.951** ±0.003
	Global	0.705 ±0.003	0.737 ±0.003	0.876 ±0.003	**0.908** ±0.002	0.829 ±0.004	0.524 ±0.011	0.938 ±0.006
	BinMin	**0.753** ±0.003	**0.767** ±0.002	**0.919** ±0.001	0.904 ±0.002	**0.851** ±0.001	0.516 ±0.008	0.936 ±0.005
One vs One	Random	0.732 ±0.004	0.735 ±0.003	0.904 ±0.002	0.866 ±0.002	0.836 ±0.002	0.606 ±0.006	0.960 ±0.003
	Global	0.744 ±0.005	0.720 ±0.003	0.881 ±0.003	**0.907** ±0.002	0.827 ±0.006	0.506 ±0.009	0.864 ±0.026
	BinMin	**0.844** ±0.003	**0.745** ±0.003	**0.941** ±0.001	0.906 ±0.002	**0.866** ±0.002	**0.609** ±0.006	**0.993** ±0.002
DAG	Random	0.739 ±0.004	0.734 ±0.002	0.900 ±0.002	0.870 ±0.002	0.834 ±0.002	**0.610** ±0.005	0.961 ±0.002
	Global	0.738 ±0.004	0.718 ±0.003	0.880 ±0.003	**0.904** ±0.001	0.827 ±0.007	0.474 ±0.011	0.881 ±0.022
	BinMin	**0.848** ±0.003	**0.749** ±0.003	**0.941** ±0.001	**0.904** ±0.002	**0.854** ±0.002	0.609 ±0.004	**0.995** ±0.001

Table 6.1 *This table shows the estimated final classification accuracy results and the corresponding standard error of the mean estimates. For each decomposition technique, the best result is indicated in bold face.*

standard partition, was randomly split 100 times into a training and a test set of equal size. In an analogous manner to the first category of benchmark datasets, we employed initial sets of cardinality 20 which were drawn from the training sets. Furthermore, we fixed the number of labeled examples to 100 for these smaller datasets and averaged the results over all 100 runs of random initialization and separation into training and test sets.

In our experiments, we used a modified version of Libsvm[6] (Hsu and Lin, 2002a) which learns support vector machines without bias and L2-loss in compliance with our theoretical line of reasoning. We used RBF kernels with the default choice of $\gamma = \frac{1}{\#(\text{input features})}$ and $C = 100$.

6.4.2 Experimental Results

Remember that the choice of kernel has not been optimized with respect to the given problems and multiclass decomposition techniques. Therefore, we refrain from a quantitative comparison between different multiclass techniques and restrict our presentation to a comparison of selection strategies based on the same decomposition technique. While the complete set of learning curves is given in Appendix B, we

6. We have to thank Chih-Len Lin for valuable comments on the implementation.

	Strategy	*vowel*	*vehicle*	*segment*	*dna*	*satimage*	*letter*	*shuttle*
One	Random	0.000	0.000	0.000	0.000	0.000	**0.000**	**0.000**
vs	Global	0.032	-0.005	-0.010	**0.034**	-0.013	-0.025	-0.013
All	BinMin	**0.047**	**0.020**	**0.014**	**0.034**	**0.017**	-0.049	-0.028
One	Random	0.000	0.000	0.000	0.000	0.000	**0.000**	**0.000**
vs	Global	0.020	-0.009	-0.007	**0.029**	-0.014	-0.054	-0.058
One	BinMin	**0.069**	**0.012**	**0.030**	**0.029**	**0.018**	-0.002	-0.002
	Random	0.000	0.000	0.000	0.000	0.000	0.000	0.000
DAG	Global	0.009	-0.009	-0.006	**0.031**	-0.008	-0.073	-0.059
	BinMin	**0.069**	**0.016**	**0.035**	0.028	**0.014**	**0.002**	**0.009**

Table 6.2 *This table shows the difference of classification accuracy compared to a random learner averaged over all estimation points. For each decomposition technique the best result is indicated in bold face.*

focus our presentation of the experimental results on two important measures that summarize the efficiency of a given selection strategy. The first measure is the average classification accuracy, i.e., the proportion of correctly classified test examples, at the end of the experiments with 100 and 200 labeled examples, respectively. Table 6.1 shows average classification accuracy results and associated standard error of the mean estimates. In the case of the one-versus-one and the DAG pairwise technique, the BinMin strategy achieves the best result in six out of seven problems (with one tie for DAG decomposition and the *dna* problem). For the only dataset where the BinMin selection strategy is outperformed, its average accuracy is very close to the winning strategy. In the case of one-versus-all classification, the BinMin strategy is the best strategy in four out of seven problems, while the Random and the Global strategies achieve the best results on two datasets and one dataset, respectively.

As a second performance figure, we consider the average difference in classification accuracy between a given selection strategy and the Random strategy. Assume that the classification accuracy is evaluated at l points (see the preceding section for details) and denote the corresponding average accuracy of the Random strategy by $a_1^{(\mathrm{RND})}, \dots, a_l^{(\mathrm{RND})}$ and the average accuracy of the given strategy by $a_1, \dots, a_l$. Then, the average difference is given by

$$\frac{1}{l}\sum_{i=1}^{l}\left(a_i - a_i^{(\mathrm{RND})}\right) \in [-1, 1]. \tag{6.28}$$

Clearly, this measure evaluates to zero for the Random strategy, while positive values indicate higher and negative values lower average accuracy in comparison to the Random strategy. While the final accuracy estimates compare the selection strategies at a single evaluation point, this figure captures the overall progress of learning with respect to classification accuracy. Table 6.2 shows that the BinMin strategy in combination with the one-versus-one technique achieves the maximum score in five out of seven problems (with one tie) and in the case of DAG classification in six out of seven problems. For one-versus-all decomposition, the BinMin selection strategy is the best strategy on five benchmark datasets.

In our experiments, both the original Global strategy and its generalization to pairwise decomposition were outperformed by the Random selection strategy on most of the problems suggesting that this strategy, which aims at selecting examples that are in some sense informative for all binary problems, is problematic. In contrast to this, the BinMin selection strategy based on all considered decomposition techniques clearly indicates that is possible to reduce the labeling effort in multiclass classification learning by more aggressively focusing on improving individual classifiers rather than selecting examples which more uniformly reduce the volumes of the version spaces in a balanced manner.

6.5 Discussion

We have proposed a novel extension of pool-based active learning to multiclass classification based on both the one-versus-all classification technique and two pairwise decomposition methods. A variety of experimental results demonstrates that our approach to active learning yields a significant reduction of the labeling effort in multiclass learning and outperforms a previous generalization of binary active learning proposed in (Tong, 2001).

Allwein et al. (2000); Dietterich and Bakiri (1995) proposed a general technique for constructing multiclass classifiers based on sets of binary classifiers which distinguish between subsets of classes. The decomposition of classes is encoded as a so-called Error-Correcting Output Code (ECOC) matrix and subsumes the one-versus-one and one-versus-all techniques as special cases. Yan et al. (2003) investigated the problem of labeling video data and proposed a framework for selection strategies in the context of ECOC multiclass learning which contains a selection strategy that can be viewed as an approximation to the BinMin strategy (for one-versus-all and one-versus-one decomposition of multiclass problems). In an analogous manner to (Yan et al., 2003), it is possible to generalize the BinMin strategy to the ECOC technique.

The constraint classification approach (Har-Peled et al., 2002) provides a framework for solving a variety of problems of more complex categories. In particular, a distinguishing feature of this technique in contrast to the decomposition techniques presented in this chapter is a transformation which encodes multiclass problems as single binary classification problems. Apart from computational drawbacks, there exists an appealing connection to the proposed BinMin strategy (for one-versus-all decomposition) as it is readily verified that this selection strategy can be viewed as an analogon to the Simple strategy with respect to a natural generalization of the notion of the margin for the underlying binary classification problem. As this property also holds for the generalization of pool-based active learning to label ranking functions, it may provide the basis for a unified theoretical analysis of active learning.

7 Label Ranking Learning

Overview

The effort necessary to construct sets of labeled examples in a supervised learning scenario is often disregarded, though in many applications, it is a time-consuming and expensive procedure. While this process already constitutes a major issue in classification learning, it becomes an even more substantial matter of relevance in label ranking learning, which considers the more complex target domain of total orders over a fixed set of alternatives. We introduce a novel generalization of pool-based active learning to reduce the labeling effort based on both the pairwise ranking and the constraint classification techniques for the representation of label ranking functions.

A subset of the results presented in this chapter was published in (Brinker, 2003a,b, 2004a).

7.1 Introduction

The increasing shift from predetermined and static to personalized and highly adaptive systems has affected various areas of application. Techniques for individualizing application flows and for incorporating user preferences have produced more efficient systems in the domains of e-commerce (Riecken, 2000), information retrieval and design of user interfaces (Langley, 1997), among others. A fundamental prerequisite of systems which consider individuals rather than predefined standard users is the ability to efficiently acquire accurate preference models.

We consider a special category of preference learning problems, so-called label ranking problems. The fundamental objective in (label) ranking learning is to learn a mapping from a given input space to the set of total orders over a finite and a priori fixed set of labels, which is also referred to as the set of alternatives.

Example

Let us consider the following example to illustrate the label ranking setting from a more concrete point of view: Suppose we are given a set of customers which are represented by features (such as age, income, family status, etc.) and their ordered preferences over a set of car models {Porsche, Toyota, Ford, Lada}, i.e., each customer is associated with a permutation over the set of car models. Then, the learning task consists in inducing a mapping from customers to the set of permutations of these car models for the accurate prediction of the order of preference for unseen customers.

In contrast to a classification setting, we are not only interested in the top-ranked alternative, but also in the complete preference order. By incorporating this additional information, we are able to build more powerful and more flexible prediction systems which, for instance, are able to make preference suggestions in situations where the top-ranked alternative is currently not available for some reason, e.g., an article is out of stock.

As in the case of multiclass classification learning, there exist different approaches to reduce ranking problems to sets of binary classification problems.[1] As a straightforward generalization of one-versus-one (multiclass) classification, ranking problems can be decomposed into binary classification problems by considering all pairwise decision problems between two alternatives (Fürnkranz and Hüllermeier, 2003a). Each pairwise decision problem is treated independently as a binary classification problem and label rankings are predicted by means of a voting procedure. An alternative approach to the expression of a ranking problem in terms of a single binary classification problem was proposed by Har-Peled et al. (2002). In constraint classification, the process of transforming the initial ranking problem involves both embedding the training data in a higher dimensional space and expanding single ranking examples into multiple binary classification examples.

Both approaches consider the problem of learning a ranking function in a supervised batch learning scenario. Hence, it is assumed that we are given a training set of patterns associated with the corresponding target permutations. However, in many applications, the task of assigning permutations to patterns cannot be performed automatically, but involves human decisions or cost-intensive interviews. Therefore, it is a time-consuming and expensive procedure. While the process of labeling already constitutes a major issue in classification learning, it becomes an even more substantial matter of relevance when dealing with the more complex target domain of the set of permutations: To ask for a customer's top preference is less expensive than to request a complete preference order over all possible alternatives.

We consider the pool-based active learning framework[2] for the reduction of the labeling effort in label ranking learning. In the field of kernel machines, active learning has been successfully applied to classification problems to reduce the labeling effort (Tong and Koller, 2001c; Campbell et al., 2000; Schohn and Cohn, 2000). All these approaches are restricted to either binary or multiclass classification problems and do not extend to ranking problems. We propose a novel extension of active learning to label ranking problems. Employing both the pairwise ranking (Fürnkranz and Hüllermeier, 2003a) and the constraint classification techniques (Har-Peled et al., 2002), we propose heuristic strategies for selecting new training examples which are derived on the basis of a similar line of reasoning as our extension to active multiclass learning presented in the previous chapter. We have conducted several experiments to evaluate the efficiency of our approach. The experimental results demonstrate that it is possible to achieve a significant reduction of the labeling effort in ranking learning.

The present chapter is organized as follows: The subsequent sections discuss the aforementioned techniques for expressing and training label ranking functions with an emphasis on the efficient implementation of the constraint classification

1. See the previous chapter for details on multiclass learning.
2. Section 3.2 presents a detailed introduction to the pool-based active learning framework.

technique. In Section 7.4, we investigate active learning in the case of ranking problems and introduce our novel generalization. Section 7.5 presents experimental results conducted on a number of synthetic datasets and demonstrates the benefits of our approach. Finally, in Section 7.6, we summarize our approach, point out references to related research and discuss further research topics.

7.2 Ranking Problems

The subsequent sections recapitulate the constraint classification and the pairwise ranking techniques for solving ranking problems. Furthermore, we focus on the efficient implementation of the constraint classification technique and discuss how to reduce the computational complexity and the memory requirements of this approach.

The learning problem that we are investigating can be stated in a more formal way as follows: Based on a given training set of labeled examples

$$\mathcal{L} = \{(x_1, y_1), \dots, (x_m, y_m)\} \subset \mathcal{X} \times \mathcal{S}^{(d)} \tag{7.1}$$

Learning Setting

where $\mathcal{X}$ denotes the nonempty input space and $\mathcal{S}^{(d)}$ denotes the symmetric group of degree d, i.e.,

$$\mathcal{S}^{(d)} \overset{\text{def}}{=} \big\{ y \in \mathbb{N}^d \,|\, \{[y]_1, \dots, [y]_d\} = \{1, \dots, d\} \big\}, \tag{7.2}$$

we seek to induce a ranking function

$$h : \mathcal{X} \to \mathcal{S}^{(d)} \tag{7.3}$$

for the accurate prediction of unseen patterns.[3] In other words, the target objects to be learned are total orders over a finite and a priori fixed set of alternatives which are represented as permutations. A given permutation y is interpreted as follows: If alternative i precedes alternative j in $y = ([y]_1, \dots, [y]_d)$, then alternative i is preferred over alternative j. This representation is based on the reasonable assumption that the order over the set of labels is transitive, irreflexive and anti-symmetric (for a more detailed discussion see (Fürnkranz and Hüllermeier, 2003a)).

Similarity Measure

In compliance with related research, we employ the Spearman rank correlation coefficient (rank correlation) as the evaluation metric to measure the similarity between true rankings y and predicted rankings y'. It was originally proposed as a nonparametric rank statistic by Spearman (1904) to measure the strength of the associations between two variables (Lehmann and D'Abrera, 1998). Formally, the Spearman rank correlation coefficient is defined as follows:

3. Note that in contrast to previous chapters, h denotes a ranking function and not a classifier. To distinguish between ranking functions and binary classifiers, we impose the convention that binary classifiers always take their weight vectors as subscripts in this chapter.

Definition 7.1 Spearman Rank Correlation Coefficient (Spearman, 1904)
The Spearman rank correlation coefficient (rank correlation) is defined as

$$\mathrm{s}_{\mathrm{rank}} : \mathcal{S}^{(d)} \times \mathcal{S}^{(d)} \to [-1, +1] \tag{7.4}$$

$$(y, y') \mapsto 1 - \frac{6 \sum_{i=1}^{d} ([y]_i - [y']_i)^2}{d(d^2-1)}. \tag{7.5}$$

The rank correlation calculates the sum of squared rank distances and is normalized such that it evaluates to -1 for reversed preference orders and to $+1$ for identical orders. Note that we do not refer to the rank correlation as a loss function because it operates on a reversed scale, i.e., high rank correlation corresponds to low loss. However, a straightforward transformation integrates the rank correlation into the loss/risk-framework as defined in Section 1.1.

In the subsequent sections, we will review techniques for the representation of ranking functions which are based on linear binary classifiers and binary classifiers in general, respectively. Both to increase expressivity and to make use of typically highly accurate binary base classifiers, we embed patterns from the input space $\mathcal{X}$ using a kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ into the kernel-induced feature space $\mathcal{F}$ where the kernel feature map is denoted by $\phi : \mathcal{X} \to \mathcal{F}$.

7.2.1 Constraint Classification

The constraint classification approach (Har-Peled et al., 2002) provides an elegant framework for solving a variety of more complex learning problems, such as label ranking and multilabel problems, based on (binary) linear classifiers. In this section, we restrict our discussion to ranking problems in the linear case, i.e., $\mathcal{X} = \mathbb{R}^N$ endowed with the canonical dot product $\langle \cdot, \cdot \rangle$ to avoid a lengthy and rather technical presentation. We will comment on how to efficiently integrate the concept of kernels in the subsequent section.

Let us proceed to a formal definition of the considered hypothesis class of ranking functions:

Definition 7.2 Linear Sorting Function
Suppose we are given weight vectors $\mathbf{w}_1, \ldots, \mathbf{w}_d \in \mathbb{R}^N$ of linear functions. Then, the corresponding linear sorting function is defined as

$$h : \mathbb{R}^N \to \mathcal{S}^{(d)} \tag{7.6}$$

$$\mathbf{x} \mapsto \operatorname*{argsort}_{i=1,\ldots,d} \langle \mathbf{w}_i, \mathbf{x} \rangle \tag{7.7}$$

where argsort *returns a permutation of* $\{1, \ldots, d\}$*, such that i precedes j if* $\langle \mathbf{w}_i, \mathbf{x} \rangle > \langle \mathbf{w}_j, \mathbf{x} \rangle$ *(in the case of equality, i precedes j if $i < j$).*

The process of transforming the initial ranking problem involves both embedding the training data in a higher dimensional space and expanding single ranking examples into multiple binary classification examples: Let $\vartheta(\mathbf{x}, i)$ denote an embedding of $\mathbf{x}$ in $\mathbb{R}^{Nd}$ such that the features of $\mathbf{x} = ([\mathbf{x}]_1, \ldots, [\mathbf{x}]_N)^\top$ are copied into the features

$(i-1)N+1, \ldots, i\,N$ of the augmented vector and the remaining features are set to 0.

We expand each ranking example $(\mathbf{x}, y)$ into a set $\mathcal{L}^{+}(\mathbf{x}, y)$ of $d-1$ positive binary classification examples in $\mathbb{R}^{Nd} \times \{+1\}$

$$\mathcal{L}^{+}(\mathbf{x}, y) \overset{\text{def}}{=} \bigcup_{i=1,\ldots,d-1} \left\{ \left(\vartheta(\mathbf{x}, [y]_i) - \vartheta(\mathbf{x}, [y]_{i+1}), +1 \right) \right\} \tag{7.8}$$

and a set of $d-1$ negative examples in $\mathbb{R}^{Nd} \times \{-1\}$

$$\mathcal{L}^{-}(\mathbf{x}, y) \overset{\text{def}}{=} \bigcup_{i=1,\ldots,d-1} \left\{ \left(\vartheta(\mathbf{x}, [y]_{i+1}) - \vartheta(\mathbf{x}, [y]_i), -1 \right) \right\}. \tag{7.9}$$

The transformed training set $\bar{\mathcal{L}}$ is defined as the union of all expanded examples:

$$\bar{\mathcal{L}} \overset{\text{def}}{=} \bigcup_{i=1,\ldots,m} \mathcal{L}^{+}(\mathbf{x}_i, y_i) \cup \mathcal{L}^{-}(\mathbf{x}_i, y_i). \tag{7.10}$$

Hence, it holds for the cardinality of $\bar{\mathcal{L}}$ that $|\bar{\mathcal{L}}| = 2(d-1)m$.

Suppose we employ an arbitrary linear learning algorithm to calculate a classifier $h_{\bar{\mathbf{w}}}(\cdot) = \operatorname{sign}(\langle \bar{\mathbf{w}}, \cdot \rangle)$ (with $\bar{\mathbf{w}} \in \mathbb{R}^{Nd}$) consistent with the binary training set $\bar{\mathcal{L}}$, i.e.,

$$h_{\bar{\mathbf{w}}}(\bar{\mathbf{x}}) = \operatorname{sign}(\langle \bar{\mathbf{w}}, \bar{\mathbf{x}} \rangle) = \bar{y} \quad \text{for all } (\bar{\mathbf{x}}, \bar{y}) \in \bar{\mathcal{L}}. \tag{7.11}$$

Furthermore, we consider $\bar{\mathbf{w}}$ as the concatenation of d N-dimensional weight vectors $\mathbf{w}_1, \ldots, \mathbf{w}_d$ and by this means define a linear sorting function:

$$h : \mathbb{R}^N \to \mathcal{S}^{(d)} \tag{7.12}$$

$$\mathbf{x} \mapsto \operatorname*{argsort}_{i=1,\ldots,d} \langle \mathbf{w}_i, \mathbf{x} \rangle. \tag{7.13}$$

Since $h_{\bar{\mathbf{w}}}(\cdot) = \operatorname{sign}(\langle \bar{\mathbf{w}}, \cdot \rangle)$ is a consistent linear classifier, it follows that for all $(\bar{\mathbf{x}}, +1) = \left(\vartheta(\mathbf{x}, [y]_i) - \vartheta(\mathbf{x}, [y]_{i+1}), +1 \right) \in \bar{\mathcal{L}}$,

$$\langle \bar{\mathbf{w}}, \bar{\mathbf{x}} \rangle = \langle \mathbf{w}_{[y]_i}, \mathbf{x} \rangle - \langle \mathbf{w}_{[y]_{i+1}}, \mathbf{x} \rangle > 0. \tag{7.14}$$

Hence, the linear sorting function h correctly orders the two alternatives $[y]_i$ and $[y]_{i+1}$. Moreover, since all constraints on consecutive alternatives are encoded as positive binary examples, h is consistent with the original set of label ranking examples.

In fact, both the set of positive and the set of negative examples are sufficient to ensure consistency of the linear sorting function h. While the expansion into both positive and negative examples makes this framework applicable regardless of the underlying linear learning algorithm, we can exploit the fact that the training set is symmetric around the origin. Har-Peled et al. (2002) proposed a perceptron-style algorithm which requires only the positive set of expanded examples to learn a linear sorting function.

In the case of support vector machines as base classifiers, the one-class algorithm proposed by Schölkopf et al. (2001) can be modified such that it operates exclusively on the positive set of examples. While we do not make use of this modification because our theoretical reasoning is based on a $C-$parametrization whereas (Schölkopf et al.,

2001) considers a $\nu-$parametrization, we will nevertheless discuss this novel modification in the following section since it is extremely useful in a standard batch learning setting to reduce the computational demands for training linear sorting functions.

Apart from theoretical analysis, one should not implement the constraint classification framework by explicitly expanding examples. It is more efficient with respect to both computational and memory demands to store the constraints imposed by consecutive alternatives and references to original examples in expanded examples and to employ a suitable (meta-)kernel. Furthermore, the standard kernelization technique can be incorporated at this point in an elegant manner. We will discuss these implementation issues in more details in the subsequent section.

7.2.2 Implementation of Kernel Constraint Classification

The previous section introduced the constraint classification technique for solving label ranking problems. We restricted our presentation to the input space $\mathcal{X} = \mathbb{R}^N$ endowed with the canonical dot product to provide a more intuitive and less technical presentation. In this section, we incorporate the concept of kernels into the constraint classification technique in order to extend the applicability of this approach to a wider field of application.

We embed patterns from the input space $\mathcal{X}$ using a kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ into the kernel-induced feature space $\mathcal{F}$ where the kernel feature map is denoted by $\phi : \mathcal{X} \to \mathcal{F}$ and consider the class of linear sorting functions in the feature space $\mathcal{F}$ as our hypothesis space. In other words, we focus on ranking functions of the following structure:

$$h : \mathcal{X} \to \mathcal{S}^{(d)} \tag{7.15}$$

$$x \mapsto \operatorname*{argsort}_{i=1,\dots,d} \langle \mathbf{w}_i, \phi(x) \rangle_{\mathcal{F}} \tag{7.16}$$

where $\mathbf{w}_1, \dots, \mathbf{w}_d \in \mathcal{F}$ denote weight vectors of linear functions in the kernel-induced feature space $\mathcal{F}$.

As in the linear case, we transform the initial ranking problem by embedding the training data into a higher dimensional space and expand single ranking examples into multiple binary classification examples. Consider a training example $(x, y) \in \mathcal{X} \times \mathcal{S}^{(d)}$ which is mapped into the feature space $\mathcal{F}$ by $x \mapsto \phi(x)$. In the following, we expand this example into $2(d-1)$ binary classification examples in $\mathcal{F}^d \times \{-1, +1\}$. To this end, we define a mapping

$$\psi^+ : \mathcal{X} \times \mathcal{S}^{(d)} \times \{1, \dots, d-1\} \to \mathcal{F}^d \tag{7.17}$$

where for given $x \in \mathcal{X}$, $y \in \mathcal{S}^{(d)}$ and $i \in \{1, \dots, d-1\}$, $\psi^+(x, y, i)$ is defined componentwise by

$$[\psi^+(x, y, i)]_n \overset{\text{def}}{=} \begin{cases} \phi(x) & \text{if } n = [y]_i, \\ -\phi(x) & \text{if } n = [y]_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \quad \text{for } n = 1, \dots, d. \tag{7.18}$$

Note that the components of $\psi^+(x, y, i)$ are not real-valued features in general, but may also be elements of an infinite-dimensional kernel feature space. Moreover, in the above definition, 0 denotes the zero element in $\mathcal{F}$. $\psi^-(x, y, i)$ is defined in an analogous manner with a componentwise change of sign.

We proceed to the definition of the expansion of a given ranking example (x, y) into a set $\mathcal{L}^+(x, y)$ of $d-1$ positive binary classification examples in $\mathcal{F}^d \times \{+1\}$

$$\mathcal{L}^+(x, y) \overset{\text{def}}{=} \bigcup_{i=1,\dots,d-1} \{(\psi^+(x, y, i), +1)\} \tag{7.19}$$

and a set of $d-1$ negative examples in $\mathcal{F}^d \times \{-1\}$

$$\mathcal{L}^-(x, y) \overset{\text{def}}{=} \bigcup_{i=1,\dots,d-1} \{(\psi^-(x, y, i), -1)\}. \tag{7.20}$$

The expanded training set $\bar{\mathcal{L}}$ is defined as the union of all expanded examples:

$$\bar{\mathcal{L}} \overset{\text{def}}{=} \bigcup_{i=1,\dots,m} \mathcal{L}^+(x_i, y_i) \cup \mathcal{L}^-(x_i, y_i). \tag{7.21}$$

To complete the embedding, we define a (meta-)kernel $k_{(\text{meta})}$ on elements in $\mathcal{F}^d$ which are included in the expanded training set $\bar{\mathcal{L}}$: Suppose we are given two arbitrary examples $(\bar{x}, y), (\bar{x}', y') \in \bar{\mathcal{L}}$. Then, the components of $\bar{x}$ and $\bar{x}'$ can be expressed as

$$[\bar{x}]_n = \beta_n \phi(x_{l_n}) \quad \text{and} \quad [\bar{x}']_n = \beta'_n \phi(x_{l'_n}) \tag{7.22}$$

with $\beta_n, \beta'_n \in \{-1, 0, +1\}$ and $l_n, l'_n \in \{1, \dots, m\}$. The kernel $k_{(\text{meta})}$ is defined as follows:

$$k_{(\text{meta})} : \mathcal{F}^d \times \mathcal{F}^d \to \mathbb{R} \tag{7.23}$$

$$(\bar{x}, \bar{x}') \mapsto \langle \bar{x}, \bar{x}' \rangle_{\mathcal{F}^d} \overset{\text{def}}{=} \sum_{n=1}^{d} \beta_n \beta'_n k(x_{l_n}, x_{l'_n}). \tag{7.24}$$

It can be readily verified that the above definition of $k_{(\text{meta})}$ yields a positive semidefinite kernel if the original kernel k is positive semidefinite (see (Herbrich, 2002) for details on combining kernel functions).

Note that there are at most two nonzero terms in this sum, independently of the number d of alternatives. Moreover, the same two original training examples are involved in every nonzero term. Thus, by exploiting the sparsity of the expanded examples and the symmetry of the kernel we can calculate dot products at practically the same computational cost as in the input space $\mathcal{F}$. Similarly, an indirect sparse encoding yields only a constant demand for memory space for each expanded example, provided that we store a copy of the initial dataset. More technically, ψ^+ and ψ^- have to be redefined as follows:

$$\psi^+ : \mathcal{L} \times \mathcal{S}^{(d)} \times \{1, \dots, d-1\} \to \{1, \dots, m\} \times \{1, \dots, d\} \times \{1, \dots, d\} \tag{7.25}$$

$$(x_j, y, i) \mapsto (j, [y]_i, [y]_{i+1}) \tag{7.26}$$

and

$$\psi^- : \mathcal{L} \times \mathcal{S}^{(d)} \times \{1, \ldots, d-1\} \to \{1, \ldots, m\} \times \{1, \ldots, d\} \times \{1, \ldots, d\} \quad (7.27)$$
$$(x_j, y, i) \mapsto (j, [y]_{i+1}, [y]_i). \quad (7.28)$$

The corresponding optimized kernel is given by

$$k_{(\text{meta})}\big((j, i_1, i_2), (j', i_1', i_2')\big) \overset{\text{def}}{=} (\delta_{i_1, i_1'} - \delta_{i_1, i_2'} - \delta_{i_2, i_1'} + \delta_{i_2, i_2'})\, k(x_j, x_{j'}) \quad (7.29)$$

where δ denotes the Kronecker delta function.

Let us return to the initial definition of the kernel on $\mathcal{F}^d$ for the remainder of this section to avoid an unnecessarily technical presentation. Under the assumption that the kernel is chosen such that the expanded binary classification problem is linearly separable, we can train a kernel machine in $\mathcal{F}^d$

$$h_{\bar{w}} : \mathcal{F}^d \to \{-1, +1\} \quad (7.30)$$
$$\bar{x} \mapsto \operatorname{sign}\big(\langle \bar{\mathbf{w}}, \bar{x} \rangle_{\mathcal{F}^d}\big) \quad (7.31)$$

which is consistent with the binary training set $\bar{\mathcal{L}}$. The components of training examples $(\bar{x}_i, \bar{y}_i) \in \bar{\mathcal{L}}$ are denoted by $[\bar{x}_i]_n = \beta_{i,n}\phi(x_{l_{i,n}})$ for $n = 1, \ldots, d$. The weight vector $\bar{\mathbf{w}}$ can be expanded in terms of the $2(d-1)m$ training examples:

$$\bar{\mathbf{w}} = \sum_{i=1}^{2(d-1)m} \alpha_i \bar{x}_i \quad (7.32)$$

where the components $n = 1, \ldots, d$ are given by

$$[\bar{\mathbf{w}}]_n = \sum_{i=1}^{2(d-1)m} \alpha_i \beta_{i,n}\, \phi(x_{l_{i,n}}). \quad (7.33)$$

Now, the weight vector $\bar{\mathbf{w}} \in \mathcal{F}^d$ of the linear classifier is decomposed into its d components to construct a linear sorting function:

$$h : \mathcal{X} \to \mathcal{S}^{(d)}$$
$$x \mapsto \operatorname*{argsort}_{n=1,\ldots,d} \langle [\bar{\mathbf{w}}]_n, \phi(x) \rangle_{\mathcal{F}} = \operatorname*{argsort}_{n=1,\ldots,d} \sum_{i=1}^{2(d-1)m} \alpha_i \beta_{i,n}\, k(x_{l_{i,n}}, x). \quad (7.34)$$

The ranking function h does not depend on direct computations of the kernel feature map anymore since it is expressed exclusively in terms of the kernel function.

The proof that h forms a consistent ranking function for the original ranking dataset follows the same line of reasoning as in the linear case, which was examined in the previous section.

In the remaining part of this section, we will sketch a modification of a one-class large margin algorithm proposed by Schölkopf et al. (2001) in the context of quantile estimation and novelty detection which requires only the positive set of examples to calculate a label ranking function.

Schölkopf et al. (2001) proposed the following quadratic program to separate a training set from the origin with maximum margin:

$$\underset{\substack{\bar{\mathbf{w}}\in\mathcal{F}^d,\,\rho\in\mathbb{R}\\ \xi_1,\ldots,\xi_{d(m-1)}\in\mathbb{R}}}{\text{minimize}} \quad \frac{1}{2}\|\bar{\mathbf{w}}\|^2_{\mathcal{F}^d} + \frac{1}{\nu d(m-1)}\sum_{i=1}^{d(m-1)} \xi_i - \rho \tag{7.35}$$

$$\text{subject to} \quad \langle \bar{\mathbf{w}}, \bar{x}_i\rangle_{\mathcal{F}^d} \geq \rho - \xi_i, \quad \xi_i \geq 0 \quad \text{for } i = 1, \ldots, d(m-1) \tag{7.36}$$

where the training set is assumed to be ordered such that

$$(\bar{x}_1, \bar{y}_1), \ldots, (\bar{x}_{d(m-1)}, \bar{y}_{d(m-1)}) \in \bar{\mathcal{L}}^+ \tag{7.37}$$

and

$$(\bar{x}_{d(m-1)+1}, \bar{y}_{d(m-1)+1}), \ldots, (\bar{x}_{2d(m-1)}, \bar{y}_{2d(m-1)}) \in \bar{\mathcal{L}}^- . \tag{7.38}$$

Here, the parameter $\nu \in (0, 1]$ controls the trade-off between the two conflicting objectives, large margin and correct separation. If we substitute the weight vector $\bar{\mathbf{w}}^*$ by $-\bar{\mathbf{w}}^*$ in the solution of the above-defined quadratic program, then it forms the solution for the negative dataset $\bar{\mathcal{L}}^-$ as a consequence of the symmetry of the datasets $\bar{\mathcal{L}}^+$ and $\bar{\mathcal{L}}^-$ around the origin. Therefore, the linear classifier $h_{\bar{\mathbf{w}}^*}$ forms the support vector machine solution for the entire training set $\bar{\mathcal{L}}$.

Not only does this modification substantially accelerate the training process because it operates on half the original dataset, but it also circumvents convergence problems that we encountered occasionally when solving the original symmetric problem with a standard solver for support vector machines.

7.2.3 Pairwise Ranking

Pairwise ranking (Fürnkranz and Hüllermeier, 2003a) is a generalization of one-versus-one multiclass classification (Knerr et al., 1990) which learns a separate binary classifier for each of the $d(d-1)/2$ pairs of class labels (alternatives). Each binary classifier $h_{\mathrm{w}_{ij}}$ (with $1 \leq i < j \leq d$) determines for a given example whether or not alternative i is preferred over alternative j. The training set for $h_{\mathrm{w}_{ij}}$ consists of the complete set of input patterns $\{x_1, \ldots, x_m\}$ with the binary class label of each input pattern x being assigned depending on whether i precedes j in the associated ranking $y = ([y]_1, \ldots, [y]_d)$ (positive class) or vice versa (negative class).

For the prediction of a ranking, Fürnkranz and Hüllermeier (2003a) suggested to evaluate all binary classifiers $h_{\mathrm{w}_{ij}}(x) \in \{-1, +1\}$ (with $1 \leq i < j \leq d$) and interpret the binary predictions as votes for alternative i and j, respectively. Finally, the set of alternatives is sorted in descending order with respect to the sum of votes. In the case of ties, though it might be suboptimal, we prioritize alternatives with lower indices as suggested by Hsu and Lin (2002a) for one-versus-one classification. Formally, this

Standard Pairwise Voting

aggregation technique can be stated as follows:

$$h : \mathcal{X} \to \mathcal{S}^{(d)} \tag{7.39}$$

$$x \mapsto \operatorname*{argsort}_{i=1,\ldots,d} \Big(\sum_{j=i+1}^{d} \max(h_{\mathrm{w}_{ij}}(x), 0) + \sum_{j=1}^{i-1} \max(-h_{\mathrm{w}_{ji}}(x), 0) \Big). \tag{7.40}$$

We refer to this prediction technique as standard pairwise voting (pw). Standard pairwise voting was proposed as a heuristic method for the aggregation of binary votes in pairwise ranking. Recently, there has been considerable progress in the theoretical foundation of this technique as it was proved by Hüllermeier and Fürnkranz (2004b) that pairwise voting effectively yields a risk minimizing prediction with respect to the sum of squared rank distances loss-function, which is an affine transformation of the Spearman rank correlation coefficient.

Pairwise ranking provides a framework that is applicable to the wide spectrum of underlying binary classifiers which are *class-symmetric*, i.e., the problem of discriminating class i against class j is identical to discriminating class j against class i (see also Section 6.2.1). It is readily verified that this assumption holds for support vector machines which will be considered as base classifiers in the following.

Aggregation of the predictions of the base classifiers is still an area of ongoing research in pairwise ranking (Hüllermeier and Fürnkranz, 2004a). Intuitively, whenever there is a close decision between two alternatives, i.e., the corresponding binary example is close to the classification boundary, it is not necessarily a favorable strategy to assign a complete vote to one of the alternatives. Therefore, in addition to the above-defined standard pairwise voting technique, we investigate a modified aggregation technique: Using an approach proposed by Platt (1999b) which conducts a logistic regression to convert functional distances into class probabilities, we estimate posterior (positive) class probabilities $f'_{\mathrm{w}_{ij}}(x) \in [0, 1]$ instead of binary class labels and assign partial votes to both alternatives:

Probabilistic Pairwise Voting

$$h' : \mathcal{X} \to \mathcal{S}^{(d)} \tag{7.41}$$

$$x \mapsto \operatorname*{argsort}_{i=1,\ldots,d} \Big(\sum_{j=i+1}^{d} f'_{\mathrm{w}_{ij}}(x) + \sum_{j=1}^{i-1} (1 - f'_{\mathrm{w}_{ji}}(x)) \Big). \tag{7.42}$$

We refer to this technique as probabilistic pairwise voting (ppw). The corresponding binary classifiers are given by $h'_{\mathrm{w}_{ij}}(\cdot) \overset{\text{def}}{=} \operatorname{sign}(f'_{\mathrm{w}_{ij}}(\cdot) - \frac{1}{2})$. Hüllermeier and Fürnkranz (2004a) conducted controlled experiments which suggest that it is favorable to employ the binarized standard pairwise voting technique if the base learner is sufficiently strong, while probabilistic pairwise voting was the superior technique in the case of a weaker base learner. Hüllermeier and Fürnkranz (2004a) provide an intuitive explanation for this empirical observation: Binarization can be considered as a reinforcement of the predictions of the base classifiers, which might be reasonable provided that these prediction are sufficiently reliable.

7.3 Computational Complexity

Let us consider a ranking problem consisting of m training examples where the number of alternatives is denoted by d in a standard supervised batch learning setting. In Section 7.2.1, we discussed the constraint classification technique proposed by Har-Peled et al. (2002) which transforms a given ranking problem into a single binary classification problem. The associated transformation process is based on an expansion of the ranking training set into $2(d-1)m$ binary classification examples and an embedding into the higher dimensional space $\mathcal{F}^d$.

However, as shown in Section 7.2.2, it is not necessary to explicitly increase the dimension of the problem by using an indirect sparse encoding. Moreover, independently of d, calculating dot products is as expensive as in the original feature space $\mathcal{F}$ in terms of computational time, and only a constant amount of memory space for each expanded example is required (if we preserve a copy of the original training set). Therefore, the computational complexity when using a kernel machine with training time $\mathcal{O}(n^\alpha)$ amounts to $\mathcal{O}((2dm)^\alpha)$ and $\mathcal{O}((dm)^\alpha)$, respectively, if we consider the one-class algorithm. Since the running time for training support vector machines is empirically estimated at $\mathcal{O}(n^2)$ (Platt, 1999a; Joachims, 1999a), learning a label ranking function is of order $\mathcal{O}(d^2m^2)$ in this particular case. The pairwise ranking technique requires computational time of equal order (Fürnkranz and Hüllermeier, 2003b). The prediction of a ranking is of computational complexity of order $\mathcal{O}(dm + d \log d)$ for constraint classification, whereas for pairwise ranking the computational complexity amounts to $\mathcal{O}(d^2m + d \log d)$.

Apart from theoretical analysis, in our experiments the straightforward implementation of the pairwise ranking technique was roughly two orders of magnitude faster than the constraint classification technique. Even though the one-class modification can substantially expedite the training process, the training time is still not competitive with pairwise ranking. Similarly, in multiclass learning all-in-one approaches tend to be less efficient in terms of training time than binary decomposition methods (Hsu and Lin, 2002a).

7.4 Active Ranking Learning

This section introduces novel heuristic active learning selection strategies for both the constraint classification and the pairwise ranking technique. We will follow a similar line of reasoning as for the generalization of pool-based active learning to multiclass classification in the previous chapter, which is based on the version space model for binary classification. The key idea for generalizing active learning both to the multiclass and ranking setting is to reduce these learning problems of more complex categories to binary classification learning, which has been extensively studied in the active learning framework.

Let us consider the constraint classification framework for the representation of ranking functions. As described in Sections 7.2.1 and 7.2.2, the hypothesis class of ranking functions in constraint classification is given by linear sorting functions in the kernel-induced feature space $\mathcal{F}$, i.e., ranking functions of the following structure:

$$h : \mathcal{X} \to \mathcal{S}^{(d)} \tag{7.43}$$
$$x \mapsto \operatorname*{argsort}_{i=1,\dots,d} \langle \mathbf{w}_i, \phi(x)\rangle_{\mathcal{F}} \tag{7.44}$$

with weight vectors $\mathbf{w}_1, \dots, \mathbf{w}_d \in \mathcal{F}$. Furthermore, each given ranking example (x, y) is expanded into a set of binary classification examples $\mathcal{L}^+(x, y) \cup \mathcal{L}^-(x, y)$ and the weight vectors of the linear sorting function are determined by the weight vector of the linear classifier trained on the entire expanded set of binary examples. In summary, the initial ranking problem is transformed into a single binary classification problem.

Active Constraint Classification

In analogy to the generalization of pool-based active learning to multiclass learning, we consider a best worst-case approach with respect to the volume reduction ratio of the version space on the corresponding binary classification problem. As in Section 6.3, the ratio of volume reduction when augmenting the training set by an additional labeled example $(\bar{x}, \bar{y})$ is approximated by $\frac{1+\bar{y}\langle \bar{\mathbf{w}}^{(\text{svm})}, \bar{x}\rangle_{\mathcal{F}^d}}{2}$ where $\bar{\mathbf{w}}^{(\text{svm})}$ denotes the weight vector of a support vector machine trained on the expanded set of binary examples. Hence, the selection strategy is given by:

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x\in\mathcal{U}} \max_{y\in\mathcal{S}^{(d)}} \min_{\substack{(\bar{x},\bar{y})\in \\ \mathcal{L}^+(x,y)\cup\mathcal{L}^-(x,y)}} \frac{1 + \bar{y}\langle \bar{\mathbf{w}}^{(\text{svm})}, \bar{x}\rangle_{\mathcal{F}^d}}{2} \tag{7.45}$$
$$= \operatorname*{argmin}_{x\in\mathcal{U}} \max_{y\in\mathcal{S}^{(d)}} \min_{\substack{(\bar{x},\bar{y})\in \\ \mathcal{L}^+(x,y)\cup\mathcal{L}^-(x,y)}} \bar{y}\langle \bar{\mathbf{w}}^{(\text{svm})}, \bar{x}\rangle_{\mathcal{F}^d} \tag{7.46}$$
$$= \operatorname*{argmin}_{x\in\mathcal{U}} \max_{y\in\mathcal{S}^{(d)}} \min_{(\bar{x},\bar{y})\in\mathcal{L}^+(x,y)} \bar{y}\langle \bar{\mathbf{w}}^{(\text{svm})}, \bar{x}\rangle_{\mathcal{F}^d} \tag{7.47}$$
$$= \operatorname*{argmin}_{x\in\mathcal{U}} \max_{y\in\mathcal{S}^{(d)}} \min_{i=1,\dots,d-1} \left(\langle[\bar{\mathbf{w}}^{(\text{svm})}]_{[y]_i}, \phi(x)\rangle_{\mathcal{F}} - \langle[\bar{\mathbf{w}}^{(\text{svm})}]_{[y]_{i+1}}, \phi(x)\rangle_{\mathcal{F}}\right) \tag{7.48}$$
$$= \operatorname*{argmin}_{x\in\mathcal{U}} \min_{\substack{i,j=1,\dots,d \\ i\neq j}} \left|\langle[\bar{\mathbf{w}}^{(\text{svm})}]_i, \phi(x)\rangle_{\mathcal{F}} - \langle[\bar{\mathbf{w}}^{(\text{svm})}]_j, \phi(x)\rangle_{\mathcal{F}}\right|. \tag{7.49}$$

Note that equality holds in (7.47) because of the symmetry of the positive and the negative binary dataset and equality holds in (7.48) as a consequence of the construction of the expanded dataset.

As an important consequence of the above simplification of the envisaged selection strategy, the computational complexity with respect to the number of alternatives d for evaluating the selection criterion for a single unlabeled example is reduced to $\mathcal{O}(d \log d)$ because it requires sorting the d real-valued outputs of the linear functions and performing a scan on the sorted list. Since in the original formulation (7.45) discrete optimization is conducted over a set of cardinality

Computational Complexity

$$|\mathcal{S}^{(d)}|2(d-1) = 2(d-1)d!, \tag{7.50}$$

it becomes evident that this simplification is crucial for the reduction of the computational complexity to a moderate level.

Moreover, there exists an appealing connection to the SIMPLE selection strategy for binary classification learning, if we consider the following generalized notion of the margin which was introduced by Har-Peled et al. (2002):

Definition 7.3 Generalized Margin
The margin $\rho_h : \mathcal{X} \times \mathcal{S}^{(d)} \to \mathbb{R}$ of a ranking example (x, y) with respect to the linear sorting function $h : \mathcal{X} \to \mathcal{S}^{(d)}$, $x \mapsto \operatorname{argsort}_{i=1,\dots,d} \langle \mathbf{w}_i, \phi(x) \rangle_{\mathcal{F}}$, is defined as

$$\rho_h(x, y) \overset{def}{=} \min_{i=1,\dots,d-1} \langle \mathbf{w}_{[y]_i}, \phi(x) \rangle_{\mathcal{F}} - \langle \mathbf{w}_{[y]_{i+1}}, \phi(x) \rangle_{\mathcal{F}}. \tag{7.51}$$

We rewrite (7.48) using the generalized notion of the margin with respect to the linear sorting function $h : \mathcal{X} \to \mathcal{S}^{(d)}$, $x \mapsto \operatorname{argsort}_{i=1,\dots,d} \langle [\bar{\mathbf{w}}^{(\mathrm{svm})}]_i, \phi(x) \rangle_{\mathcal{F}}$:

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x \in \mathcal{U}} \max_{y \in \mathcal{S}^{(d)}} \rho_h(x, y). \tag{7.52}$$

Hence, the proposed selection strategy can be viewed as an analogon to the Simple strategy with respect to the generalized notion of the margin.

Active Pairwise Ranking

The pairwise ranking technique learns a separate binary classifier for each pair of alternatives to solve a ranking problem. Hence, we are given a *set* of independently treated binary problems which is not directly amenable to analysis in the binary version space model. In order to derive a heuristic selection strategy, we consider a best worst-case approach with respect to maximum version space reduction for any of the version spaces corresponding to the binary classification problems. This generalization of pool-based active learning follows a similar line of reasoning as for the derivation of the BinMin strategy in one-versus-one multiclass learning.

In contrast to the constraint classification framework, approximate reduction ratios have to be evaluated on different binary classification problems. More precisely, for each unlabeled example and permutation, we have to calculate the approximate reduction ratio on $d(d-1)/2$ version spaces. Denoting binary base classifiers by $h_{\mathrm{w}_{ij}} : \mathcal{X} \to \{-1, +1\}$, $x \mapsto \operatorname{sign}(\langle \mathbf{w}_{ij}^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}})$ (for $1 \leq i < j \leq d$), this selection strategy is given by:

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x \in \mathcal{U}} \max_{y \in \mathcal{S}^{(d)}} \min_{1 \leq i < j \leq d} \frac{1 + y^{(ij)} \langle \mathbf{w}_{ij}^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}}}{2} \tag{7.53}$$

$$= \operatorname*{argmin}_{x \in \mathcal{U}} \max_{y \in \mathcal{S}^{(d)}} \min_{1 \leq i < j \leq d} y^{(ij)} \langle \mathbf{w}_{ij}^{(\mathrm{svm})}, \phi(x) \rangle_{\mathcal{F}} \tag{7.54}$$

where the binary class label $y^{(ij)}$ associated with the pattern x in the problem alternative i-versus-alternative j is determined by the corresponding ranking y:

$$y^{(ij)} \overset{\mathrm{def}}{=} \begin{cases} +1 & \text{if } i \text{ precedes } j \text{ in } y, \\ -1 & \text{if } j \text{ precedes } i \text{ in } y. \end{cases} \quad (\text{for } 1 \leq i < j \leq d). \tag{7.55}$$

Unfortunately, it is computationally very expensive to compute this selection strategy for any but small d since for each unlabeled example x, optimization is performed over $|\mathcal{S}^{(d)}|(d-1)d/2 = d!(d-1)d/2$ possibilities. Therefore, we consider the following approximation to simplify this selection strategy and expedite the evaluation process: We approximate the permutation yielding maximum minimum binary margin by the minimum binary margin of the predicted permutation of the given unlabeled ranking example. Denoting the corresponding binary class label of the predicted ranking by $h(x)^{(ij)}$ in analogy to (7.55), the selection strategy simplifies to:

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x \in \mathcal{U}} \Big(\min_{1 \leq i < j \leq d} h(x)^{(ij)} \langle \mathbf{w}_{ij}^{(\text{svm})}, \phi(x) \rangle_{\mathcal{F}} \Big). \tag{7.56}$$

Here, the computational complexity with respect to the number of alternatives reduces to $\mathcal{O}(d^2)$ for each unlabeled example.

In the case of the probabilistic pairwise voting technique, we consider an analogous best worst-case approach with respect to minimum binary class probabilities. Formally, this selection strategy is given by

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x \in \mathcal{U}} \left(\min_{1 \leq i < j \leq d} h'(x)^{(ij)} \Big(f'_{\mathrm{w}_{ij}^{(\text{svm})}}(x) - \frac{1}{2} \Big) \right) \tag{7.57}$$

as an approximation of the computationally more demanding strategy

$$\mathcal{U} \mapsto \operatorname*{argmin}_{x \in \mathcal{U}} \max_{y \in \mathcal{S}^{(d)}} \left(\min_{1 \leq i < j \leq d} y^{(ij)} \Big(f'_{\mathrm{w}_{ij}^{(\text{svm})}}(x) - \frac{1}{2} \Big) \right) \tag{7.58}$$

where $h'(x)^{(ij)}$ is defined in an analogous manner to $h(x)^{(ij)}$ with respect to the probabilistic pairwise voting technique. As for standard pairwise voting, the computational complexity of (7.57) amounts to $\mathcal{O}(d^2)$.

7.5 Experiments

7.5.1 Experimental Setting

We have conducted a set of experiments using support vector machines[4] as binary linear learners to evaluate the performance of our novel selection strategies. In the absence of suitable real-world datasets, we generated artificial data originating from three different settings, which shall be described in the following paragraphs.

Linear: We replicated a setting proposed by Fürnkranz and Hüllermeier (2003a) which operates in the context of expected utility theory: An expected utility maximizing agent is given a set $\{1, \ldots, d\}$ of alternative actions to choose from. The agent faces a problem of *decision under uncertainty* where alternative i yields utility $[U]_{ij} \in \mathbb{R}$ if the world is in state $\omega_j \in \Omega = \{\omega_1, \ldots, \omega_N\}$. The probability of state ω_j is denoted by $[\mathbf{p}]_j$ and therefore the expected utility of alternative i evaluates to

$$\mathrm{E}(i) = \sum_{j=1}^{N} [\mathbf{p}]_j \, [U]_{ij}. \tag{7.59}$$

Expected utilities give rise to a natural order over the set of alternative actions. We

4. Our implementation is based on a modified version of the LIBSVM-library (Chang and Lin, 2001).

assume the set of alternatives to be in decreasing order with respect to expected utility in the following. Let us assume the probability vector $\mathbf{p} = ([\mathbf{p}]_1, \ldots, [\mathbf{p}]_N)^\top$ to be the input pattern of a ranking example where the number of alternatives d and the set of world states Ω are fixed and the $d \times N$ utility matrix U has independently and uniformly distributed entries $[U]_{ij} \in [0, 1]$. Then, for a given probability vector $\mathbf{p}$ the above-defined decision-theoretic scenario gives rise to an order over the set of alternative actions. Now, a set of m input patterns is independently drawn from a uniform distribution over $\{\mathbf{p} \in \mathbb{R}^N \mid [\mathbf{p}]_1 + \cdots + [\mathbf{p}]_N = 1,\ [\mathbf{p}]_i \geq 0 \text{ for } i = 1, \ldots, N\}$ and assigned to the corresponding permutations in order to generate a ranking dataset. Note that this setting corresponds to a noise-free scenario in the constraint classification framework (with linear kernels) since for a given input pattern $\mathbf{p}$ an alternative way of expressing the corresponding ranking is $y = \operatorname{argsort}_{i=1,\ldots,d} \langle \mathbf{u}_i^\top, \mathbf{p} \rangle$ (with $\mathbf{u}_i$ denoting the ith row vector of U). We conducted our experiments on this dataset using a linear kernel with penalty parameter $C = 100$.

MinMax: This setting is a modification of the previous setting and considers the (non-linear) preference relation generated by

$$\mathrm{E}(i) = \min_{j=1,\ldots,N} \max \left\{ [U]_{ij}, 1 - [\mathbf{p}]_j \right\}. \tag{7.60}$$

It can be viewed as a special case of a pessimistic criterion for the evaluation of the worth of an alternative in a possibilistic decision framework (Dubois et al., 2001). To stay consistent with the herein stated assumptions, for each input pattern $\mathbf{p}$ a single feature is randomly selected and set to 1. On this problem, we used an RBF kernel with the default choice of $\gamma = 1/N$ and penalty parameter $C = 100$.

QRank: We trained a naive Bayes classifier on the vehicle multiclass dataset from the UCI Repository of machine learning databases (Blake and Merz, 1998). For each example the set of possible class labels (alternatives) was ordered with respect to the a posteriori probabilities assigned by the naive Bayes classifier. From a more abstract point of view, we consider the problem of learning a qualitative replication of the order over a set of alternatives induced by a probabilistic classifier. As in the previous setting, we used an RBF kernel with the default value of $\gamma = 1/N$ and $C = 100$.

For both the Linear and the MinMax scenario, we fixed the number of input features to $N = 10$. For each number of alternatives $d \in \{5, 10, 15, 20\}$, we generated 100 different datasets consisting of 2000 examples, each dataset originating from a different utility matrix U. Moreover, each dataset was randomly split into a training set and a test set of equal size. In the QRank scenario, it is not possible to sample new data for each experimental run since the underlying probability distribution is unknown. Instead, the given dataset was randomly split into a training set and a test set of equal size for each run, in compliance with related research on real-world data in pool-based active learning. While new training examples were selected from the training sets, the average rank correlation (see Definition 7.1) was evaluated on the test sets.

In addition to our novel generalization of pool-based active learning to the pairwise ranking and constraint classification techniques, we investigated random selection of new examples as a baseline strategy for each of the approaches. We started with a randomly selected set of 10 labeled examples in all experiments and sequentially

Setting	Strategy[*]	10	20	30	40	50	60	70	80	90	100
Linear ($d = 10$)	random cc	0.419 ±0.010	0.594 ±0.007	0.697 ±0.005	0.753 ±0.004	0.793 ±0.004	0.821 ±0.003	0.842 ±0.003	0.856 ±0.003	0.869 ±0.002	0.879 ±0.002
	active cc	0.412 ±0.010	0.602 ±0.007	0.710 ±0.005	0.770 ±0.004	0.809 ±0.004	0.837 ±0.003	0.855 ±0.003	0.872 ±0.002	0.885 ±0.002	0.896 ±0.002
	random pw	0.356 ±0.013	0.463 ±0.011	0.532 ±0.010	0.582 ±0.009	0.622 ±0.008	0.651 ±0.007	0.677 ±0.007	0.699 ±0.006	0.717 ±0.006	0.730 ±0.005
	active pw	0.361 ±0.012	0.476 ±0.011	0.566 ±0.008	0.627 ±0.007	0.671 ±0.007	0.705 ±0.006	0.731 ±0.005	0.753 ±0.005	0.769 ±0.005	0.784 ±0.005
	random ppw	0.394 ±0.013	0.513 ±0.010	0.586 ±0.009	0.640 ±0.007	0.676 ±0.006	0.703 ±0.005	0.723 ±0.005	0.741 ±0.005	0.758 ±0.004	0.771 ±0.004
	active ppw	0.389 ±0.012	0.544 ±0.009	0.644 ±0.007	0.705 ±0.006	0.744 ±0.005	0.771 ±0.004	0.791 ±0.004	0.807 ±0.004	0.819 ±0.004	0.831 ±0.003
MinMax ($d = 10$)	random cc	0.502 ±0.007	0.634 ±0.007	0.718 ±0.006	0.767 ±0.005	0.803 ±0.005	0.833 ±0.004	0.853 ±0.004	0.868 ±0.003	0.881 ±0.003	0.891 ±0.003
	active cc	0.506 ±0.008	0.683 ±0.007	0.763 ±0.005	0.812 ±0.005	0.844 ±0.005	0.868 ±0.004	0.887 ±0.004	0.900 ±0.004	0.910 ±0.004	0.919 ±0.004
	random pw	0.614 ±0.011	0.807 ±0.009	0.886 ±0.007	0.917 ±0.006	0.928 ±0.005	0.933 ±0.004	0.936 ±0.004	0.937 ±0.004	0.938 ±0.004	0.939 ±0.004
	active pw	0.621 ±0.011	0.931 ±0.004	0.936 ±0.004	0.939 ±0.003	0.940 ±0.003	0.942 ±0.003	0.944 ±0.003	0.945 ±0.003	0.945 ±0.003	0.946 ±0.003
	random ppw	0.480 ±0.010	0.720 ±0.008	0.828 ±0.007	0.878 ±0.006	0.901 ±0.005	0.912 ±0.005	0.920 ±0.004	0.923 ±0.004	0.924 ±0.004	0.926 ±0.004
	active ppw	0.511 ±0.010	0.752 ±0.008	0.874 ±0.005	0.907 ±0.004	0.918 ±0.004	0.922 ±0.004	0.926 ±0.004	0.928 ±0.004	0.930 ±0.004	0.932 ±0.004
QRank	random cc	0.613 ±0,008	0.677 ±0,004	0.718 ±0,004	0.741 ±0,003	0.759 ±0,003	0.773 ±0,003	0.782 ±0,003	0.793 ±0,003	0.800 ±0,002	0.807 ±0,003
	active cc	0.599 ±0.007	0.687 ±0.004	0.741 ±0.003	0.764 ±0.003	0.782 ±0.002	0.798 ±0.002	0.810 ±0.002	0.819 ±0.002	0.828 ±0.002	0.837 ±0.002
	random pw	0.655 ±0.008	0.710 ±0.005	0.743 ±0.004	0.762 ±0.003	0.779 ±0.003	0.789 ±0.003	0.799 ±0.002	0.807 ±0.002	0.817 ±0.002	0.823 ±0.002
	active pw	0.649 ±0.007	0.737 ±0.004	0.785 ±0.003	0.812 ±0.002	0.823 ±0.002	0.831 ±0.002	0.838 ±0.002	0.843 ±0.002	0.847 ±0.001	0.849 ±0.001
	random ppw	0.600 ±0.008	0.673 ±0.005	0.708 ±0.003	0.726 ±0.003	0.741 ±0.003	0.756 ±0.003	0.766 ±0.003	0.775 ±0.003	0.785 ±0.002	0.792 ±0.002
	active ppw	0.615 ±0.009	0.703 ±0.004	0.754 ±0.003	0.785 ±0.002	0.804 ±0.002	0.816 ±0.002	0.828 ±0.002	0.832 ±0.002	0.836 ±0.002	0.841 ±0.002

[*] constraint classification (cc), standard pairwise (pw), probabilistic pairwise (ppw).

Table 7.1 *Experimental results of label ranking learning for the considered scenarios. The columns of this table correspond to the evaluation points, ranging from* 10 *initial examples to* 100 *examples with increments of* 10*. The generalization accuracy was estimated based on the Spearman rank correlation coefficient and the results are supplemented with standard error of the mean estimates.*

selected 90 examples using the different selection strategies. The rank correlation was evaluated after every 10 rounds and finally the results were averaged over all 100 datasets generated in each of the settings.

For $d \leq 5$, we included the computationally demanding original strategies based on the selection criteria stated in (7.54) and (7.58) in our experimental setup in order to provide a comparison to their approximate counterparts given in (7.56) and (7.57).

7.5.2 Experimental Results

Remember that the choice of kernel has not been optimized with respect to the different techniques. Therefore, we refrain from a quantitative comparison between different label ranking approaches and restrict our presentation to an evaluation of the random learning strategy and the corresponding active counterparts. Table 7.1 shows average rank correlations and corresponding standard errors of the mean for different numbers of labeled examples. The complete set of learning curves is depicted in Figures C.1-C.9 in Appendix C.

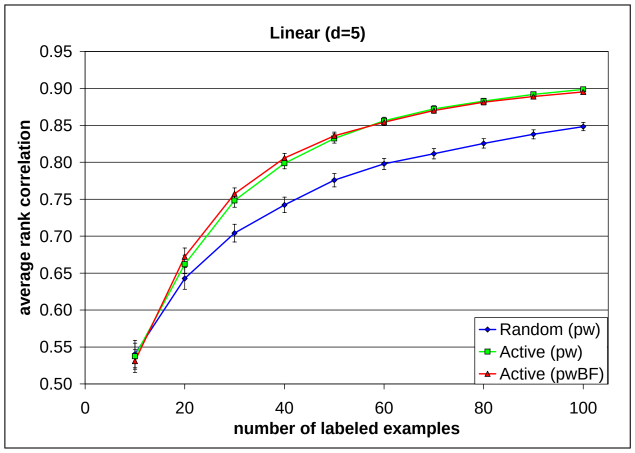
Linear: All active strategies consistently outperform their random counterparts for all choices of the number of alternatives. While there is a substantial increase in accuracy in the case of $d = 5$ alternatives, it becomes marginal in the case of constraint classification with the number of alternatives increasing. In contrast to this, for the pairwise decomposition techniques, the absolute rise in accuracy at fixed numbers of labeled examples increases with the number of alternatives.

MinMax: The active standard pairwise strategy achieves a level of accuracy for 20 labeled examples (for all choices of d) that is very close to that of random selection for 100 examples. For the probabilistic pairwise technique there is a substantial increase in accuracy for $d = 5, 10$, whereas for $d = 15, 20$ the active strategy is superior at the beginning while random selection slightly outperforms its active counterpart at the end. In the case of constraint classification, we observed a pattern similar to the former setting: Active learning consistently outperforms random learning. The gain in accuracy decreases with the number of alternatives increasing.

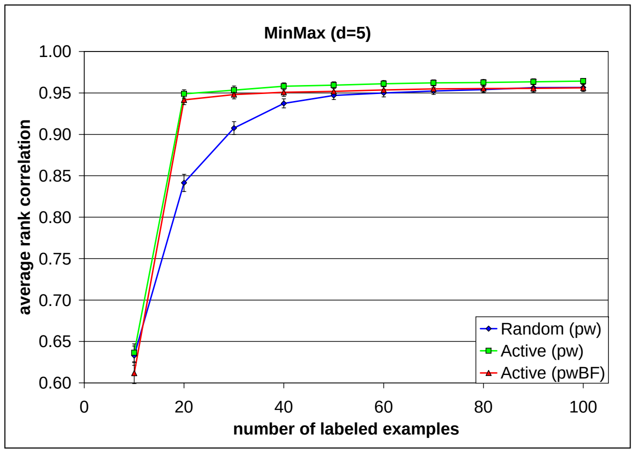
QRank: Active constraint classification learning consistently outperforms random learning. Compared to the former approach, both pairwise ranking strategies yield an even more substantial decrease of the labeling effort: The reduction is roughly two-fold, i.e., actively learning circa 50 examples yields the same estimated generalization accuracy as randomly learning 100 examples.

As in classification learning, the efficiency of active learning clearly depends on the given ranking problem. In our experiments, active learning based on the constraint classification approach consistently outperforms random learning. Moreover, active learning based on both standard and probabilistic pairwise decomposition yields a more substantial relative reduction of the labeling effort in most of the experiments. Furthermore, the computational effort for solving a ranking problem based on the constraint classification technique was more than two orders of magnitude higher than the computational effort for both pairwise ranking techniques.

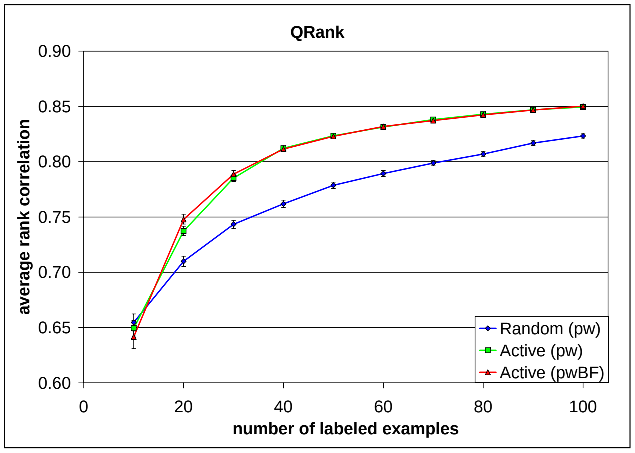
Moreover, the approximate pairwise selection strategies (7.56) and (7.57) generate experimental learning curves (see Figures 7.1 and 7.2) which are hardly distinguishable from their original counterparts. Thus, this approximation substantially decreased the computational complexity without a concomitant loss of accuracy in our experiments.



(a)

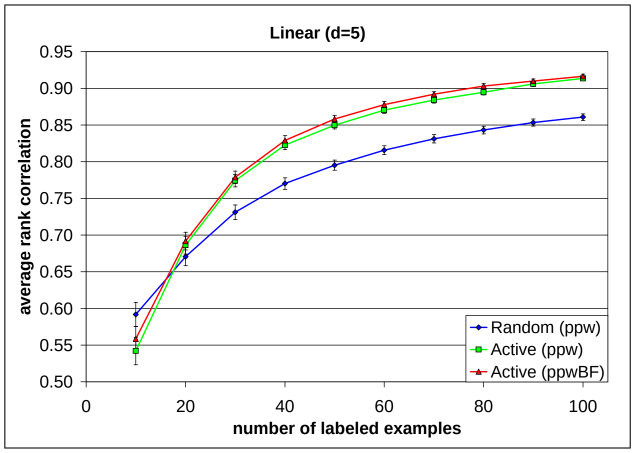


(b)

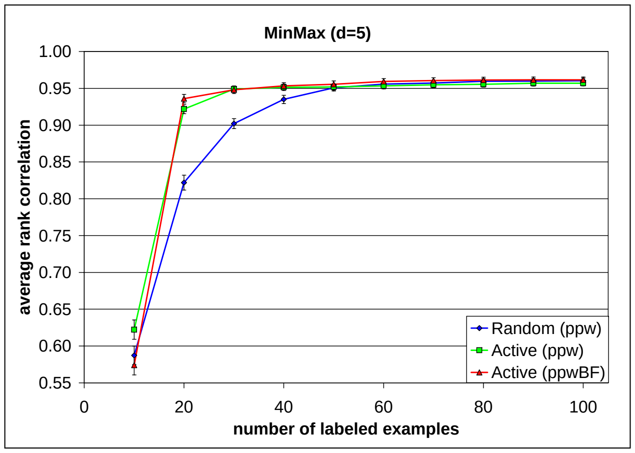


(c)

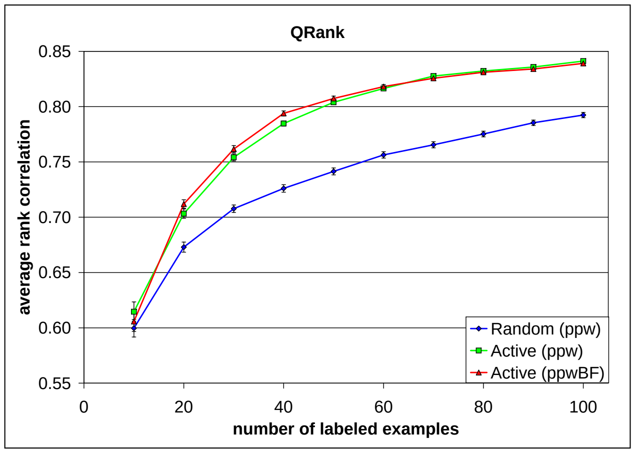
Figure 7.1 *The experimental learning curves corresponding to the approximate selection strategy for standard pairwise voting, which is referred to as Active(pw), are hardly distinguishable from the original strategy Active(pwBF).*



(a)



(b)



(c)

Figure 7.2 *The experimental learning curves corresponding to the approximate selection strategies for probabilistic pairwise voting, which is referred to as Active(ppw), are hardly distinguishable from the original strategy Active(ppwBF).*

7.6 Related Work

Beyond the label ranking model, alternative preference models which consider different learning scenarios have been examined. In the field of statistical decision theory and mathematical economics, the problem of learning a (preference) utility function (von Neumann and Morgenstern, 1944) of a given individual is referred to as preference elicitation. While in preference elicitation the objective is to assign real-valued utility scores to examples according to a single user's preferences, label ranking functions assign a finite preference order to each example. More generally, in the former model, examples correspond to decision alternatives, whereas in the latter model a finite set of alternatives is a priori fixed and examples correspond to different individuals.

Learning a utility function can be considered as a regression problem. In the field of kernel methods, Crammer and Singer (2002) introduced an online algorithm for learning a utility function on an ordinal scale which is used to order a set of unseen examples with respect to their ordinal utility scores. This problem is also referred to as ordinal regression and was investigated in the batch setting by Herbrich et al. (2000). Joachims (2002) proposed a large margin algorithm which utilizes clickthrough data (weblog data) to order documents retrieved from multiple search engines with respect to their relevance. A set of documents is ordered according to the signed distances to a hyperplane which is determined such that it (approximately) minimizes the number of discordant pairwise preferences (inversions) on the training set.

Beyond the class of regression-based preference models, Cohen et al. (1999) proposed a two-stage approach to preference learning: In stage one, the algorithm learns a probabilistic preference function on pairs of given examples which in stage two is used to order a given set of unseen examples in some (approximately) optimal sense. More recently, Haddawy et al. (2004) examined the question of how to represent various assumptions about pairwise preferences in an artificial neural network architecture to guide the elicitation process.

7.7 Discussion

We introduced novel extensions of pool-based active learning to label ranking learning based on both the constraint classification technique and pairwise decomposition of preferences. Experimental results clearly indicate that pool-based active learning can significantly reduce the labeling effort in ranking learning.

While the labeling process already constitutes a major issue in classification learning, it becomes an even more substantial matter of relevance in the more complex target domain of rankings. Therefore, this category of learning problems in particular can benefit from the active learning approach.

Label ranking learning has received increasing attention in the machine learning community in recent years. It is an area of ongoing research and many interesting questions are still unanswered. More sophisticated prediction schemes for pairwise ranking are to be investigated, among others. A natural means for the prediction of a ranking is to maximize the sum over pairwise preference functions. However, Cohen et al. (1999) showed that it is NP-hard to solve this problem. We conducted limited experimental analysis on a prediction technique adapted from (Cohen et al., 1999) which exhibits the appealing property that it is guaranteed to approximate the NP-hard problem of finding a ranking which maximizes the sum over pairwise preference functions up to a factor of 2. In spite of this theoretical property, this modification did not improve generalization accuracy in comparison to standard pairwise voting.

Many companies collect customer preference data which fit the label ranking model in order to evaluate existing products and respond to new trends in the market, among others. Unfortunately, to our best knowledge, datasets of sufficient size for benchmark testing are not publicly available so far. Therefore, experimental analysis in label ranking learning is restricted to synthetical data, despite its practical relevance. However, it would be extremely useful to evaluate the different representation techniques and the proposed selection strategies on real-world problems.

Moreover, the assumption that total preference orders are provided for training a ranking function might not match typical characteristics of learning problems in practice. Generalizing the constraint classification and the pairwise ranking techniques to partial orders and weak orders over the predefined set of alternatives seems to be a promising direction of further research.

8 Convergence Analysis

Overview

This chapter embarks on a more thorough theoretical analysis of volume-based selection strategies in pool-based active learning. We derive a convergence theorem for the SIMPLE strategy in the case of the maximum radius ball approximation of the center of mass and present a survey of related results.

8.1 Convergence Analysis

It was shown in Section 2.4 that the center of the largest inscribable ball in version space corresponds to the weight vector of the maximum margin classifier provided that the training examples in the kernel-induced feature space are normalized to unit length.[1] Furthermore, if the version space is regularly shaped, then it is a reasonable approximation of the center of mass of the version space (Herbrich et al., 2001). However, if the version space is elongated in some direction, this approximation is likely to become inaccurate and it is not clear whether the SIMPLE strategy, which selects examples with minimum distance to the maximum margin hyperplane, effectively reduces the volume of the version space (see Figure 8.1).

In spite of the potential failure to accurately approximate the center of mass, we will prove in this section that in the finite-dimensional real vector space $\mathbb{R}^N$ the volume of the version space converges to zero under the assumption that we are able to sequentially augment the set of labeled examples by new examples with zero distance to the maximum margin hyperplanes.

As a prerequisite for the proof of our main theorem, we need the following geometric lemma, which provides a lower bound on the radius of a ball that can be placed in a convex body which is contained in the N-dimensional unit ball as a function of its volume:

Lemma 8.1 (Fine et al., 2002)
Let C be a convex body contained in the N-dimensional unit ball $\mathcal{B}_1^{(N)}(0)$ $(N > 1)$. Let ∂C be the surface of this body and assume that ∂C is smooth[2] at all points except

1. It suffices to assume that examples are normalized to a fixed length. For the sake of simplicity, we assume a unit normalization.
2. Intuitively, the notion of smoothness of the boundary of a convex set refers to the property of having sharp corners. For a formal definition see (Kannan and Nolte, 1998).

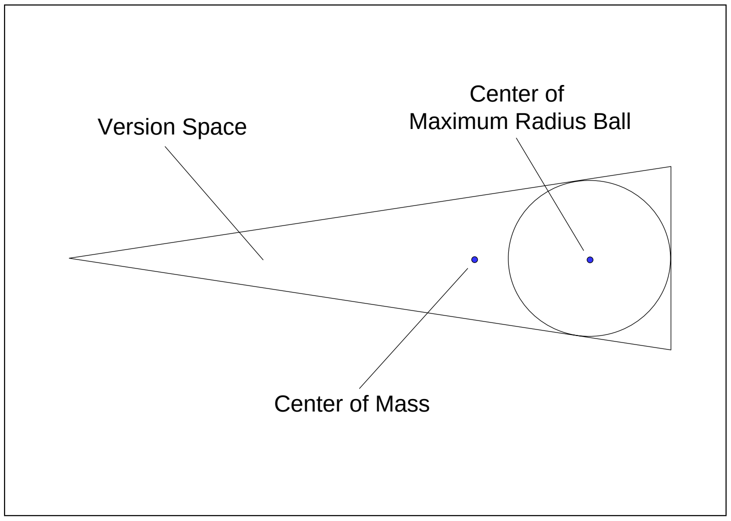


Figure 8.1 *This schematic geometric figure illustrates that the center of the largest inscribable ball does not accurately approximate the center of mass if the version space is elongated in some direction.*

for a set of zero measure. Then, there exists a ball of radius r contained in C such that

$$r \geq \frac{\Gamma\left(\frac{N}{2}+1\right)}{N\pi^{\frac{N}{2}}}\,\mathrm{vol}(C) \tag{8.1}$$

where the Γ-function is defined as

$$\Gamma : \mathbb{R}_{>0} \to \mathbb{R} \tag{8.2}$$

$$x \mapsto \int_0^\infty t^{x-1}e^{-t}\,dt. \tag{8.3}$$

Remark 8.2

The smoothness requirement holds for all the convex bodies that will be addressed in this section (see (Fine et al., 2002) for details).

Remark 8.3

Note that the following property holds for the Γ-function:

$$\Gamma\left(N+\tfrac{1}{2}\right) = \frac{(2N)!\pi^{1/2}}{N!2^{2N}} \quad \textit{for } N \in \mathbb{N}_0. \tag{8.4}$$

We recapitulate volume formula for N-dimensional balls and cones (Sommerville, 1958):

Lemma 8.4 Volume of N-dimensional Ball
The volume of an N-dimensional ball $\mathcal{B}_r^{(N)}$ with radius $r > 0$ is given by [3]

$$\operatorname{vol}\left(\mathcal{B}_r^{(N)}\right) = \frac{r^N \, \pi^{\frac{N}{2}}}{N \, \Gamma\left(\frac{N}{2}+1\right)}. \tag{8.5}$$

Lemma 8.5 Volume of N-dimensional Cone
The volume of an N-dimensional cone $\mathcal{C}_{h,r}^{(N)}$ of height h and basis radius $r > 0$, i.e., the basis is an $(N-1)$-dimensional ball $\mathcal{B}_r^{(N-1)}$ of radius $r > 0$, is given by

$$\operatorname{vol}\left(\mathcal{C}_{h,r}^{(N)}\right) = \frac{h}{N} \operatorname{vol}\left(\mathcal{B}_r^{(N-1)}\right) = \frac{h \, r^{N-1} \, \pi^{\frac{N-1}{2}}}{N \, \Gamma\left(\frac{N+1}{2}\right)}. \tag{8.6}$$

Endowed with these prerequisites, we can now proceed to our main theorem:

Theorem 8.6 Convergence of Volume Sequence
Suppose we are given a fixed feature map $\phi : \mathcal{X} \to \mathcal{F} = \mathbb{R}^N$ (with $N > 1$), which embeds patterns into the real Euclidean feature space $\mathcal{F}$ (endowed with the canonical dot product), and the binary target space $\mathcal{Y} = \{-1, +1\}$. Assume that all examples are normalized to unit length in feature space. Let us define $\mathcal{L}_0 \overset{def}{=} \emptyset$ and assume that $\mathcal{L}_i = \mathcal{L}_{i-1} \cup \{(x, y)\}$ for $i \geq 1$ where $\langle \mathbf{w}_i^{(svm)}, \phi(x) \rangle = 0$ with $\mathbf{w}_i^{(svm)}$ denoting the maximum margin classifier corresponding to $\mathcal{L}_i$. Denote the version space of linear classifiers consistent with the set of labeled examples $\mathcal{L}_i$ by

$$\mathcal{V}_i \overset{def}{=} \{\mathbf{w} \in \mathcal{F} \mid y\langle \mathbf{w}, \phi(x) \rangle > 0 \;\; \text{for all} \;\; (x, y) \in \mathcal{L}_i, \; \|\mathbf{w}\| \leq 1\} \quad \text{for } i \geq 0. \tag{8.7}$$

Then, the following property holds for the sequence of volumes:

$$\lim_{i \to \infty} \operatorname{vol}(\mathcal{V}_i) = 0. \tag{8.8}$$

Proof We assumed that all examples are normalized to unit length in feature space. Therefore, the appropriately scaled weight vector $\mathbf{w}_i^{(\text{svm})}$ of the maximum margin classifier for $\mathcal{L}_i$ corresponds to the center of a ball $\mathcal{B}_{\gamma_i}^{(N)}$ in the compactified version space $\overline{\mathcal{V}_i}$ with maximum radius γ_i.[4] Note that the compactification does not increase the volume of the version space.

For $i \geq 1$ let us consider an N-dimensional cone $\mathcal{C}_{1-\gamma_i,\gamma_i}^{(N)}$ of height $1 - \gamma_i$ and basis radius γ_i. Assume that the cone is positioned such that its apex coincidences with the origin and its basis is contained in the maximum radius ball $\mathcal{B}_{\gamma_i}^{(N)}$. Then, any

3. Since the Lebesgue measure is translation-invariant, specifying the center of the ball is not necessary. In the following, we will omit this parameter if not relevant.

4. While in general the ball yielding maximum radius inside a compact convex set is not necessarily unique, the maximum radius itself is unique. However, in the special case of the (compactified) version space, the maximum radius ball is unique as a direct consequence of the existence of a unique maximum margin classifier.

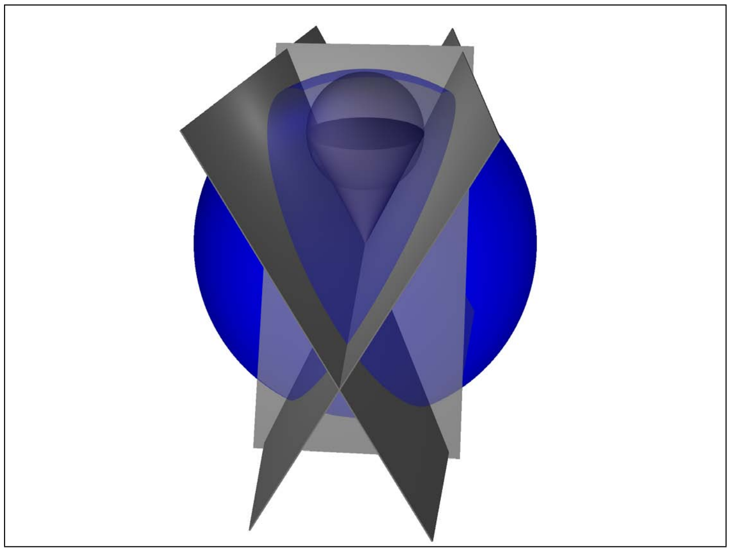


Figure 8.2 *The center of mass of the version space is approximated by the center of a ball $\mathcal{B}_{\gamma}^{(N)}$ inscribable in version space with maximum radius. If we position a cone of height $1-\gamma$ and basis radius γ such that the apex coincidences with the origin and its basis is contained in the ball $\mathcal{B}_{\gamma}^{(N)}$, then the entire cone is a subset of the version space. Note that in this figure, we have omitted a fourth hyperplane limiting the version space from the front direction to enable visibility.*

$\mathbf{w} \in \mathcal{C}_{1-\gamma_i,\gamma_i}^{(N)}$ can be rewritten as $\mathbf{w} = \lambda \mathbf{w}'$ where $\mathbf{w}' \in \mathcal{B}_{\gamma_i}^{(N)}$ and $\lambda \geq 0$. Hence,

$$y\langle \mathbf{w}, \phi(x)\rangle = \lambda y \langle \mathbf{w}', \phi(x)\rangle \geq 0 \quad \text{for all } (x,y) \in \mathcal{L}_i \tag{8.9}$$

because $\mathbf{w}' \in \overline{\mathcal{V}_i}$. Since this property holds for all $\mathbf{w} \in \mathcal{C}_{1-\gamma_m,\gamma_m}^{(N)}$, we conclude that the entire cone is a subset of the compactified version space, i.e., $\mathcal{C}_{1-\gamma_m,\gamma_m}^{(N)} \subset \overline{\mathcal{V}_i}$ (see Figure 8.2 for a geometric illustration).
From Lemma 8.5, it follows for the volume of the cone that

$$\operatorname{vol}\left(\mathcal{C}_{1-\gamma_i,\gamma_i}^{(N)}\right) = \frac{(1-\gamma_i)\,\gamma_i^{N-1}\,\pi^{\frac{N-1}{2}}}{N\,\Gamma\left(\frac{N+1}{2}\right)}. \tag{8.10}$$

It can be readily verified that $\operatorname{vol}\left(\mathcal{C}_{1-\gamma_i,\gamma_i}^{(N)}\right)$ is monotonously increasing in γ_i for $0 \leq \gamma_i \leq \frac{1}{2}$. Since for $i \geq 1$ the version space is reduced to at least half the unit ball, we conclude that $\gamma_i \leq \frac{1}{2}$ for $i \geq 1$.
Lemma 8.1 provides a lower bound on the radius γ_i of a ball inscribable in $\overline{\mathcal{V}_i}$ in terms

of its volume:

$$\gamma_i \geq \frac{\Gamma\left(\frac{N}{2}+1\right)}{N\pi^{\frac{N}{2}}} \operatorname{vol}(\overline{\mathcal{V}_i}) = \frac{\Gamma\left(\frac{N}{2}+1\right)}{N\pi^{\frac{N}{2}}} \operatorname{vol}(\mathcal{V}_i). \tag{8.11}$$

$$\tag{8.12}$$

Thus, the following bound holds for the volume of the cone:

$$\operatorname{vol}\left(\mathcal{C}^{(N)}_{1-\gamma_i,\gamma_i}\right) = \frac{(1-\gamma_i)\,\gamma_i^{N-1}\,\pi^{\frac{N-1}{2}}}{N\,\Gamma\left(\frac{N+1}{2}\right)} \tag{8.13}$$

$$\geq \frac{\left(1-\frac{\Gamma\left(\frac{N}{2}+1\right)}{N\pi^{\frac{N}{2}}}\operatorname{vol}(\mathcal{V}_i)\right)\left(\frac{\Gamma\left(\frac{N}{2}+1\right)}{N\pi^{\frac{N}{2}}}\operatorname{vol}(\mathcal{V}_i)\right)^{N-1}\pi^{\frac{N-1}{2}}}{N\,\Gamma\left(\frac{N+1}{2}\right)} \tag{8.14}$$

$$= c_1^{(N)}\left(1-c_2^{(N)}\operatorname{vol}(\mathcal{V}_i)\right)\operatorname{vol}(\mathcal{V}_i)^{N-1} \tag{8.15}$$

where the constants $c_1^{(N)}$ and $c_2^{(N)}$ are defined as

$$c_1^{(N)} \overset{\text{def}}{=} \frac{\left(\frac{\Gamma(\frac{N}{2}+1)}{N\pi^{\frac{N}{2}}}\right)^{N-1}\pi^{\frac{N-1}{2}}}{N\,\Gamma(\frac{N+1}{2})} = \frac{\Gamma(\frac{N}{2}+1)^{N-1}}{\Gamma(\frac{N+1}{2})\,N^N\,\pi^{\frac{(N-1)^2}{2}}} \quad \text{and} \tag{8.16}$$

$$c_2^{(N)} \overset{\text{def}}{=} \frac{\Gamma(\frac{N}{2}+1)}{N\pi^{\frac{N}{2}}}. \tag{8.17}$$

If we augment the set of labeled examples $\mathcal{L}_i$ by a new example (x, y) whose image $\phi(x)$ is perpendicular to $\mathbf{w}_i^{(\text{svm})}$ in the feature space $\mathcal{F}$, i.e., $\langle \mathbf{w}_i^{(\text{svm})}, \phi(x)\rangle = 0$, the corresponding hyperplane with normal vector $\phi(x)$ bisects the cone into two parts of equal volume. Therefore, the volume of the version space is reduced by at least half the volume of the cone. Thus, using (8.15), we can bound the volume of the version space $\mathcal{V}_{i+1}$ in terms of the volume of the previous version space $\mathcal{V}_i$ by

$$\operatorname{vol}(\mathcal{V}_{i+1}) = \operatorname{vol}(\overline{\mathcal{V}_{i+1}}) \tag{8.18}$$

$$\leq \operatorname{vol}(\mathcal{V}_i) - \tfrac{1}{2}\operatorname{vol}\left(\mathcal{C}^{(N)}_{1-\gamma_i,\gamma_i}\right) \tag{8.19}$$

$$\leq \operatorname{vol}(\mathcal{V}_i) - \tfrac{1}{2}\,c_1^{(N)}\left(1-c_2^{(N)}\operatorname{vol}(\mathcal{V}_i)\right)\operatorname{vol}(\mathcal{V}_i)^{N-1} \tag{8.20}$$

$$= \operatorname{vol}(\mathcal{V}_i)\left(1-\tfrac{1}{2}\,c_1^{(N)}\left(1-c_2^{(N)}\operatorname{vol}(\mathcal{V}_i)\right)\operatorname{vol}(\mathcal{V}_i)^{N-2}\right). \tag{8.21}$$

In the following, we conduct an analysis of the sequence of upper bounds on the volumes of the version spaces. Therefore, we have to investigate the sequence $(a_i)_{i\geq 0}$ defined by the following recurrence:

$$a_0 \overset{\text{def}}{=} \operatorname{vol}\left(\mathcal{B}_1^{(N)}\right), \tag{8.22}$$

$$a_i \overset{\text{def}}{=} a_{i-1}\left(1-\tfrac{1}{2}\,c_1^{(N)}(1-c_2^{(N)}a_{i-1})a_{i-1}^{N-2}\right) \quad \text{for } i \geq 1 \tag{8.23}$$

with $N > 1$ being fixed.

The sequence $(a_i)_{i\geq 0}$ satisfies the following properties:

1. $(a_i)_{i\geq 0}$ is non-increasing.
2. $a_i \geq 0$ for all $i \geq 0$.

Property 1. Trivial.

Property 2. The elements of the sequence have been derived as upper bounds of volumes and therefore are nonnegative.

From 1. and 2. we conclude that the sequence converges. Let $u \overset{\text{def}}{=} \lim_{i\to\infty} a_i$ denote its limit. Then by the laws of limits and the continuity of the power function the following holds:

$$u = \lim_{i\to\infty} a_i = \lim_{i\to\infty}\left(a_{i-1}\left(1-\frac{1}{2}c_1^{(N)}(1-c_2^{(N)}a_{i-1})a_{i-1}^{N-2}\right)\right) \tag{8.24}$$

$$= u\underbrace{\left(1-\frac{1}{2}c_1^{(N)}(1-c_2^{(N)}u)u^{N-2}\right)}_{\substack{\in(0,1)\\ \text{for } u\in(0,\mathrm{vol}(\mathcal{B}_1^{(N)})]}} \tag{8.25}$$

$$\Rightarrow \quad u = 0. \tag{8.26}$$

Since the elements of the sequence $(a_i)_{i\geq 0}$ are upper bounds of the nonnegative elements of the sequence $(\mathrm{vol}(\mathcal{V}_i))_{i\geq 0}$, the proof is completed. ■

Remark 8.7

According to Theorem 8.6, the volume of the set of weight vectors corresponding to consistent classifiers converges to zero under the assumption that the set of labeled examples is sequentially augmented by new examples with zero distance to the maximum margin hyperplanes. Hence, in a deterministic model, a necessary condition for this assumption to hold is that the set of classifiers generating the data is limited to a set of zero measure in feature space.

Remark 8.8

The arguably more straightforward approach to consider the volume reduction by the maximum radius ball, instead of the constructed cone, does not lead to a convergence proof for $N = 2$. Moreover, the above-defined sequence of upper bounds on the volumes provides the basis for an extended analysis of the rate of convergence. However, since the given recurrence falls into the class of nonlinear recurrence equations, which is still an area of ongoing research, further analysis seems to be quite intricate. In general, there are no closed form solutions to these recurrences. In particular, Mandelbrot and Julia sets are obtained from the chaotic behavior of quadratic recurrence equations and, moreover, logistic difference equations of the form $a_i = c\, a_{i-1}(1-a_{i-1})$ are studied in chaos theory.

For the remainder of this section, we provide a survey of related results for the SIMPLE selection strategy in the case of the exact center of mass and more accurate estimates of the center of mass of the version space. Let us proceed to a classical result from the theory of convex sets for the exact center of mass:

Theorem 8.9 (Grünbaum, 1960; Bertsimas and Vempala, 2002)
Any halfspace which contains the center of mass of a bounded convex set $C \subset \mathbb{R}^N$ contains at least $\frac{1}{e}$ of the overall volume of C.

Corollary 8.10
Suppose we are given a fixed feature map $\phi : \mathcal{X} \to \mathcal{F} = \mathbb{R}^N$ (with $N > 1$) and the binary target space $\mathcal{Y} = \{-1, +1\}$. Let us define $\mathcal{L}_0 \overset{def}{=} \emptyset$ and assume that $\mathcal{L}_{i+1} = \mathcal{L}_i \cup \{(x, y)\}$ for $i \geq 0$ where $\langle \mathbf{w}_i^{(center)}, \phi(x) \rangle = 0$ with $\mathbf{w}_i^{(center)}$ denoting the center of mass of $\mathcal{L}_i$. Then, the following property holds for the sequence of volumes:

$$\left(\frac{1}{e}\right)^i \mathrm{vol}(\mathcal{B}_1^{(N)}) \leq \mathrm{vol}(\mathcal{V}_i) \leq \left(1 - \frac{1}{e}\right)^i \mathrm{vol}(\mathcal{B}_1^{(N)}) \quad \textit{for } i \geq 0. \tag{8.27}$$

Proof Note that if any halfspace passing through the center of mass of a convex set would contain more than a fraction of $1 - \frac{1}{e}$ of the overall volume, then the union of the complementary halfspace with the bounding hyperplane would contain less than a fraction of $\frac{1}{e}$, which leads to a contradiction of Theorem 8.9. The lower bound follows analogously. ■

Thus, if we assume that the set of labeled examples is sequentially augmented by examples which correspond to restricting hyperplanes passing through the exact center of mass of the current version space, then, independently of the actual class label, the volume of the version space is reduced exponentially in terms of the number of labeled examples.

Since calculating the center of mass of high dimensional convex sets is computationally very demanding but for the simplest cases, we will consider an approximate method in the remainder of this section. In the field of kernel machines, algorithms for the approximation of the center of mass of the version space are referred to as Bayes point machines. We have already discussed the class of kernel billiard algorithms which approximate the center of mass based on random walks in the version space (Herbrich et al., 2001; Ruján and Marchand, 2000; Minka, 2001).[5] However, bounds on the accuracy of this approximation have not been established so far. We consider a direct sampling-based method in the following.

Let $\mathbf{w}_1, \ldots, \mathbf{w}_M$ denote M samples drawn independently from a uniform distribution over the version space. We approximate the center of mass by averaging over these samples:

$$\mathbf{w}^{(\mathrm{bpm})} \overset{\mathrm{def}}{=} \frac{1}{M} \sum_{i=1}^{M} \mathbf{w}_i. \tag{8.28}$$

Herbrich and Graepel (2001) considered a modification of this approach for extending Bayes point machines to large datasets which (pseudo-)samples from the version space by running a kernelized perceptron algorithm on randomly generated permutations of the training data.

5. See Sections 2.5 and 4.2 for a more detailed treatment.

We state an adapted version of a theorem given in (Bertsimas and Vempala, 2002) to provide a convergence analysis for the sampling approximation method:

Theorem 8.11 (Bertsimas and Vempala, 2002)
Fix $0 < \delta < 1$ and $\varepsilon > 0$. If we approximate the center of mass of a bounded N-dimensional convex set C by averaging over $M \overset{\text{def}}{=} \frac{N}{\delta\varepsilon^2}$ samples independently and uniformly drawn from this set, then with probability of at least $1 - \delta$ a fraction of at least $\frac{1}{e} - \varepsilon$ of the entire volume is located on either side of any hyperplane passing through this approximation.

Proof See Appendix A. ■

Corollary 8.12
Fix $0 < \delta < 1$. If we approximate the center of mass of a bounded N-dimensional convex set C by averaging over $M \overset{\text{def}}{=} \frac{N}{\delta\left(\frac{1}{e}-\frac{1}{3}\right)^2}$ samples independently and uniformly drawn from this set, with probability of at least $1 - \delta$ a fraction of at least $\frac{1}{3}$ of the entire volume is located on either side of any hyperplane passing through this approximation.

Corollary 8.13
Suppose we are given a fixed feature map $\phi : \mathcal{X} \to \mathcal{F} = \mathbb{R}^N$ (with $N > 1$) and the binary target space $\mathcal{Y} = \{-1, +1\}$. Fix $0 < \delta < 1$ and the number of samples $M \overset{\text{def}}{=} \frac{N}{\delta\left(\frac{1}{e}-\frac{1}{3}\right)^2}$. Let us define $\mathcal{L}_0 \overset{\text{def}}{=} \emptyset$ and assume that $\mathcal{L}_{i+1} = \mathcal{L}_i \cup \{(x, y)\}$ for $i \geq 0$ where $\langle \mathbf{w}_i^{(bpm)}, \phi(x)\rangle = 0$ with $\mathbf{w}_i^{(bpm)}$ approximating the center of mass of the version space $\mathcal{V}_i$ corresponding to $\mathcal{L}_i$ by averaging over M independently and uniformly drawn samples. Then, the following property holds for the sequence of expected volumes:

$$\mathrm{E}(\mathrm{vol}(\mathcal{V}_i)) \leq \mathrm{vol}(\mathcal{B}_1^{(N)}) \left(\frac{2}{3}\right)^{(1-\delta)i} \qquad \text{for } i \geq 0. \tag{8.29}$$

Proof Consider a sequence $(b_i)_{i\in\mathbb{N}}$ corresponding to the sequence of the volumes of the version space $(\mathrm{vol}(\mathcal{V}_i))_{i\in\mathbb{N}}$ with $b_i = 1$ representing a successful volume reduction by a fraction of at least $\frac{1}{3}$ and $b_i = 0$ otherwise. According to Corollary 8.12, we can lower bound the expected accumulated number of successes for the subsequence $b_1, \ldots, b_i$ of length i by a binomial distribution with parameters $(i, 1 - \delta)$ which has expectation $(1 - \delta)i$. ■

According to Corollary 8.13, the expected resulting volume reduction of the Simple selection strategy is exponential in the number of labeled training examples when approximating the center of mass using $\mathcal{O}(N)$ samples from the version space.

As a consequence of Theorem 2.5 on the construction of a data-dependent finite-dimensional kernel feature space and map for a given set of examples, Corollaries 8.10 and 8.13 hold for $N = n$ and $i = 0, \ldots, n$ where n denotes the number of initially unlabeled examples.

In the present chapter, we analyzed the Simple selection strategy under the assumption that in every step there exists an unlabeled example which is orthogonal

Summary

to the (approximate) center of mass of the current version space. We proved a novel convergence theorem which states that the SIMPLE selection strategy in the case of the maximum radius ball approximation is guaranteed to asymptotically reduce the volume of the version space to zero. Moreover, we presented a survey of related results showing that in the case of the exact center of mass the volume reduction is exponential and, in the case of the sampling-based approximation, the expected volume reduction is exponential in terms of the number of labeled examples.

9 Conclusion

The increasing trend from rather restricted to more flexible learning models, which utilize the entire spectrum of available sources of information, comprises various directions of research in machine learning. Besides the classical paradigm of *learning by (labeled) examples*, multiple pieces of information can be incorporated in order to expedite the learning process. Prior knowledge about learning problems in the form of entire regions of the input space assigned to a particular target class may be obtained from experts on the considered problem domain and can be employed to reduce the number of additional training examples necessary to attain a certain level of accuracy (Fung et al., 2003). Moreover, task-specific prior knowledge in the form of invariance properties of prediction functions, such as translation invariance in optical character recognition, can be incorporated into support vector learning by generating virtual examples or by modifying the kernel function correspondingly (Schölkopf, 1997, Chapter 4).

In addition to labeled examples and explicit prior knowledge, a valuable source of information is unlabeled data. Being situated between two extreme settings in the spectrum of machine learning, viz. supervised and unsupervised learning, semi-supervised learning assumes both a labeled and an unlabeled set of examples being submitted to the learner (see Chapter 3). The underlying objective in semi-supervised learning is to exploit the availability of unlabeled examples in order to improve the prediction function by utilizing both the set of labeled and the set of unlabeled examples.

While semi-supervised learning considers a *static* setting where we are given fixed sets of labeled and unlabeled examples, the distinguishing feature of pool-based active learning is a *dynamic* sequential process in which the learning algorithm is granted access to a set of unlabeled examples and provided with the ability to determine the order of assigning target objects with the objective of attaining a high level of accuracy without requesting the complete set of corresponding target objects. As a consequence of the labeling process, the course of action in pool-based active learning generates an increasing set of labeled examples and a decreasing set of unlabeled ones.

Let us proceed to an overview of the main contributions of the present thesis to the field of pool-based active learning with kernel machines, followed by a more extensive discussion which is supplemented by comments on interesting areas of further research.

- *Improvement of Efficiency:* In the context of binary classification learning, we proposed a novel strategy for the selection of batches of multiple examples. As the fundamental component of our approach, we incorporated a measure of diversity to provide a more efficient strategy in terms of the number of labeled examples necessary to attain a particular level of classification accuracy. Moreover, we suggested modified multiclass selection strategies for different binary decomposition methods which yield substantial improvements over previous research.
- *Generalization:* Label ranking forms a category of preference learning problems which has not been investigated in active learning research previously, despite the fact that the labeling effort is an even more essential matter of relevance here than in classification learning. Employing the constraint classification and the pairwise ranking techniques, we proposed generalizations of pool-based active learning and demonstrated a substantial improvement of the learning progress.
- *Theoretical Foundations:* We investigated a linear learning setting to analyze theoretical drawbacks of volume-based selection strategies and showed that the minimization of the volume of the version space can be viewed as a *necessary* precondition for the minimization of the proposed improved selection criterion. Moreover, we proved a novel convergence theorem for the volume-based SIMPLE selection strategy in the case of the maximum radius ball approximation.

In the present thesis, we studied pool-based active learning in the context of kernel machines both from a theoretical perspective and with respect to the application of this framework in practice. Volume-based selection strategies were analyzed in a particular linear learning setting where certain sequences of examples reveal potential shortcomings of the underlying criteria. We developed an improved selection strategy for this setting and suggested a sophisticated two-layered architecture with a view to the reduction of the computational complexity. From a theoretical point of view, the minimization of the volume of the version space is a *necessary* precondition for the minimization of the improved selection criterion. Moreover, further optimization in terms of algorithmic efficiency and more computational power is necessary to conduct experiments on a realistic scale in the future and to determine whether the proposed selection strategy can be employed as a general-purpose selection strategy.

As a generalization of the single step setting, we examined the selection of batches of multiple examples in each iteration. Apart from a reduction of computational demands, this setting is of particular relevance in domains which provide a parallel labeling component for the simultaneous determination of requested target objects, such as multiple automatized experiments in the process of chemical drug discovery. Certain aspects of the proposed selection strategy, such as the dynamic adaptation of control parameters and the combination of the underlying subcriteria, raise interesting questions and necessitate further investigation. Our substantial contribution is the introduction of a notion of diversity defined on the set of selected patterns in general. From a global perspective, incorporating a measure of diversity can be viewed as a means of controlling the level of exploration and exploitation, i.e., adjusting the objectives of covering different regions of the feature space and enforcing the knowledge about local areas of particular interest.

Beyond the binary classification setting, we studied both multiclass and label ranking learning in the pool-based active learning framework. As the decomposition of multiclass classification problems into binary ones is still an area of ongoing research, active learning will profit from advances in the theoretical foundations of decomposition techniques. In label ranking learning, the effort of labeling input patterns is a particular matter of relevance as a consequence of the more complex target space and the associated increased costs of providing target objects. Therefore, label ranking is a fertile area of application for active learning and profits particularly from the reduction of the labeling effort. Moreover, we provided a detailed discussion of the efficient kernelization of the constraint classification approach to label ranking. Generalizing the constraint classification and the pairwise ranking technique to partial orders and weak orders over the predefined set of alternatives is a promising direction of further research which allows us to extend the area of application of this setting. Adapting the proposed selection strategies to these generalized settings seems to be feasible by means of a similar line of reasoning as in the original setting.

The theoretical analysis of volume-based selection strategies in binary classification learning, in particular our novel convergence theorem for the SIMPLE selection strategy in the case of the maximum radius ball approximation, and the overall presentation of related results were based on the convex version space model. We expect the strong connection to research on the theory of convex sets to form the basis of the process of extending the theoretical foundations of this category of approaches.

The principal underlying cost model for providing target objects considered in active learning research implicitly assumes constant and uniform costs, independently of the given input pattern. This basic model can be generalized in many directions to incorporate more complex dependencies, such as class- and pattern-specific or time-varying cost notions. Moreover, in certain areas of application, it may be suitable to consider an extended cost model which includes the process of sampling patterns. As a consequence, active learning strategies have to be adapted to these more sophisticated cost models. Furthermore, even in the basic classification model, a variety of alternative evaluation measures besides classification accuracy, such as the area under the ROC curve (AUC), receive increasing attention in machine learning research and necessitate the development of specialized active learning strategies.

A fruitful, yet challenging direction of further research is the development of techniques of conducting model and hyperparameter selection in active learning, respectively. Standard techniques in the batch learning setting, such as leave-one-out and cross-validation, lack a theoretical justification in active learning and there is, moreover, experimental evidence of failure in practice (Baram et al., 2004). In certain domains comprehensive prior knowledge about the appropriate choice of kernel functions and the expected level of noise in the data is available. However, the dependence on prior knowledge and rather coarse unsupervised metrics in selecting fixed hypothesis spaces may limit the application of active learning in other domains. The development of adequate semi-supervised metrics is necessary in order to dynamically adapt and optimize models during the learning process as the proportion between labeled and unlabeled examples changes. Beyond research on general techniques in the context of pool-based active learning with kernel machines, an interleaving of active construction of patterns and model selection for the special category of trigonometric polynomial models was studied in (Sugiyama and Hidemitsu, 2003).

The preceding paragraphs have demonstrated that, while fundamental steps have been taken, a variety of interesting questions yield promising directions for further research in active learning.

III APPENDIX

Overview

The appendix contains both supplementary theoretical material and a comprehensive set of experimental learning curves. More precisely, Appendix A states a proof of Theorem 8.11 for completeness, while Appendices B and C provide the complete set of learning curves for our experimental evaluation of multiclass active learning and active learning in the case of label ranking, respectively. The experimental settings include both benchmark datasets, which are publicly available from the UCI repository of machine learning databases (Blake and Merz, 1998) and the Statlog collection (Michie et al., 1994), and artificial data.

A Bound for Sampling Method

For completeness, we state a proof of Theorem 8.11 which has been adapted from (Bertsimas and Vempala, 2002) and closely follows the herein given line of reasoning.

Definition A.1
A bounded convex set $C \subset \mathbb{R}^N$ is in isotropic position if

1. *its center of mass is the origin, i.e., for a random variable* W *which is uniformly distributed over* C*, it holds that*

$$\mathsf{E}(\mathsf{W}) = 0, \tag{A.1}$$

2. *its covariance matrix is the identity matrix:*

$$\mathsf{E}(\mathsf{W}\mathsf{W}^\top) = \mathsf{I}. \tag{A.2}$$

The former requirement can be substituted with the following equivalent requirement:
2'. *for any unit vector* $\mathbf{v} \in \mathbb{R}^N$*, it holds that*

$$\frac{1}{\mathrm{vol}(C)} \int_C \langle \mathbf{v}, \mathbf{w} \rangle^2 \, d\mathbf{w} = 1. \tag{A.3}$$

Lemma A.2
For any bounded convex set $C \subset \mathbb{R}^N$ in isotropic position, it holds that

$$\frac{1}{\mathrm{vol}(C)} \int_C \|\mathbf{w}\|^2 \, d\mathbf{w} = N. \tag{A.4}$$

Proof Denoting standard unit vectors in $\mathbb{R}^N$ by $\mathbf{e}_i^\top = (0, \ldots, 1, \ldots, 0)$ with component i being 1, we conclude from the definition of isotropy that

$$\frac{1}{\mathrm{vol}(C)} \int_C \|\mathbf{w}\|^2 \, d\mathbf{w} = \sum_{i=1}^{N} \frac{1}{\mathrm{vol}(C)} \int_C \langle \mathbf{e}_i, \mathbf{w} \rangle^2 \, d\mathbf{w} = N. \tag{A.5}$$

■

Lemma A.3
Any bounded convex set $C \subset \mathbb{R}^N$ can be transformed into isotropic position by the following affine mapping:

$$\mathbf{w} \mapsto A^{-\frac{1}{2}}\left(\mathbf{w} - \mathbf{w}^{(center)}\right) \quad \text{with} \quad A \overset{def}{=} \mathsf{E}\Big(\left(\mathsf{W} - \mathbf{w}^{(center)}\right)\left(\mathsf{W} - \mathbf{w}^{(center)}\right)^\top\Big) \tag{A.6}$$

where $\mathbf{w}^{(center)}$ denotes the center of mass of C and the random variable W is uniformly distributed over C.

Proof Trivial. ■

Lemma A.4
Suppose we are given a bounded convex set $C \subset \mathbb{R}^N$ in isotropic position. For any unit vector $\mathbf{v} \in \mathbb{R}^N$, we define a function

$$g_{\mathsf{v}} : \mathbb{R} \to \mathbb{R}_{\geq 0} \tag{A.7}$$
$$t \mapsto \frac{1}{\mathrm{vol}(C)} \int_{\mathsf{w} \in C, \langle \mathsf{v}, \mathsf{w} \rangle = t} d\mathbf{w}. \tag{A.8}$$

Then,

$$\sup_{t \in \mathbb{R}} g_{\mathsf{v}}(t) < 1. \tag{A.9}$$

Proof See (Bertsimas and Vempala, 2002). ■

Theorem A.5 (Bertsimas and Vempala, 2002)
Fix $0 < \delta < 1$ and $\varepsilon > 0$. If we approximate the center of mass of a bounded N-dimensional convex set C by averaging over $M \stackrel{\text{def}}{=} \frac{N}{\delta \varepsilon^2}$ samples independently and uniformly drawn from this set, then with probability of at least $1 - \delta$ a fraction of at least $\frac{1}{e} - \varepsilon$ of the entire volume is located on either side of any hyperplane passing through this approximation.

Proof Without loss of generality, we assume that C is in isotropic position. This assumption does not effect generality because

1. any bounded convex set can be transformed into isotropic position by an affine mapping according to Lemma A.3 and
2. on applying an affine mapping $\mathbf{w} \mapsto A^{-\frac{1}{2}} \left(\mathbf{w} - \mathbf{w}^{(\text{center})} \right)$ to C the volume scales by $\det(A^{-\frac{1}{2}})$, i.e., ratios of volumes are preserved.

Let $\mathsf{W}_1, \ldots, \mathsf{W}_M$ denote independent random variables which are uniformly distributed over C and let

$$\mathsf{W}^{(\text{bpm})} \stackrel{\text{def}}{=} \frac{1}{M} \sum_{i=1}^{M} \mathsf{W}_i \tag{A.10}$$

denote their average. Since C is assumed to be in isotropic position, it follows for $i = 1, \ldots, M$,

$$\mathrm{E}(\mathsf{W}_i) = 0 \quad \text{and} \quad \mathrm{Var}(\|\mathsf{W}_i\|) = \mathrm{E}(\|\mathsf{W}_i\|^2) = N. \tag{A.11}$$

Thus,

$$\mathrm{E}(\mathsf{W}^{(\text{bpm})}) = 0 \quad \text{and} \quad \mathrm{Var}(\|\mathsf{W}^{(\text{bpm})}\|) = \mathrm{E}(\|\mathsf{W}^{(\text{bpm})}\|^2) = \frac{N}{M}. \tag{A.12}$$

Let $\mathbf{v}$ denote an arbitrary unit vector in $\mathbb{R}^N$. We define the one-dimensional marginal distribution g_{v} by

$$g_{\mathrm{v}} : \mathbb{R} \to \mathbb{R}_{\geq 0} \tag{A.13}$$
$$t \mapsto \frac{1}{\mathrm{vol}(C)} \int_{\mathrm{w} \in C, \langle \mathrm{v}, \mathrm{w} \rangle = t} d\mathbf{w}. \tag{A.14}$$

In other words, $g_{\mathrm{v}}(t)$ is the $(N-1)$-dimensional volume of C intersected with the hyperplane $\{\mathbf{w} \in \mathbb{R}^N \mid \langle \mathbf{v}, \mathbf{w} \rangle = t\}$. As stated in Lemma A.4, it holds that $\sup_{t \in \mathbb{R}} g_{\mathrm{v}}(t) < 1$.
Since either side of a hyperplane through the center of mass of a convex set contains a fraction of at least $\frac{1}{e}$ of the volume (Grünbaum, 1960), we conclude that

$$\int_{-\infty}^{0} g_{\mathrm{v}}(t)\, dt \geq \frac{1}{e} \quad \text{and} \quad \int_{0}^{\infty} g_{\mathrm{v}}(t)\, dt \geq \frac{1}{e}. \tag{A.15}$$

Without loss of generality, we assume that $\langle \mathbf{v}, \mathrm{W}^{(\mathrm{bpm})} \rangle \leq 0$. The fraction of the volume of C that is cut off by a halfspace through $\mathrm{W}^{(\mathrm{bpm})}$ defined by the normal vector $\mathbf{v}$ is at least

$$\int_{-\infty}^{\langle \mathrm{v}, \mathrm{W}^{(\mathrm{bpm})} \rangle} g_{\mathrm{v}}(t)\, dt = \int_{-\infty}^{0} g_{\mathrm{v}}(t)\, dt - \int_{\langle \mathrm{v}, \mathrm{W}^{(\mathrm{bpm})} \rangle}^{0} g_{\mathrm{v}}(t)\, dt \tag{A.16}$$
$$\geq \frac{1}{e} - |\langle \mathbf{v}, \mathrm{W}^{(\mathrm{bpm})} \rangle| \tag{A.17}$$
$$\geq \frac{1}{e} - \|\mathrm{W}^{(\mathrm{bpm})}\| \tag{A.18}$$

where the last step follows from the Cauchy-Schwarz inequality.
By the Tschebyscheff inequality and (A.12), we obtain

$$P(\|\mathrm{W}^{(\mathrm{bpm})}| \geq \varepsilon) \leq \frac{\mathrm{E}(\|\mathrm{W}^{(\mathrm{bpm})}\|^2)}{\varepsilon^2} = \frac{N}{M\varepsilon^2} = \delta. \tag{A.19}$$

Therefore, with probability of at least $1 - \delta$

$$\int_{-\infty}^{\langle \mathrm{v}, \mathrm{W}^{(\mathrm{bpm})} \rangle} g_{\mathrm{v}}(t)\, dt \geq \frac{1}{e} - \varepsilon. \tag{A.20}$$

■

B Experimental Results: Multiclass Learning

The present section contains the complete set of learning curves for the experimental evaluation of multiclass active learning. Employing the one-versus-all, the one-versus-one and the DAG techniques for decomposing multiclass classification problems into binary ones, we compare the BinMin and the Global selection strategies, which were proposed in Chapter 6, to Random selection of new training examples. The experimental setup includes a variety of datasets both from the UCI repository of machine learning databases (Blake and Merz, 1998) and from the Statlog collection (Michie et al., 1994). Further details on the experimental evaluation procedure are given in Section 6.4.1.

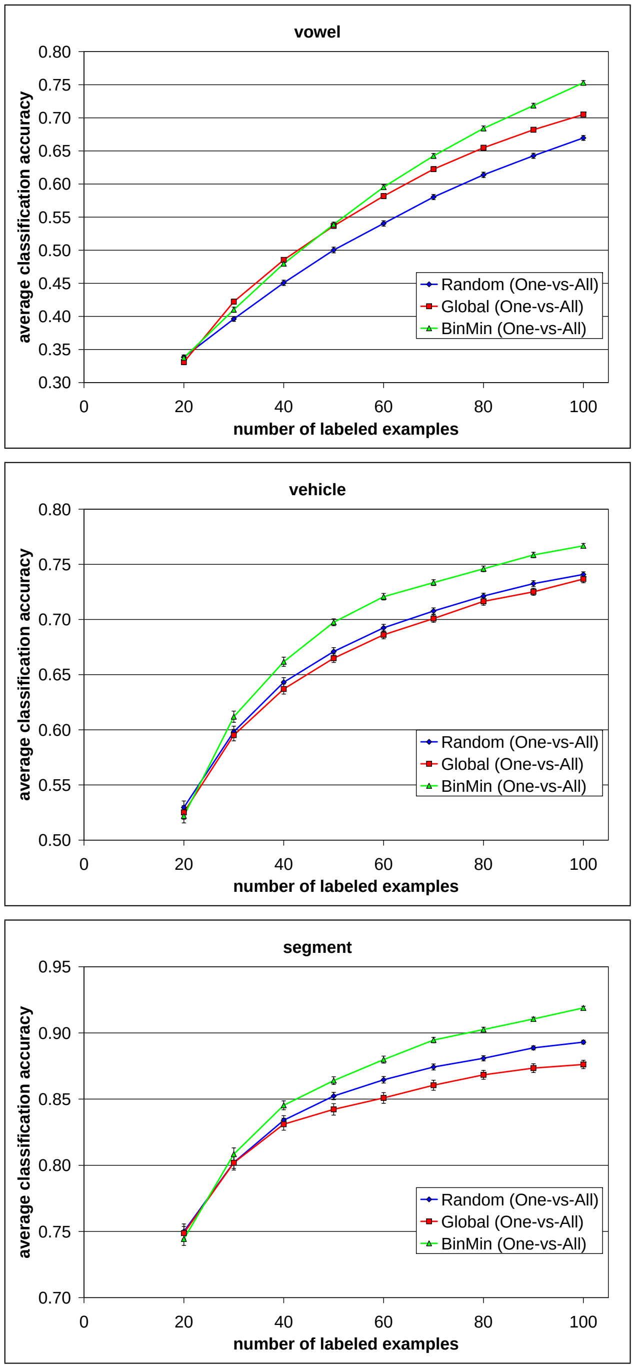


Figure B.1 *Experimental learning curves for multiclass active learning based on the one-versus-all decomposition.*

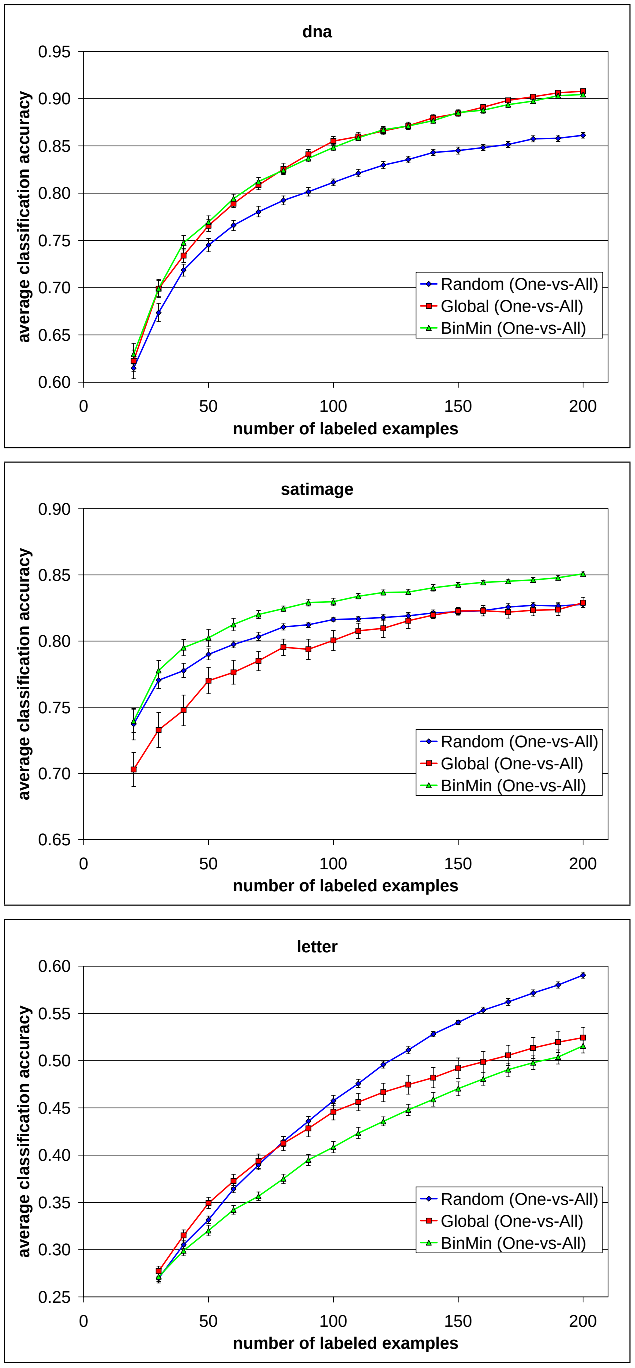


Figure B.2 *Experimental learning curves for multiclass active learning based on the one-versus-all decomposition.*

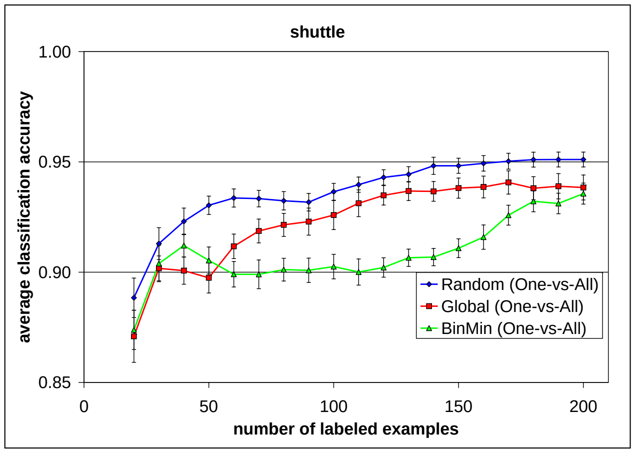
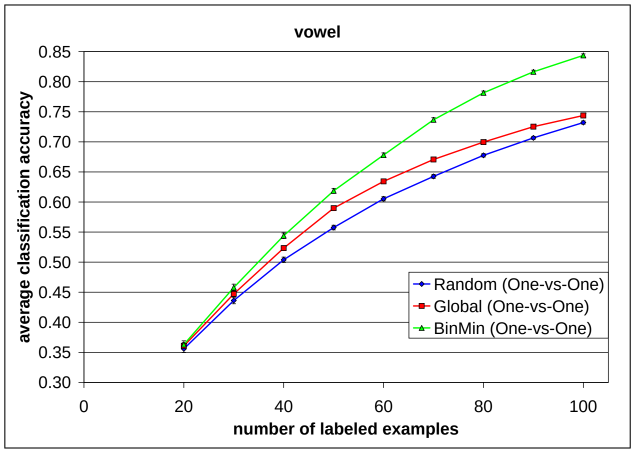
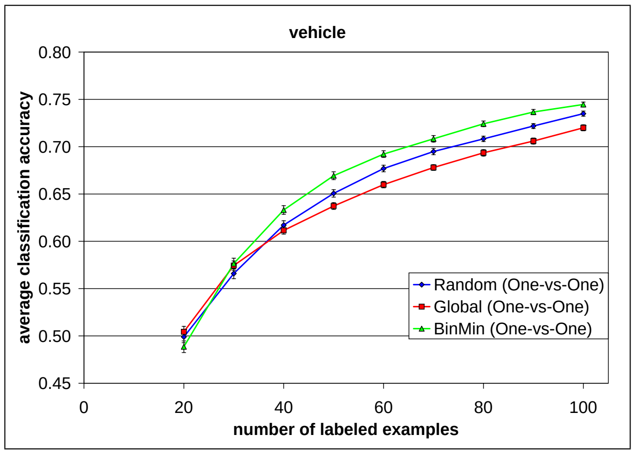


Figure B.3 *Experimental learning curves for multiclass active learning based on the one-versus-all decomposition.*





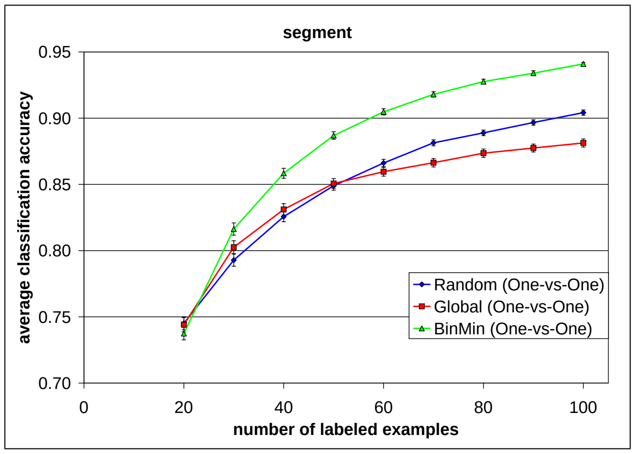


Figure B.4 *Experimental learning curves for multiclass active learning based on the one-versus-one decomposition.*

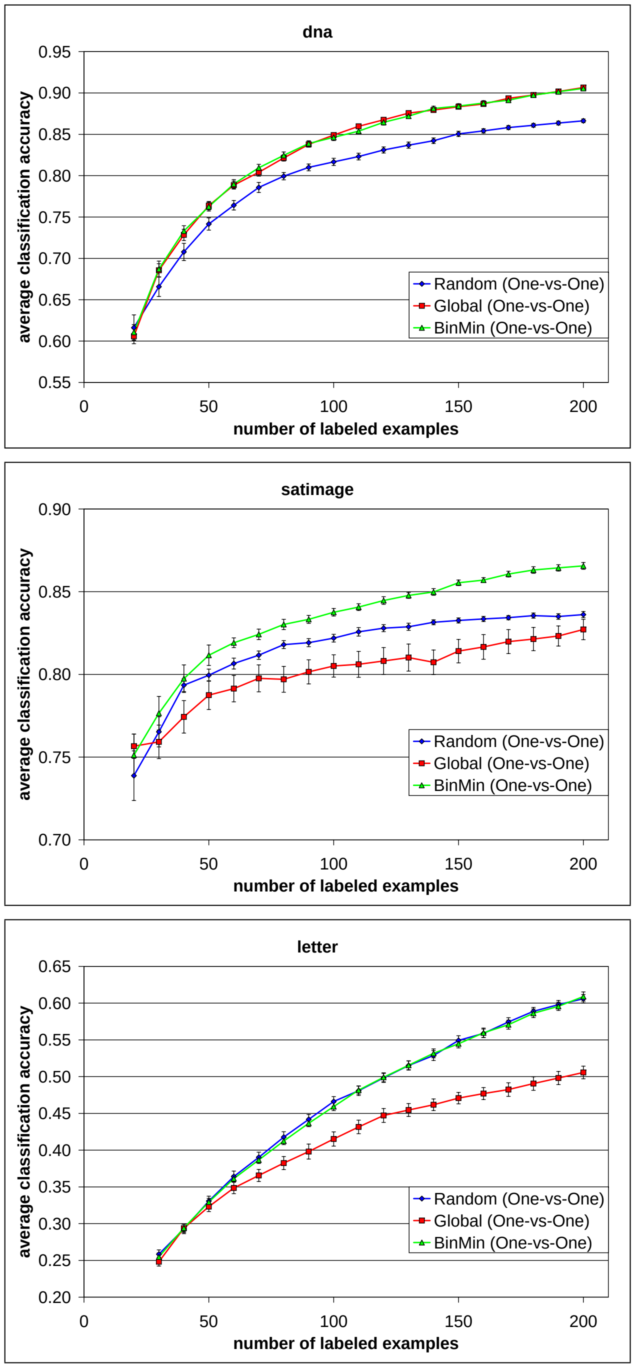


Figure B.5 *Experimental learning curves for multiclass active learning based on the one-versus-one decomposition.*

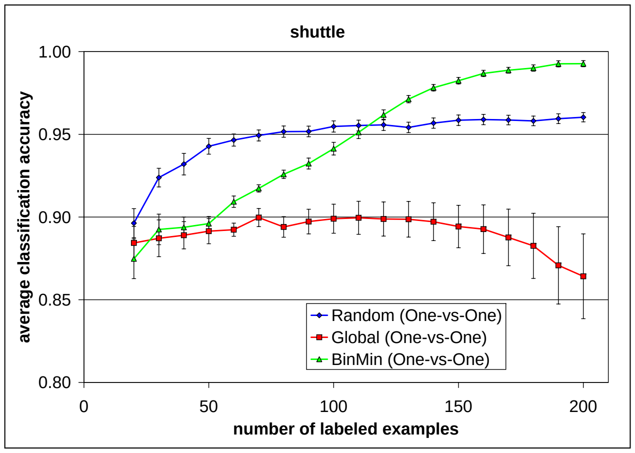


Figure B.6 *Experimental learning curves for multiclass active learning based on the one-versus-one decomposition.*

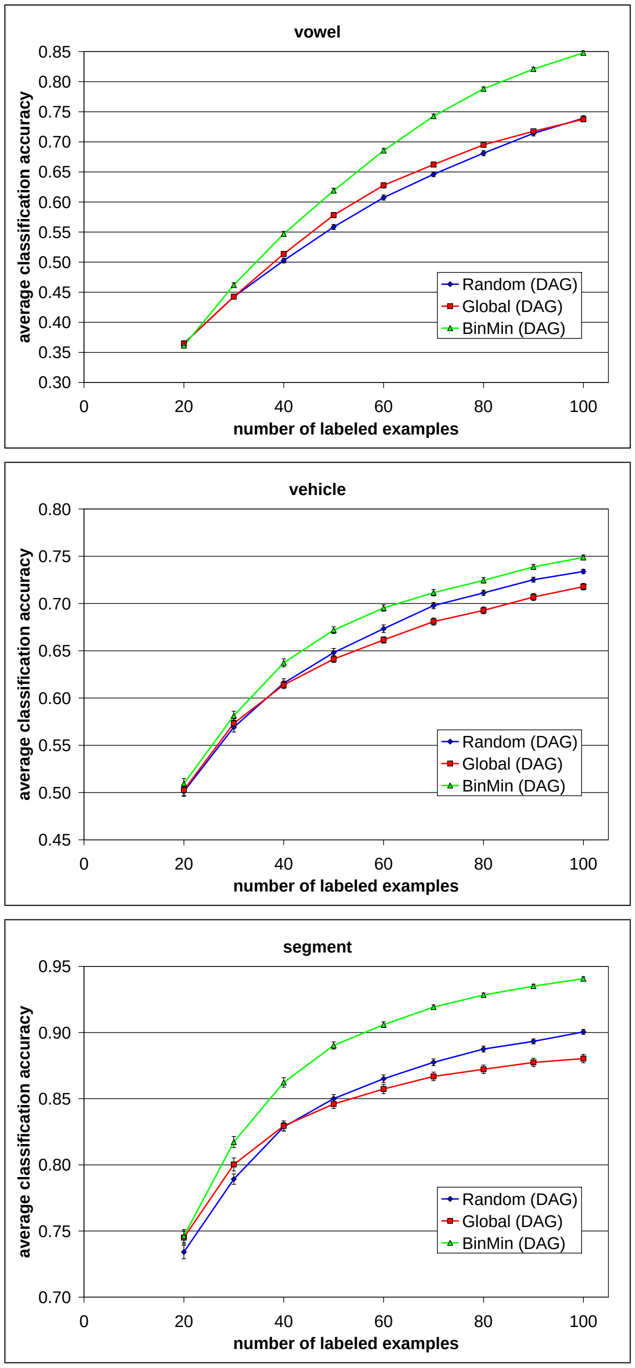
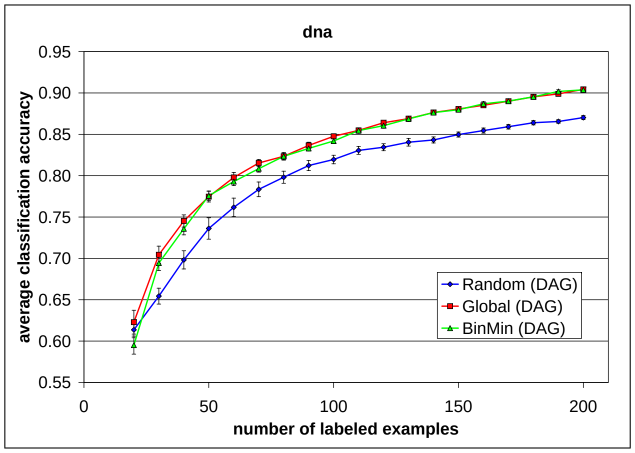
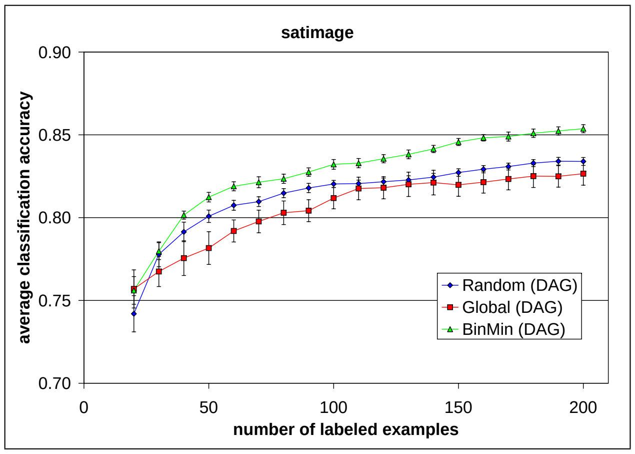


Figure B.7 *Experimental learning curves for multiclass active learning based on the DAG technique.*





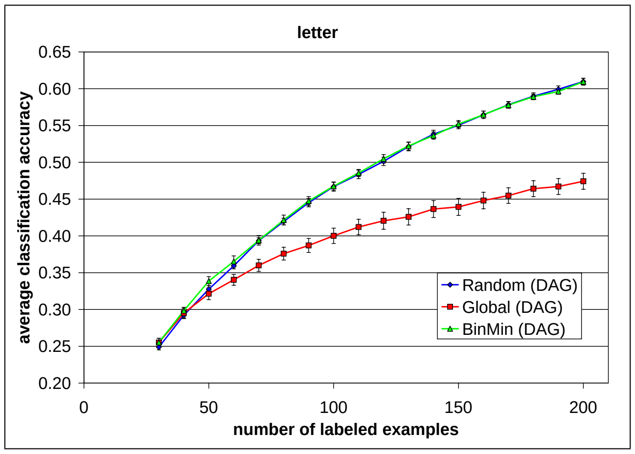


Figure B.8 *Experimental learning curves for multiclass active learning based on the DAG technique.*

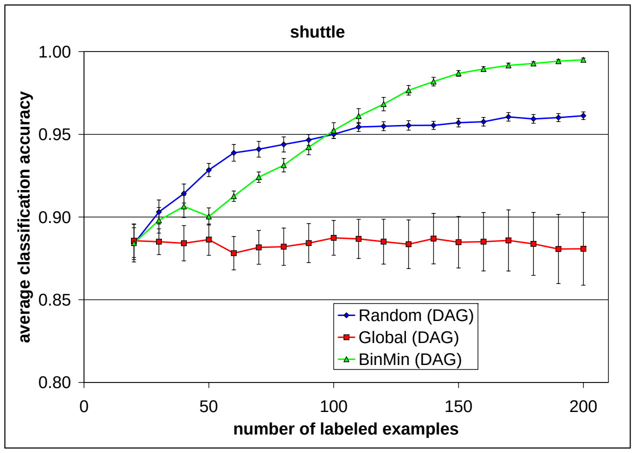
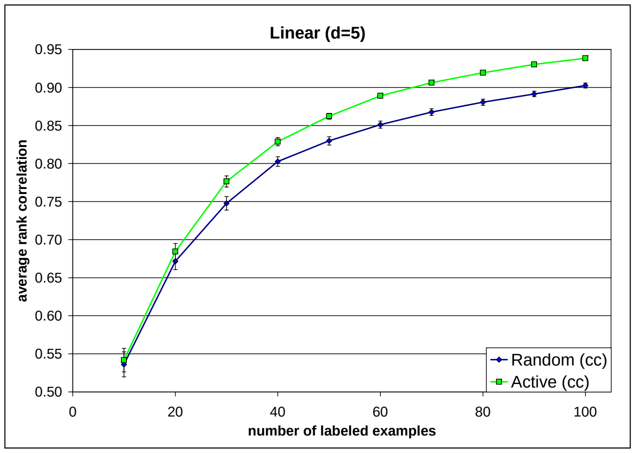
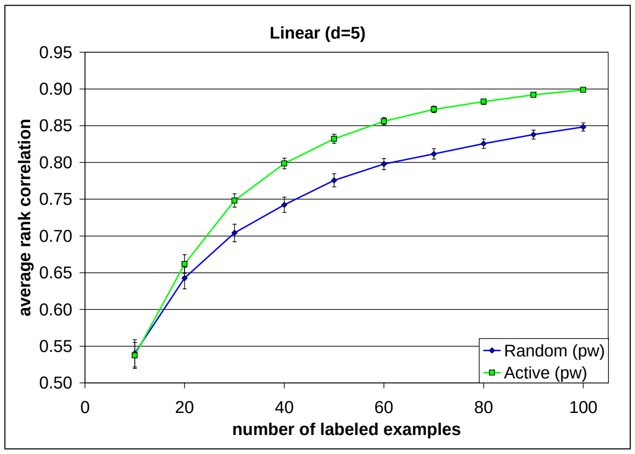


Figure B.9 *Experimental learning curves for multiclass active learning based on the DAG technique.*

C Experimental Results: Label Ranking Learning

This section contains the complete set of learning curves for the experimental evaluation of active learning in the case of label ranking. Employing the constraint classification and the pairwise ranking techniques for solving label ranking problems, we compare the active selection strategies which were proposed in Chapter 7 to the corresponding random counterparts. The experimental setup includes a variety of datasets originating from different settings, which were described in Section 7.5.1.





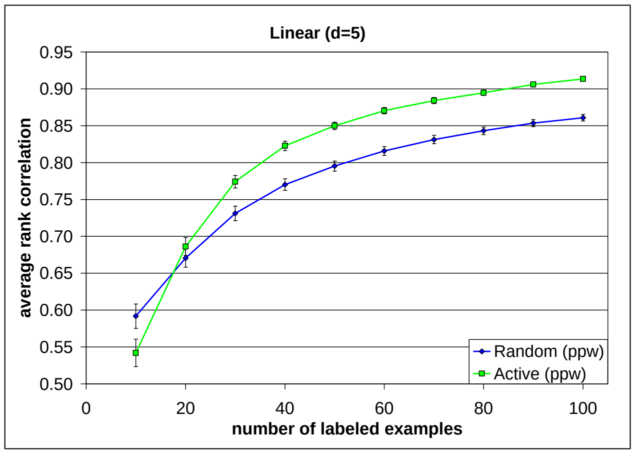
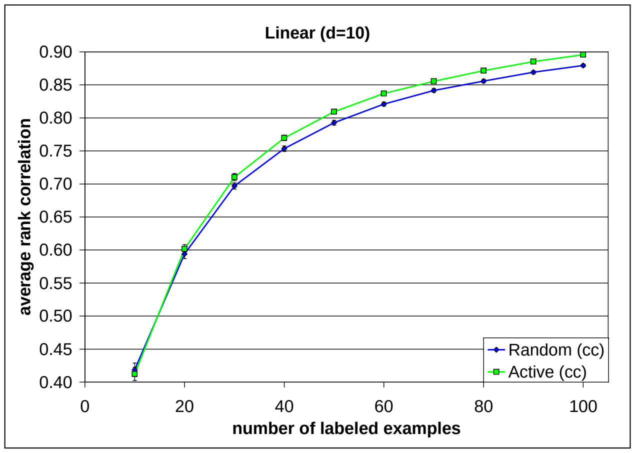
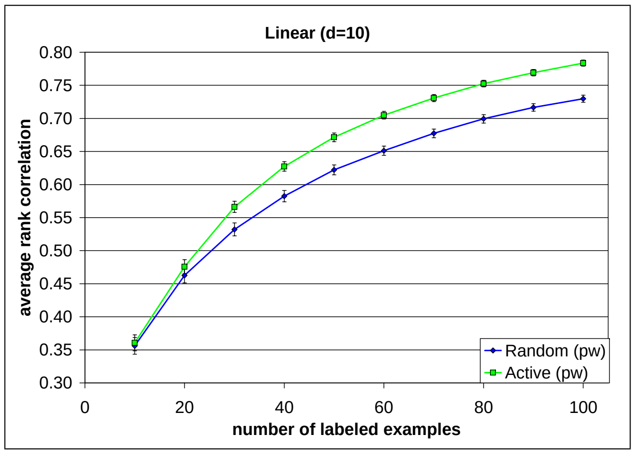


Figure C.1 *Experimental learning curves for label ranking learning on the Linear problem (for $d = 5$).*





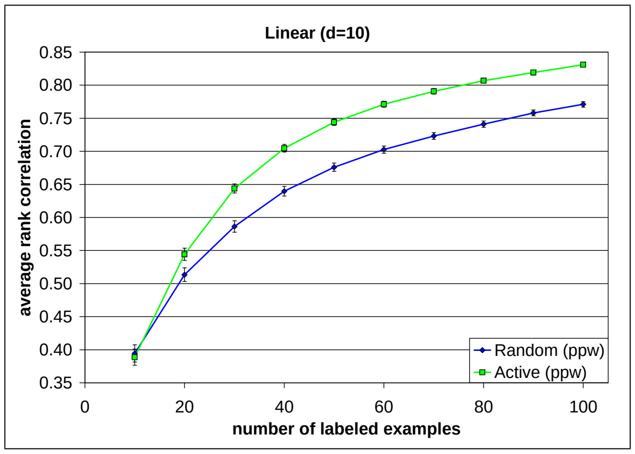


Figure C.2 *Experimental learning curves for label ranking learning on the Linear problem (for $d = 10$).*

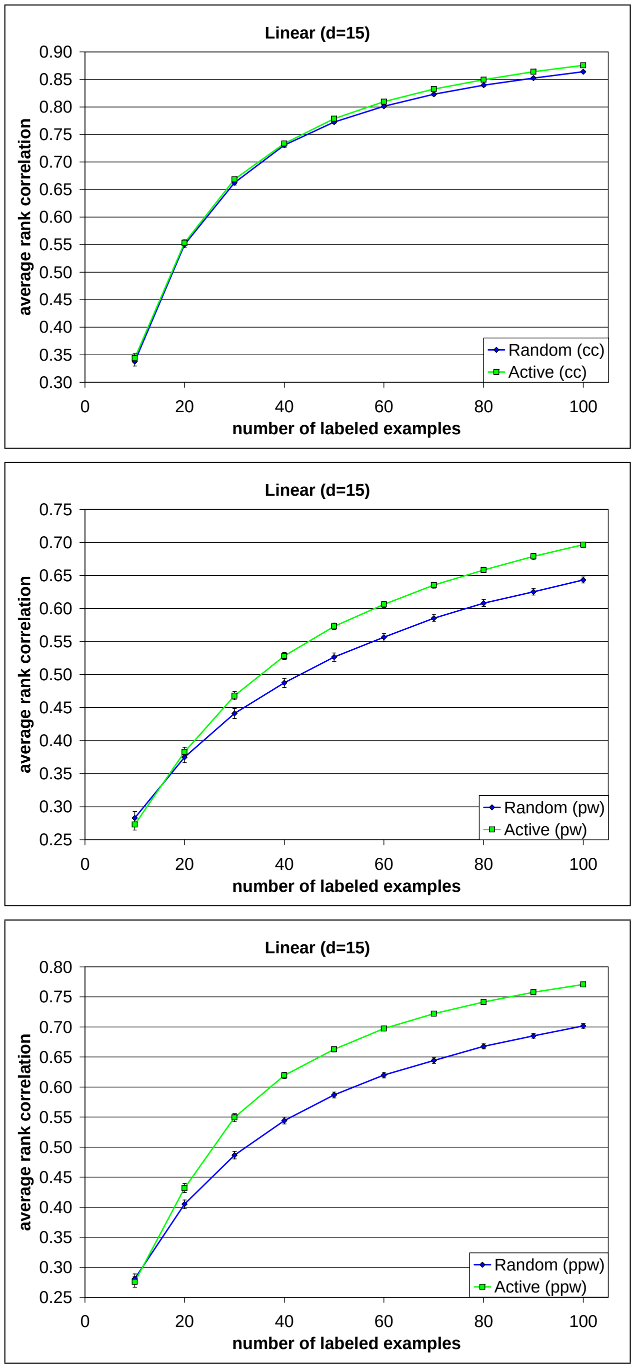


Figure C.3 *Experimental learning curves for label ranking learning on the Linear problem (for $d = 15$).*

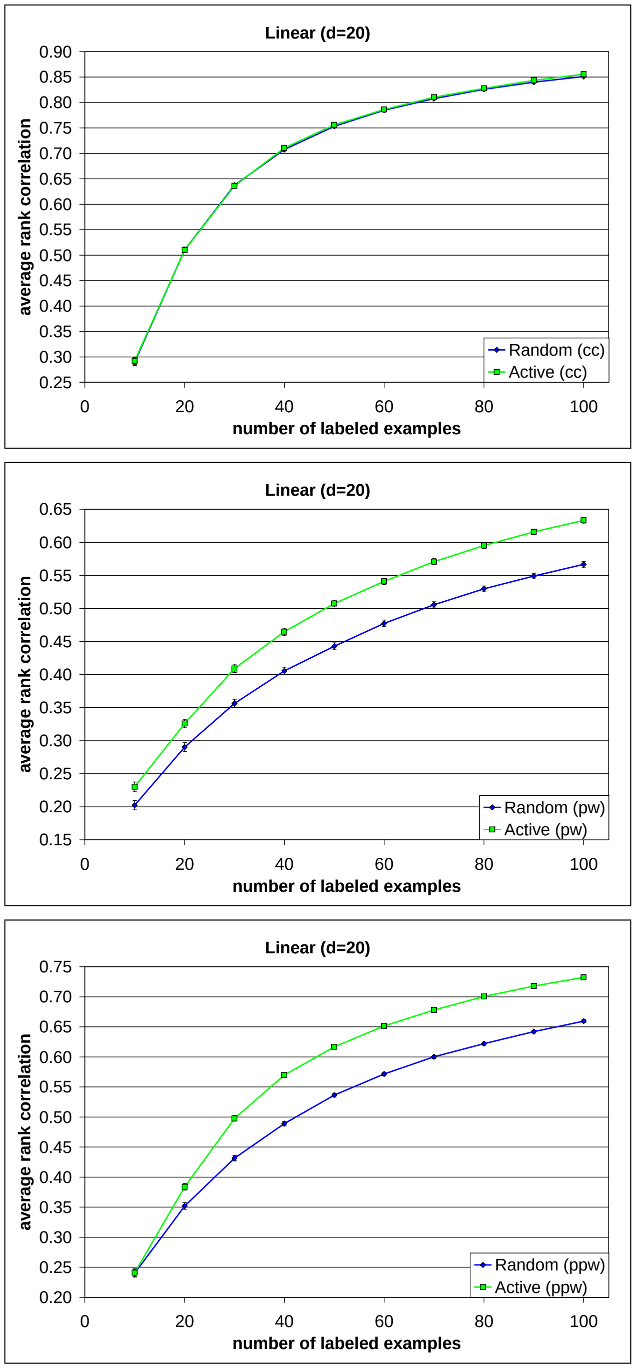
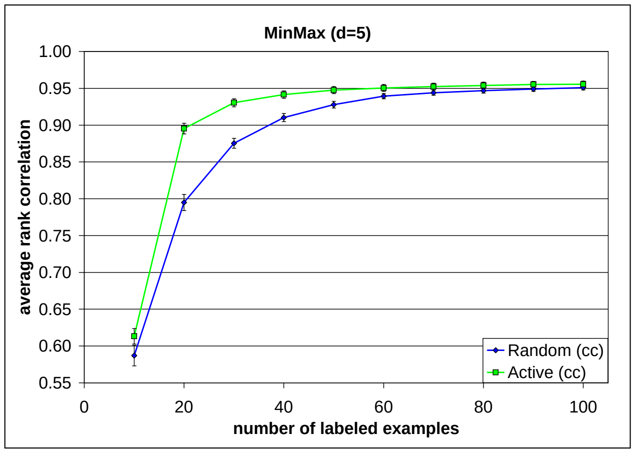
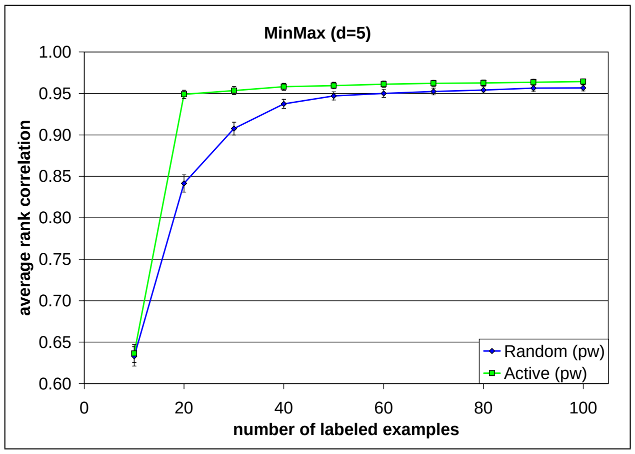


Figure C.4 *Experimental learning curves for label ranking learning on the Linear problem for* $(d = 20)$.





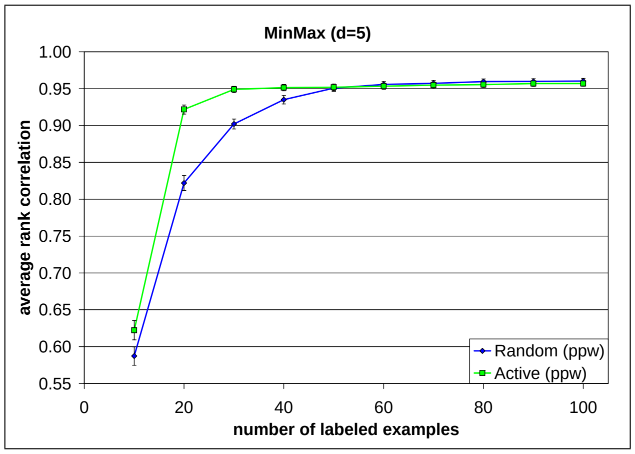
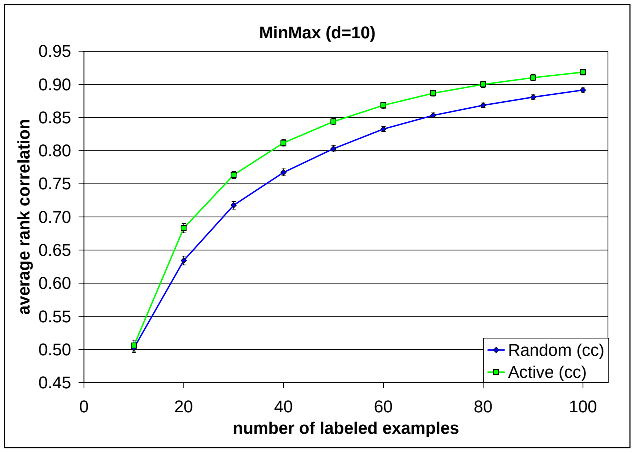
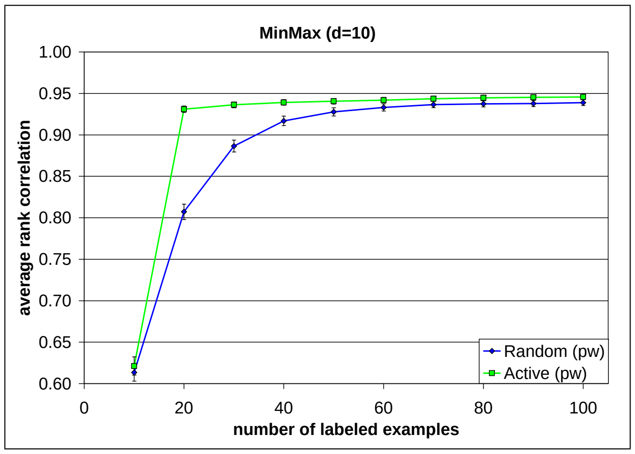


Figure C.5 *Experimental learning curves for label ranking learning on the MinMax problem (for $d = 5$).*





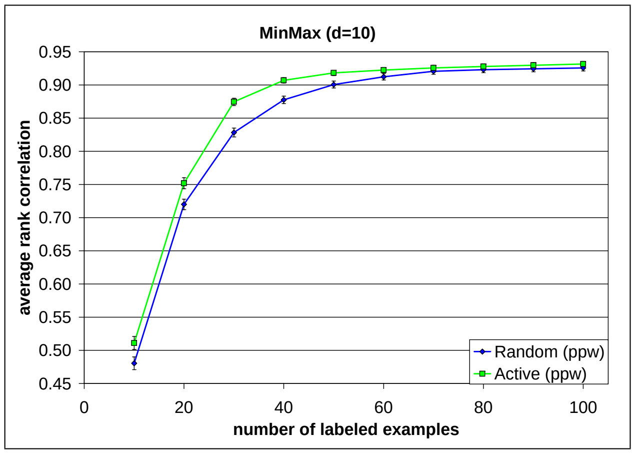
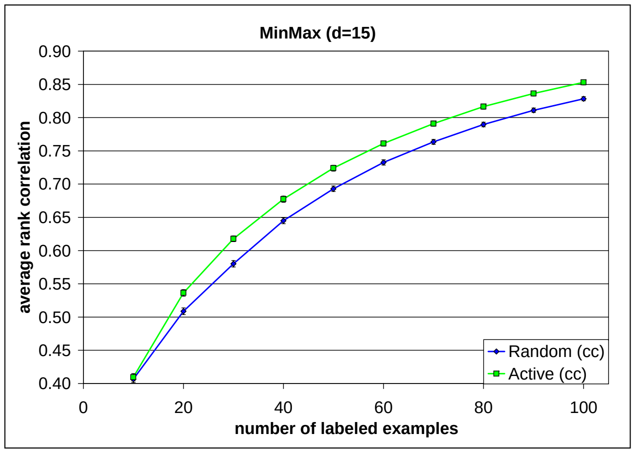
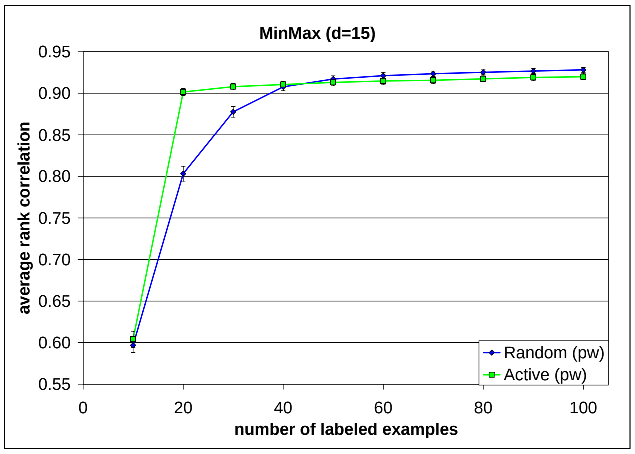


Figure C.6 *Experimental learning curves for label ranking learning on the MinMax problem (for* $d = 10$*).*





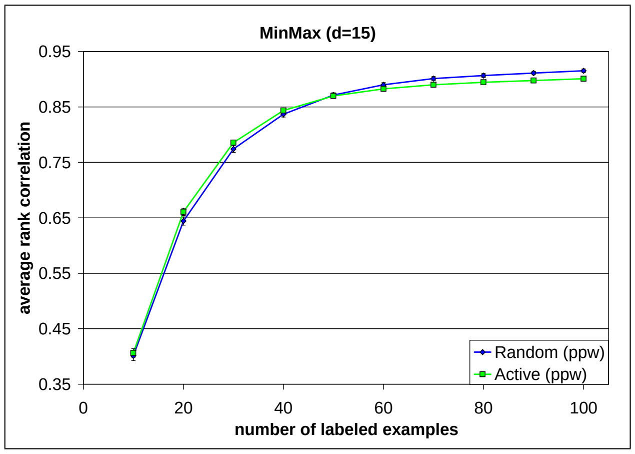
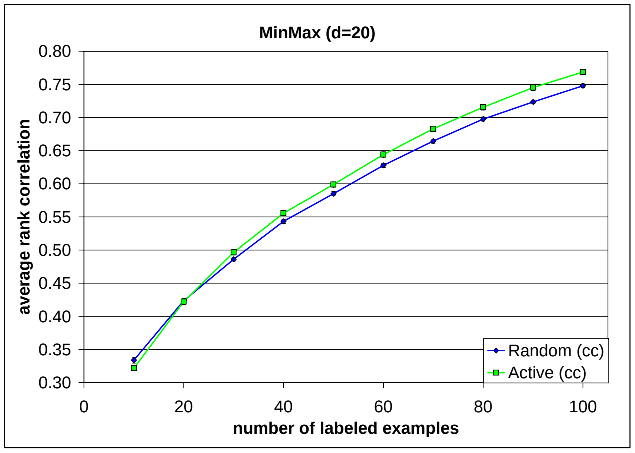
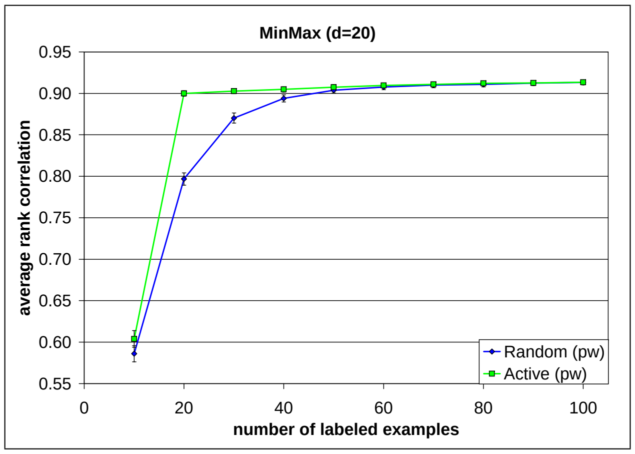


Figure C.7 *Experimental learning curves for label ranking learning on the MinMax problem (for $d = 15$).*





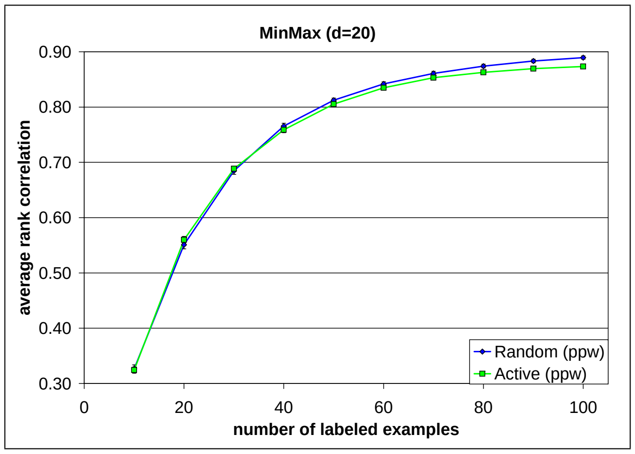
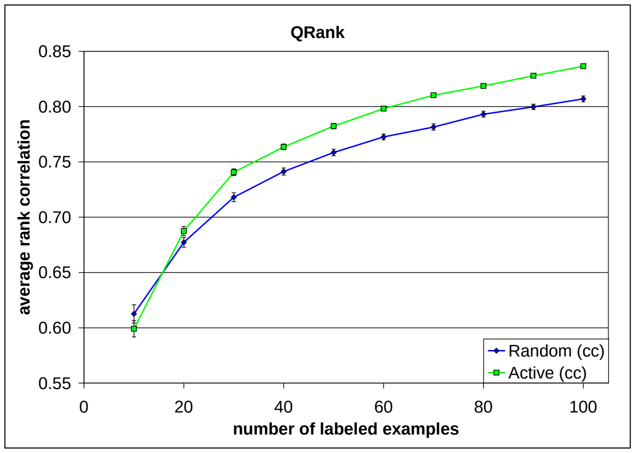
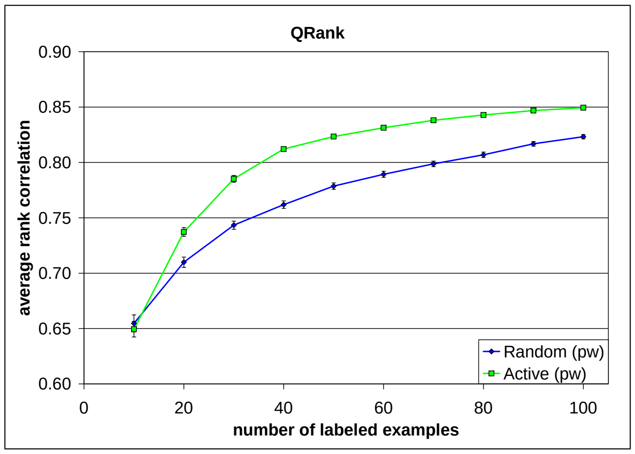


Figure C.8 *Experimental learning curves for label ranking learning on the MinMax problem (for $d = 20$).*





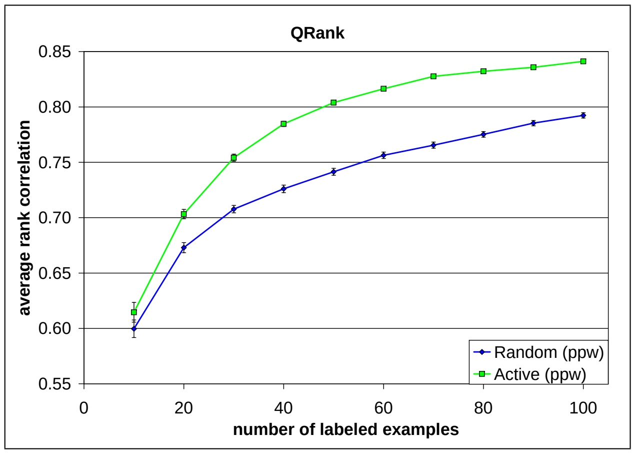


Figure C.9 *Experimental learning curves for label ranking learning on the QRank problem.*

List of Figures

List of Tables

Notation

We largely adopt the notation introduced by Schölkopf and Smola (2002). Unless otherwise noted, the following symbols are used throughout this thesis.

$\mathbb{N}$	the set of natural numbers
$\mathbb{R}$	the set of reals
$\mathcal{X}$	input space, e.g., $\mathbb{R}^N$
$\mathcal{Y}$	target space, e.g., $\{-1, +1\}$, $\{1, \ldots, d\}$
N	dimension of input space
$N_{\mathcal{F}}$	dimension of feature space
d	number of classes (in the case of multiclass problems), number of alternatives (in the case of label ranking problems)
$\mathcal{S}^{(d)}$	symmetric group of degree d, set of permutations of length d
$x, x_i \in \mathcal{X}$	input pattern, unlabeled example
$\mathbf{x}, \mathbf{x}_i$	input pattern in the case of $\mathcal{X} = \mathbb{R}^N$
$[\mathbf{x}]_i$, $[\bar{\mathbf{w}}]_i$	ith component of a vector $\mathbf{x} \in \mathbb{R}^N$ and $\bar{\mathbf{w}} \in \mathcal{F}^d$, respectively
$y, y_i \in \mathcal{Y}$	target object
$z = (x, y)$	labeled example
$\mathrm{X}, \mathrm{Y}, \mathrm{Z}$	random variables on $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{X} \times \mathcal{Y}$
m	number of labeled examples
n	number of unlabeled examples
$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$	kernel (function)
$\mathcal{F}$	kernel-induced feature space
$\phi : \mathcal{X} \to \mathcal{F}$	kernel feature map
$\mathbf{w}$	weight vector in feature space
b	bias, constant offset, threshold
$\mathcal{H}$	hypothesis space of linear classifiers
$\mathcal{W}$	weight space

$\mathcal{U} = \{x_1, \ldots, x_n\}$	set of unlabeled examples/patterns
$\mathcal{L} = \{(x_1, y_1), \ldots, (x_m, y_m)\}$	set of labeled examples
$\lvert \cdot \rvert$	absolute value (if argument is a real number), cardinality (if argument is a set)
$\langle \cdot, \cdot \rangle$	standard dot product in $\mathbb{R}^N$
$\langle \cdot, \cdot \rangle_{\mathcal{F}}$	dot product in kernel-induced feature space
$\lVert \cdot \rVert$	2-norm in $\mathbb{R}^N$ (Euclidean distance), $\lVert \mathbf{x} \rVert \overset{\text{def}}{=} \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$
$\lVert \cdot \rVert_{\mathcal{F}}$	2-norm in kernel-induced feature space
P	probability measure
P^{ζ}	probability distribution of a random variable ζ
$\mathrm{E}(\zeta)$	expectation of a random variable ζ
$f : \mathcal{X} \to \mathbb{R}$	real valued function
$h : \mathcal{X} \to \mathcal{Y}$	prediction function, hypothesis (in the case of binary classification $h(x) = \mathrm{sign}(f(x))$)
$\rho_f(x, y)$	margin of function f on the example (x, y) (in the case of binary classification $yf(x)$)
ρ_f	minimum margin of f on the training set, i.e., $\rho_f \overset{\text{def}}{=} \min_{1 \le i \le m} \rho_f(x_i, y_i)$
$\mathrm{R}(h)$	expected risk of h
$\mathrm{R}_{\mathrm{emp}}(h)$	empirical risk of h
α_i	Lagrange multiplier
ξ_i	slack variable
C	regularization constant for support vector machines
l_p^N	space of absolutely convergent sequences of length $N \in \mathbb{N} \cup \{\infty\}$ with respect to the p-norm $(p \in \mathbb{N} \cup \{\infty\})$
$L_p(\Omega)$	space of μ-integrable functions on a finite measure space (Ω, μ) with respect to the p-norm $(p \in \mathbb{N} \cup \{\infty\})$
span	linear span of a set
$\mathcal{B}_r^{(N)}$, $\mathcal{B}_r^{(N)}(c)$	closed N-dimensional ball of radius $r > 0$ (and with center c)
$\mathcal{C}_{h,r}^{(N)}$	N-dimensional cone of height h and basis radius $r > 0$

References

Abe, Naoki, and Mamitsuka, Hiroshi. Query learning strategies using boosting and bagging. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998)*, pages 1–9, 1998.

Allwein, Erin L., Schapire, Robert E., and Singer, Yoram. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000.

Angluin, Dana. Queries and concept learning. *Journal of Machine Learning*, 2:319–342, 1988.

Argamon-Engelson, Shlomo, and Dagan, Ido. Committee-based sample selection for probabilistic classifiers. *Journal of Artificial Intelligence Research*, 11:335–360, 1999.

Baldi, Pierre, and Brunak, Soeren. *Bioinformatics : The Machine Learning Approach*. MIT Press, 2001.

Baram, Yoram, El-Yaniv, Ran, and Luz, Kobi. Online choice of active learning algorithms. *Journal of Machine Learning Research*, 5:255–291, 2004. ISSN 1533-7928.

Baum, Eric B. Neural net algorithms that learn in polynomial time from examples and queries. *IEEE Transactions on Neural Networks*, 2(1):5–19, 1991.

Ben-David, Shai, and Simon, Hans Ulrich. Efficient learning of linear perceptrons. In Leen, Todd K., Dietterich, Thomas G., and Tresp, Volker, editors, *Advances in Neural Information Processing Systems 13 (NIPS 2000)*, pages 189–195. MIT Press, 2001.

Bennett, Kristin P., and Demiriz, Ayhan. Semi-supervised support vector machines. In *Advances in Neural Information Processing Systems 11 (NIPS 1998)*, pages 368–374. MIT Press, 1999. ISBN 0-262-11245-0.

Bertsimas, Dimitris, and Vempala, Santosh. Solving convex programs by random walks. In *Proceedings of the 34th ACM Symposium on the Theory of Computing (STOC '02)*, pages 109–115, Montreal, 2002.

Blake, C. L., and Merz, C. J. UCI repository of machine learning databases, 1998. Data available at `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Boser, Bernhard E., Guyon, Isabelle, and Vapnik, Vladimir. A training algorithm for optimal margin classifiers. In *Computational Learing Theory*, pages 144–152, 1992.

Brinker, Klaus. Active learning of preferences. In *Tagungsband der GI-Informatiktage*

2003, pages 62–65, Bad Schussenried, 2003a. Konradin-Verlag.

Brinker, Klaus. Active learning of ranking functions. In *Proceedings of the 26th German Conference on Artificial Intelligence (KI-2003) Workshop - Preference Learning: Models, Methods, Applications*, pages 23–33, 2003b.

Brinker, Klaus. Incorporating diversity in active learning with support vector machines. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003)*, pages 59–66, Menlo Park, California, 2003c. AAAI Press.

Brinker, Klaus. Active learning of label ranking functions. In Greiner, Russ, and Schuurmans, Dale, editors, *Proceedings of the Twenty-First International Conference on Machine Learning (ICML 2004)*, pages 129–136, 2004a.

Brinker, Klaus. On multiclass active learning with support vector machines. In Mántaras, de Ramon López, and Saitta, Lorenza, editors, *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004)*, pages 969–970. IOS Press, 2004b.

Burges, Christopher J. C. A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining*, 2(2), 1998.

Campbell, Colin, Cristianini, Nello, and Smola, Alex J. Query learning with large margin classifiers. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, pages 111–118, 2000.

Cauwenberghs, Gert, and Poggio, Tomaso. Incremental and decremental support vector machine learning. In *Advances in Neural Information Processing Systems 13 (NIPS 2000)*, volume 13, 2001.

Chang, Chih-Chung, and Lin, Chih-Jen. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Cohen, William W., Schapire, Robert E., and Singer, Yoram. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.

Cohn, David A., Atlas, Les, and Ladner, Richard E. Improving generalization with active learning. *Journal of Machine Learning*, 15(2):201–221, 1994.

Cohn, David A., Ghahramani, Zoubin, and Jordan, Michael I. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

Cortes, Corinna, and Vapnik, Vladimir. Support vector networks. *Journal of Machine Learning*, 20:273 – 297, 1995.

Crammer, Koby, and Singer, Yoram. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

Crammer, Koby, and Singer, Yoram. Pranking with ranking. In Dietterich, T. G., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems 14 (NIPS 2001)*, pages 641–647, Cambridge, MA, 2002. MIT Press.

Cristianini, Nello, and Shawe-Taylor, John. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

DeCoste, Dennis. Anytime interval-valued outputs for kernel machines: Fast support vector machine classification via distance geometry. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 99–106. Morgan Kaufmann Publishers Inc., 2002. ISBN 1-55860-873-7.

Dietterich, Thomas G., and Bakiri, Ghulum. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2: 263–286, 1995.

Dubois, Didier, Prade, Henri, and Sabbadin, Régis. Decision-theoretic foundation of qualitative possibility theory. *European Journal of Operational Research*, 128(3): 459–478, 2001.

Dyer, Martin, Frieze, Alan, and Kannan, Ravi. A random polynomial time algorithm for approximating the volume of convex sets. *Journal of the Association for Computing Machinary*, 38:1–17, 1991.

Eisenberg, Bonnie, and Rivest, Ronald L. On the sample complexity of pac-learning using random and chosen examples. In Fulk, M., and Case, J., editors, *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 154–162, San Mateo, CA, 1990. Morgan Kaufmann.

Elekes, György. A geometric inequality and the complexity of computing volume. *Discrete and Computational Geometry*, 1:289–292, 1986.

Elstrodt, Jürgen. *Maß- und Integrationstheorie*. Springer, 1996.

Fine, Shai, Gilad-Bachrach, Ran, and Shamir, Eli. Query by committee, linear separation and random walks. *Theoretical Computer Science*, 284(1):25–51, 2002. ISSN 0304-3975.

Freund, Yoav, Seung, H. Sebastian, Shamir, Eli, and Tishby, Naftali. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3): 133–168, 1997.

Friedman, Jerome H. Another approach to polychotomous classification. Technical report, Department of Statistics, Stanford University, Stanford, CA, 1996.

Fung, Glenn, and Mangasarian, Olvi L. Semi-supervised support vector machines for unlabeled data classification. *Optimization Methods and Software*, 15(1):29–44, 2001.

Fung, Glenn, Mangasarian, Olvi L., and Shavlik, Jude. Knowledge-based support vector machine classifiers. In Becker, S., Thrun, S., and Obermayer, K., editors, *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, pages 521–528, Cambridge, MA, 2003. MIT press.

Fürnkranz, Johannes. Round robin classification. *Journal of Machine Learning Research*, 2:721–747, 2002. ISSN 1533-7928.

Fürnkranz, Johannes, and Hüllermeier, Eyke. Pairwise preference learning and ranking. In *Proceedings of the 14th European Conference on Machine Learning (ECML 2003)*, pages 145–156, Cavtat, Croatia, 2003a. Springer-Verlag.

Fürnkranz, Johannes, and Hüllermeier, Eyke. Pairwise preference learning and ranking. Technical report, Austrian Research Institute for Artificial Intelligence, 2003b.

Gilad-Bachrach, Ran, Navot, Amir, and Tishby, Naftali. Kernel query by committee (KQBC). Technical Report 2003-88, Leibniz Center, the Hebrew University, 2003.

Gilad-Bachrach, Ran, Navot, Amir, and Tishby, Naftali. Bayes and tukey meet at the center point. In *Proceedings of the 17th Conference on Computational Theory (COLT 2004)*, 2004.

Goh, King-Shy, Chang, Edward Y., and Lai, Wei-Cheng. Multimodal concept-dependent active learning for image retrieval. In *ACM International Conference on Multimedia*, pages 564–571, New York, NY, 2004.

Graepel, Thore, Herbrich, Ralf, and Obermayer, Klaus. Bayesian transduction. In Solla, S.A., Leen, T.K., and Müller, K-R., editors, *Advances in Neural Information Processing Systems 12 (NIPS 1999)*, pages 456–462. MIT Press, 2000.

Grünbaum, Branko. Partitions of mass-distributions and convex bodies by hyperplanes. *Pacific J. Math.*, 10:1257–1261, 1960.

Haddawy, Peter, Ha, Vu, Restificar, Angelo, Geisler, Benjamin, and Miyamoto, John. Preference elicitation via theory refinement. *Journal of Machine Learning Research*, 4(3):317–337, 2004. ISSN 1533-7928.

Har-Peled, Sariel, Roth, Dan, and Zimak, Dav. Constraint classification: A new approach to multiclass classification and ranking. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, 2002.

Herbrich, Ralf. *Learning Kernel Classifiers*. MIT Press, 2002.

Herbrich, Ralf, and Graepel, Thore. Large scale bayes point machines. In Leen, Todd K., Dietterich, Thomas G., and Tresp, Volker, editors, *Advances in Neural Information Processing Systems 13 (NIPS 2000)*, pages 528–534. MIT Press, 2001.

Herbrich, Ralf, and Graepel, Thore. A pac-bayesian margin bound for linear classifiers. *IEEE Transactions on Information Theory*, 2002.

Herbrich, Ralf, Graepel, Thore, and Campbell, Colin. Bayes point machines. *Journal of Machine Learning Research*, 1:245–279, 2001.

Herbrich, Ralf, Graepel, Thore, and Obermayer, Klaus. *Advances in Large Margin Classifiers*, chapter Large margin rank boundaries for ordinal regression, pages 115–132. MIT Press, Cambridge, MA, 2000.

Hsu, Chih-Wei, and Lin, Chih-Jen. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425, 2002a.

Hsu, Chih-Wei, and Lin, Chih-Jen. A simple decomposition method for support vector machines. *Journal of Machine Learning*, 46:291–314, 2002b. Implementation available at `http://www.csie.ntu.edu.tw/~cjlin/bsvm/`.

Hüllermeier, Eyke, and Fürnkranz, Johannes. Comparison of ranking procedures in pairwise preference learning. In *10th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU–04)*, pages 535–542, 2004a.

Hüllermeier, Eyke, and Fürnkranz, Johannes. Ranking by pairwise comparison: A note on risk minimization. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE–04)*, 2004b.

Joachims, Thorsten. Making large–scale SVM learning practical. In Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods — Support Vector Learning*, pages 169–184, Cambridge, MA, 1999a. MIT Press.

Joachims, Thorsten. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999)*, Bled, Slovenia, 1999b.

Joachims, Thorsten. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2002.

Kannan, Ravi, Lovász, László, and Simonovits, Miklós. Random walks and an $O^*(n^5)$ volume algorithm for convex bodies. *Random Struct. Algorithms*, 11(1):1–50, 1997.

Kannan, Ravi, and Nolte, Andreas. Local search in smooth convex sets. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS '98)*, pages 218–226, 1998.

Knerr, Stefan, Personnaz, Léon, and Dreyfus, Gérard. Single-layer learning revisited: A stepwise procedure for building and training a neural network. In Soulié, F. Fogelman, and Hérault, J., editors, *Neurocomputing: Algorithms, Architectures and Applications*, volume F68 of NATO ASI Series, pages 41–50. Springer-Verlag, 1990.

Langley, Pat. Machine learning for adaptive user interfaces. In *Proceedings of the 21st German Annual Conference on Artificial Intelligence (KI-1997)*, pages 53–62, 1997.

Lehmann, Erich L., and D'Abrera, H. J. M. *Nonparametrics: Statistical Methods Based on Ranks, rev. ed.* Prentice-Hall, Englewood Cliffs, NJ, 1998.

Lewis, David D., and Catlett, Jason. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the 11th International Conference on Machine Learning (ICML 1994)*, 1994.

Lewis, David D., and Gale, William A. A sequential algorithm for training text classifiers. In Croft, W. Bruce, and Rijsbergen, van Cornelis J., editors, *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 3–12, Dublin, IE, 1994. Springer Verlag, Heidelberg, DE.

Liere, Ray, and Tadepalli, Prasad. Active learning with committees for text categorization. In *Proceedings of the AAAI-97*, Providence, RI, 1997.

Lorenz, Falko. *Lineare Algebra II*. Spektrum Akademischer Verlag, 1996.

Lovász, László, and Vempala, Santosh. Simulated annealing in convex bodies and an $o*(n^4)$ volume algorithm. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, page 650. IEEE Computer Society, 2003. ISBN 0-7695-2040-5.

MacKay, David J. C. *Bayesian Methods for Adaptive Models*. PhD thesis, California Institute of Technology, 1991.

McCallum, Andrew K., and Nigam, Kamal. Employing EM in pool-based active learning for text classification. In Shavlik, Jude W., editor, *Proceedings of the*

Fifteenth International Conference on Machine Learning (ICML 1998), pages 350–358, Madison, US, 1998. Morgan Kaufmann Publishers, San Francisco, US.

Mercer, J. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society, London*, A 209:415–446, 1909.

Michie, Donald, Spiegelhalter, David J., and Taylor, C. C. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, 1994. Data available at `ftp.ncc.up.pt/pub/statlog/`.

Minka, Thomas P. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT Media Lab, 2001.

Mitchell, Tom M. Generalization as search. *Journal of Artificial Intelligence*, 18: 203–226, 1982.

Mitchell, Tom M. *Machine Learning*. McGraw-Hill, New York, 1997.

Muslea, Ion, Minton, Steve, and Knoblock, Craig. Active + semi-supervised learning = robust multi-view learning. In *Proceedings of the Nineteenth International Conference on Machine Learning (ICML 2002)*, pages 435–442, 2002.

Platt, John. Fast training of support vector machines using sequential minimal optimization. In Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods — Support Vector Learning*, pages 185–208, Cambridge, MA, 1999a. MIT Press.

Platt, John. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In Smola, A.J., Bartlett, P., Schoelkopf, B., and Schuurmans, D., editors, *Advances in Large Margin Classifiers*, pages 61–74, Cambridge, MA, 1999b. MIT Press.

Platt, John, Cristianini, Nello, and Shawe-Taylor, John. Large margin dags for multiclass classification. In Solla, S.A., Leen, T.K., and Mueller, K.-R., editors, *Advances in Neural Information Processing Systems 12 (NIPS 1999)*, pages 547–553, 2000.

Riecken, Doug. Personalized views of personalization. *Communications of the ACM*, 43(8):26–28, 2000.

Roy, Nicholas, and McCallum, Andrew. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 441–448. Morgan Kaufmann, San Francisco, CA, 2001.

Ruján, Pal, and Marchand, Mario. Computing the bayes kernel classifier. In Smola, A.J., Bartlett, P.L., Schölkopf, B., and Schuurmans, D., editors, *Advances in Large Margin Classifiers*, pages 329–348, Cambridge, MA, 2000. MIT Press.

Saar-Tsechansky, Maytal, and Provost, Foster. Active sampling for class probability estimation and ranking. *Journal of Machine Learning*, 54(2):153–178, 2004.

Schohn, Greg, and Cohn, David. Less is more: Active learning with support vector machines. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, pages 839–846. Morgan Kaufmann, San Francisco, CA, 2000.

Schölkopf, Bernard, Platt, John, Shawe-Taylor, John, Smola, Alex J., and Williamson, Robert C. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1472, 2001.

Schölkopf, Bernhard. *Support Vector Learning*. R. Oldenbourg Verlag, Munich, 1997.

Schölkopf, Bernhard, Herbrich, Ralf, and Smola, Alex J. A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory and and 5th European Conference on Computational Learning Theory*, pages 416–426. Springer-Verlag, 2001. ISBN 3-540-42343-5.

Schölkopf, Bernhard, and Smola, Alexander J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, 2002.

Seeger, Matthias. Learning with labeled and unlabeled data. Technical report, Edinburgh University, 2001.

Seung, H. Sebastian, Opper, Manfred, and Sompolinski, Haim. Query by committee. In *Proceedings of the Fifth Annual ACM Workshop on Computaional Learning Theory*, pages 287–294, 1992.

Shawe-Taylor, John, and Cristianini, Nello. Further results on the margin distribution. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory (COLT 1999)*, pages 278–285. ACM Press, 1999. ISBN 1-58113-167-4.

Shen, Dan, Zhang, Jie, Su, Jian, and Tan, Guodong Zhou Chew-Lim. Multi-criteria-based active learning for named entity recognition. In *42nd Meeting of the Association of Computational Linguistics (ACL04)*, 2004.

Sommerville, Duncan MacLaren Young. *An Introduction to the Geometry of N Dimensions*. Dover Publications, Inc., New York, NY, 1958.

Spearman, Charles. The proof and measurement of association between two things. *American Journal of Psychology*, 15:72–101, 1904.

Sugiyama, Masashi, and Hidemitsu, . Active learning with model selection — simultaneous optimization of sample points and models for trigonometric polynomial models. *IEICE Transactions on Information and Systems*, E86-D(12):2753–2763, 2003.

Tong, Simon. *Active Learning: Theory and Applications*. PhD thesis, Stanford University, 2001.

Tong, Simon, and Chang, Edward. Support vector machine active learning for image retrieval. In *Proceedings of the Ninth ACM International Conference on Multimedia*, pages 107–118. ACM Press, 2001. ISBN 1-58113-394-4.

Tong, Simon, and Koller, Daphne. Active learning for parameter estimation in bayesian networks. In Leen, Todd K., Dietterich, Thomas G., and Tresp, Volker, editors, *Advances in Neural Information Processing Systems 13 (NIPS 2000)*, pages 647–653. MIT Press, 2001a.

Tong, Simon, and Koller, Daphne. Active learning for structure in Bayesian networks. In *Proceedings of the Seventeenth International Joint Conference on Artificial*

Intelligence (IJCAI 2001), pages 863–869, 2001b.

Tong, Simon, and Koller, Daphne. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001c.

Valiant, Leslie G. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.

Vapnik, Vladimir. *Statistical Learning Theory*. John Wiley, N.Y., 1998.

Vapnik, Vladimir, and Chervonenkis, Alexei. *Theory of Pattern Recognition [in Russian]*. Nauka, Moscow, 1974. (German Translation: W. Wapnik & A. Tscherwonenkis, *Theorie der Zeichenerkennung*, Akademie–Verlag, Berlin, 1979).

von Neumann, John, and Morgenstern, Oskar. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

Warmuth, Manfred K., Liao, Jun, Rätsch, Gunnar, Mathieson, Michael, Putta, Santosh, and Lemmen, Christian. Active learning with support vector machines in the drug discovery process. *Journal of Chemical Information and Computer Sciences*, 43(2):667–673, 2003.

Warmuth, Manfred K., Rätsch, Gunnar, Mathieson, Michael, Liao, Jun, and Lemmen, Christian. Active learning in the drug discovery process. In Dietterich, T.G., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems 14 (NIPS 2001)*, volume 14, pages 1449–1456, 2002.

Weston, Jason, and Watkins, Chris. Support vector machines for multi-class pattern recognition. In *Proceedings of the Seventh European Symposium On Artificial Neural Networks*, pages 219–224, 1999.

Williamson, Robert C., Smola, Alexander J., and Schölkopf, Bernhard. Generalization bounds for regularization networks and support vector machines via entropy numbers of compact operators. *IEEE Transactions on Information Theory*, 47(6):2516–2532, 2001.

Yan, Rong, Yang, Jie, and Hauptmann, Alex G. Automatically labeling data using multi-class active learning. In *Proceedings of the 9th IEEE International Conference on Computer Vision*, pages 516–523, Nice, France, 2003.