University of Innsbruck

Institute of Computer Science

Intelligent and Interactive Systems

On the Missing Value Problem Using Kernels

Chris Wendler
chris.wendler@student.uibk.ac.at

M.Sc. Thesis

Supervisor: Sandor Szedmak
sandor.szedmak@aalto.fi
23rd November 2016

In memory of my father, Jimmy.

Abstract

Machine learning tasks lurk wherever large amounts of data are of concern. Not only computer science applications such as social networks or webshops but also problems occurring in areas like life science or economics give rise for different machine learning tasks such as object classification, item recommendation or the prediction of unknown relationships. Despite the variety of these tasks, their underlying optimization problems are often similar and can be cast as special cases of the missing value problem, in which the missing values of a table are inferred using the observed ones.

This thesis aims to illuminate the theoretical foundations required to understand such problems ranging from the formalization to the solution of the corresponding optimization problems. In order to do so, the application of kernel methods to learning tasks of increasing difficulty, starting with classical and ending with structured-output learning tasks, is investigated. The implicit knowledge given by the data is modeled by a linear operator between Hilbert spaces, in which the input and output data are represented. Utilizing the notion of reproducing kernels, the resulting hypothesis spaces are accessible in an elegant way. Different learning tasks can be characterized by loss functions measuring the quality of a certain hypothesis with respect to the task. Given a loss function and a hypothesis space, a hypothesis is found by regularized risk minimization.

In the end, the previous efforts culminate in a learning framework capable of handling the missing value problem for structured objects. This thesis shows that the application of the kernel trick allows for the solution of various learning tasks in a unified and efficient way.

Acknowledgments

I would first like to thank my thesis supervisor Sandor Szedmak of the Aalto university. Sandor was a very patient supervisor, who spent a lot of time and effort in answering my questions, of which I had a lot. He gave me absolute freedom over the topic and contents of my thesis, which made writing my master thesis a refreshing and challenging experience.

I would also like to thank Senka Krivic for providing me with as many datasets and example tasks as I wanted, and Roswitha Kathrein for proofreading my thesis.

Finally, I must express my gratitude to my family and my girlfriend without whose unconditional support this thesis never would have been completed.

Thank you!

Contents

Nomenclature

$\langle \cdot, \cdot \rangle_{\mathcal{H}}$ inner product in corresponding to the Hilbert space $\mathcal{H}$

$\langle \cdot, \cdot \rangle_{Frobenius}$ the Frobenius inner product

$\gamma(w)$ the generalization of the SVM margin to the structured-output case

$\hat{d}(w, X)$ the margin of the hyperplane parametrized by w with respect to the set of points $X \subset \mathcal{H}$, where $\mathcal{H}$ is a Hilbert space

$\mathcal{H}'$ the topological dual space of a Hilbert space $\mathcal{H}$, which contains linear and continuous forms

$\mathcal{H}_\phi$ Hilbert space corresponding to the feature space mapping ϕ

$\mathcal{S}^\perp$ orthogonal complement of a subspace $\mathcal{S}$ of a Hilbert space

$\mathcal{V} \otimes \mathcal{W}$ the tensor product of the vector spaces $\mathcal{V}$ and $\mathcal{W}$

$\mathcal{X}$ input space

$\mathcal{Y}$ output space

$\mathcal{Z}$ input-output space

$\mathscr{A}$ learning algorithm

$\mathscr{C}_w$ a compatibility function that measures the compatibility of elements of different sets, parametrized by w

$\mathscr{H}$ hypothesis space

$\mathscr{H}_\phi$ $:= \{g : \mathcal{X} \to \mathbb{R} : g = f \circ \phi, f \in \mathcal{H}_\phi^*$ and feature space mapping $\phi\}$

$\mathscr{H}_k$ $:= \{g : \mathcal{X} \to \mathbb{R} : g = \sum_{i=1}^n \alpha_i k(x_i, \cdot),$ for $n \in \mathbb{N}, x_1, \ldots, x_n \in \mathcal{X}, \alpha_1, \ldots \alpha_n \in \mathbb{R}$ and k is a kernel function$\}$

$\mathbf{L}^2(\mathcal{M})$ the space of square-integrable functions defined on the set $\mathcal{M}$

$\mathbf{z}$ training sample

∇f the gradient of the function f

$\phi(\cdot)$ feature space mapping

$\phi(x)$ feature vector of input point x

$\phi_i(x)$ i-th feature of the feature vector of input point x

$B_r(x)$ open ball with radius r around x

$c(\cdot,\cdot)$ loss function

$c_{01}(\cdot,\cdot)$ zero one loss function

c_{hinge} the hinge loss function

$c_{sq}(\cdot,\cdot)$ squared loss function

J a joint kernel function defined on the Cartesian product of several sets

$k_{\mathcal{X}}$ positive definite kernel function defined on the set $\mathcal{X}$

$l^2(\mathbb{K})$ the space of square-summable sequences over the field $\mathbb{K}$

L_x the evaluation functional over a Hilbert space of functions $\mathcal{H}$ for the point $x \in \mathcal{X}$

$R[\cdot]$ risk functional

$R_{emp}[\cdot;\mathbf{z}]$ empirical risk functional with respect to the sample $\mathbf{z}$

$R_{reg}[\cdot;\mathbf{z}]$ regularized empirical risk functional with respect to the sample $\mathbf{z}$

$v \otimes w$ the tensor product of the vectors v and w

$y^*(x)$ the solution of the pre-image problem of a structured-output method for the point x

Linear Algebra

$\langle\cdot,\cdot\rangle$ inner product

$\mathcal{H}$ Hilbert space

$\mathcal{P}_{(w,b)}$ affine hyperplane parametrized by normal vector w and bias b

$d_{(w,b)}(x)$ signed distance between point x and hyperplane $\mathcal{P}_{(w,b)}$

$D_{\mathcal{H}}(\cdot,\cdot)$ the metric induced by the inner product in $\mathcal{H}$

Probability Theory

$\int \cdot d\mu$ Lebesgue integral with respect to the measure μ

$\mathbb{E}_{(\mathcal{X},\mathcal{Y})}[\cdot]$ expected value of a function with respect to the joint input-output probability distribution

$\mathbf{P}_{(\mathcal{X},\mathcal{Y})}$ joint probability distribution on $\mathcal{X} \times \mathcal{Y}$

$\mathbf{P}_{\mathcal{X}}$ a probability measure on $\mathcal{X}$ such that the triple $(\mathcal{X}, \mathscr{X}, \mathbf{P}_{\mathcal{X}})$ is a probability space

$\mathbf{P}_{\mathcal{Z}}$ joint probability distribution on $\mathcal{X} \times \mathcal{Y}$

Abbreviations

i.i.d. independently and identically distributed

MMMVM Maximum Margin Multi Valued Mappings

MMR Maximum Margin Regression

p.d. kernel positive definite kernel

r.k. reproducing kernel

RKHS reproducing kernel Hilbert space

SVM Support Vector Machine

List of Figures

List of Tables

Declaration

By my own signature I declare that I produced this work as the sole author, working independently, and that I did not use any sources and aids other than those referenced in the text. All passages borrowed from external sources, verbatim or by content, are explicitly identified as such.

Signed: ..... Date: .....

Chapter 1

Introduction

In the age of multimedia machine learning has gained popularity, the internet is flooded by data such as images, movies and texts. Despite the flood of data, knowledge still presents a scarce resource. In a certain sense, machine learning aims at closing the gap between data and knowledge. Informally, one could define machine learning as the process of finding ways to understand data, which is in most cases coupled to a task. In this context, the knowledge or the understanding of the data is usually modeled as a function and the task is used to define a performance measure. Therefore, given data and a task, the objective of a machine learning algorithm is to find a function that optimally solves the task with respect to a performance measure. Additionally, the learned function should improve - with respect to the performance measure - with an increasing amount of data available and be able to generalize to unseen data. This definition of machine learning is consistent with the one of Mitchell (1997).

Depending on the type of data and the performance measure three broad categories are distinguished: *supervised*, *unsupervised* and *reinforcement learning* (Bishop, 2006). This thesis is mainly concerned with supervised learning. In supervised learning tasks the objective is to find a functional relationship connecting elements of an input set with elements of an output set, based on a given training sample comprised by input/output pairs. Typical supervised learning tasks are classification and regression, where the input space is an arbitrary set and the output spaces are a discrete set and a metric space (e.g. the real numbers) respectively. If the output space also is a more or less arbitrary set comprised by complex (structured) objects, then according to Bakir et al. (2007), Tsochantaridis et al. (2005b), Nowozin and Lampert (2011) and Weston et al. (2007) we talk about *structured output learning*. Related to the high costs of extensive labeling of large datasets it is of interest to also consider training samples with incomplete labeling, i.e. for some input objects the corresponding output objects are unknown. In that situation we talk about *weakly supervised learning*. One can go even further and omit the distinction between input and output space and learn relations between an arbitrary number of sets, which leads us to the *missing value problem* (Little and Rubin, 1986).

Without much effort it is observable that supervised learning problems can always be cast as weakly supervised learning problems, which can be always cast as missing value problems. Therefore, if the missing value problem was solved, the weakly supervised and the supervised learning problem would be solved as well. If all variables lived in a field, e.g. the real numbers, we would talk about the matrix completion problem. Unfortunately, according to Johnson (1990), the matrix completion problem and consequently the missing value problem are not solvable without further assumptions.

The goal of this thesis is to incrementally develop a framework that is capable of solving the missing value problem under certain assumptions. In order to do so, we first study classification and regression tasks in Chapter 2 and show a possible solution by risk minimization using

suitable input representations, i.e. Hilbert spaces. Secondly, we show that a certain class of hypothesis spaces – more precisely, reproducing kernel Hilbert spaces of real valued functions – can be used implicitly and efficiently by considering so-called kernel functions in Chapters 3 and 4. In Chapter 5 all the previously introduced concepts are combined in order to address the structured output learning task. Chapter 6 illuminates the missing value problem from the relation learning perspective and concludes with a framework capable of addressing the missing value problem, which is demonstrated by its application on an affordance learning dataset. Eventually, Chapter 7 is going to present the conclusion of the thesis.

Chapter 2

Machine Learning

In this chapter the fundamentals of machine learning required to understand the remaining thesis are introduced. For a more extensive overview please have a look at Bishop (2006) and for a more specific self-contained introduction please have a look Herbrich (2001). This chapter is largely based on Herbrich (2001).

2.1 Background

As machine learning is a broad field, in which many research areas overlap, this section will briefly summarize the basic concepts and notations relevant for this thesis.

2.1.1 Types of Problems and Tasks

Machine learning problems are classified into three broad categories by means of the data available and in terms of the nature of the feedback signal, namely:

- *Supervised learning:* Given a sample of input-output pairs, the objective is to find a function mapping any input to an output in order to minimize the disagreement with future input/output observations. The inputs could for instance be images of certain objects and the outputs the class labels of the objects depicted.
- *Unsupervised learning:* Given data sample, the objective is to find the structure underlying the data which for instance could be captured by the probability distribution of the data or simply a more compact representation of the data.
- *Reinforcement learning:* Given a situation, the objective is to find the best action in order to reach a certain goal. In contrast to the supervised learning task, here the optimal action is not available during the training phase. Instead, the learner has to gain information about the quality of actions by the rewards it gets. There are many ways to design different reward functions - particularly, it is not necessary that every action is rewarded individually. To incorporate a variety of reward functions it is common practice to choose the actions that maximize the expected value of the reward function.

Despite the fact that the objectives of the different learning categories seem different at first glance, the underlying task in all of them is to generalize from data.

2.1.2 The Missing Value Problem

Due to the fact that in some cases it is costly to obtain a large sample of annotated data points, it makes sense to consider learning problems that contain data points with missing output values,

which are referred to as *semi-supervised learning* problems. When working with real life data sets it can make sense to even go further and consider a more general type of learning problems. In a more general case the training data could be composed by objects, for which different observations exist. However, for individual objects some of the observations might be missing. This type of learning problem can be considered as a *missing value* problem, where the data is given in the form of a table, with rows that correspond to objects and columns that correspond to certain observations of the objects. The goal is to infer the missing entries of the table.

Figure 2.1 summarizes the differences between the mentioned learning problems. It is observable that the supervised learning problem can be cast as a special case of the semi-supervised learning problem and that the semi-supervised learning problem can be cast as a special case of the missing value problem. Another important specialization of the missing value problem is the matrix completion problem, where the goal is to recover the missing entries of a partially given matrix. In contrast to the missing value problem in which the entries of the table live in arbitrary sets, in the matrix completion problem the entries are typically assumed to be real numbers. The probably most popular example concerning applications relying on the solution of the matrix completion problem are recommender systems, in which the goal is to predict missing ratings in a partially given user-item matrix, for an introduction to recommender systems see Jannach et al. (2010).

This work will be ultimately concerned with the missing value problem which is not solvable in its general version. Fortunately, under certain assumptions about the nature of the missing entries of the table they can be restored. In order to understand the problem properly, it is helpful to study the special cases first.

	Supervised		Semi-supervised		Matrix completion			
	learning schemes							
Training	x_1	y_1	x_1	y_1	z_1^1	$\varnothing$	z_1^3	z_1^4
	x_2	y_2	x_2	y_2	z_2^1	z_2^2	$\varnothing$	z_2^4
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	x_m	y_m	x_m	$\varnothing$	$\varnothing$	z_m^2	z_m^3	z_m^4
Test	x_1	$\varnothing$	x_1	$\varnothing$	z_1^1	z_1^2	$\varnothing$	z_1^4
	x_2	$\varnothing$	x_2	$\varnothing$	$\varnothing$	z_2^2	z_2^3	z_2^4
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	x_m	$\varnothing$	x_m	$\varnothing$	z_m^1	$\varnothing$	z_m^3	$\varnothing$

Figure 2.1: An overview of the different learning problems.

2.1.3 The Learning Task

In the following we will be concerned with the supervised learning problem as we plan to generalize with respect to the missing value problem once we fully understand its specializations. Recall that the supervised learning problem can be formulated as a special case of the missing value problem. In the supervised learning task the goal is to discover a functional relationship between two sets, typically referred to as *input space* $\mathcal{X}$ and *output space* $\mathcal{Y}$.

Definition 1. (Input-Output space) *We call*

- $\mathcal{X}$ *the input space,*

- $\mathcal{Y}$ *the output space,*
- $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$ *the joint input-output space*

of the learning problem.

The learning of a relationship between inputs and outputs is based on the realization of a sample of several input-output pairs, which are assumed to be drawn independently and identically distributed (i.i.d.) from an unknown probability distribution. In machine learning literature the realization of the sample is often directly referred to as the sample. Therefore, we will stick to this terminology unless the context suggests the more precise statistical terminology.

Definition 2. (Training sample) *Given an input-output space $\mathcal{Z}$ and a probability measure $\mathbf{P}_{\mathcal{Z}}$ over $\mathcal{Z}$, we call the m-tuple*

$$\mathbf{z} := (z_1, \ldots, z_m) \in \mathcal{Z}^m, \tag{2.1}$$

drawn i.i.d. from $\mathbf{P}_{\mathcal{Z}}$, a training sample of size m. Additionally we call the pairs $z_i = (x_i, y_i)$ for $i \in \{1, \ldots, m\}$ training examples and define $\mathbf{x}$ as $(x_1, \ldots, x_m)$ and $\mathbf{y}$ as $(y_1, \ldots, y_m)$. We use $\mathbf{z}$ and $(\mathbf{x}, \mathbf{y})$ exchangeable.

To sum up, based on a training sample we aim to learn a functional relation between input and output space. Theoretically, this relation could be any function. Unfortunately, considering all possible functions from $\mathcal{X}$ to $\mathcal{Y}$ would result in an infeasible optimization problem, because $\mathcal{Y}^{\mathcal{X}}$ is simply too large. Therefore, typically only a subspace of $\mathcal{Y}^{\mathcal{X}}$, a so-called *hypothesis space*, is considered.

Definition 3. (Function space) *Let $\mathcal{Y}^{\mathcal{X}}$ denote the set containing all functions from $\mathcal{X}$ to $\mathcal{Y}$*

$$\mathcal{Y}^{\mathcal{X}} := \{f | f : \mathcal{X} \to \mathcal{Y}\}. \tag{2.2}$$

A subset $\mathbb{K}$

$$\mathbb{K} \subseteq \mathcal{Y}^{\mathcal{X}} \tag{2.3}$$

of $\mathcal{Y}^{\mathcal{X}}$ is called function space*. The reason for this nomenclature originates from the fact that in many applications the subset of functions is a topological space, a vector space or both. For example when $\mathcal{Y}$ is a field $\mathcal{Y}^{\mathcal{X}}$ is a vector space.*

Definition 4. (Hypothesis space) *The function space*

$$\mathscr{H} \subseteq \mathcal{Y}^{\mathcal{X}}, \tag{2.4}$$

that is considered when solving an optimization problem, is called hypothesis space and an element $h \in \mathscr{H}$ is called hypothesis.

The above definitions allow for a formulation of a more concise definition of the learning problem:

Definition 5. (Learning problem) *Given an input space $\mathcal{X}$, an output space $\mathcal{Y}$, a training sample $\mathbf{z} = (\mathbf{x}, \mathbf{y}) = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$ of size $m \in \mathbb{N}$ drawn i.i.d. from an unknown distribution $\mathbf{P}_{\mathcal{Z}}$ and a hypothesis space $\mathscr{H}$, the learning problem is to find the unknown functional relation $h : \mathcal{X} \to \mathcal{Y} \in \mathscr{H}$ between objects $x \in \mathcal{X}$ and targets $y \in \mathcal{Y}$ based on the training sample. Depending on the structure of the output space different types of learning problems are distinguished, see Table 2.1 for an overview.*

At this point we did not introduce a methodology to evaluate the quality of a given hypothesis. However, in order to address the learning problem from an optimization point of view this is mandatory. In the next section of this chapter we are going to study classical machine learning problems in order to get an intuition about evaluating the quality of given hypotheses.

Output space $\mathcal{Y}$	Type
finite set	classification learning
ordered space	preference learning
metric space	function learning
contains structured objects	structured output learning

Table 2.1: Types of learning problems based on the structure of the output space.

2.2 Learning Algorithms

A learning algorithm is an algorithm that is intended to solve a learning problem by utilizing data. Additionally, learning algorithms should perform the better the more data is available. The objective of a learning algorithm is the selection of a function from the hypothesis space.

Definition 6. (Learning algorithm) *Given an input space $\mathcal{X}$, an output space $\mathcal{Y}$ and a hypothesis space $\mathscr{H} \subseteq \mathcal{Y}^{\mathcal{X}}$, a learning algorithm $\mathscr{A}$ is a mapping*

$$\mathscr{A} : \bigcup_{n=1}^{\infty} (\mathcal{X} \times \mathcal{Y})^n \to \mathscr{H}. \tag{2.5}$$

So far, it is not clear how the selection of an element of the hypothesis space is performed, however, it is obvious that for a proper selection a quality measure is required. The quality measure is typically partially imposed by the task and partially a design choice. A closer look at the classification and regression problem leads to the analysis of the connection between task and quality measure.

2.2.1 Linear Classification

In this section the basics of linear classifiers will be introduced and their relevance illustrated in an example.

Binary Classification

The simplest classification problem is the binary classification problem. In the following, let $\mathcal{V}$ be a Euclidean vector space over the field of real numbers.

Definition 7. (Binary classification problem) *Given a sample $\mathbf{z} = (\mathbf{x}, \mathbf{y}) = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$ of size $m \in \mathbb{N}$, where the inputs $x_i \in \mathcal{X}$ are elements of an arbitrary set and the target values $y_i \in \{-1, 1\}$ correspond to binary class labels, the objective is to find a function $f : \mathcal{X} \to \mathcal{Y} \in \mathcal{Y}^{\mathcal{X}}$, that for any $x \in \mathcal{X}$ assigns the corresponding class label.*

If the input space $\mathcal{X}$ is a Euclidean vector space, the binary classification problem might be addressed by looking for hyperplanes.

Definition 8. (Linear hyperplane) *A* linear hyperplane *in a d-dimensional vector space $\mathcal{V}$ is a linear subspace of dimension $d-1$ and is characterized by*

$$\mathcal{P}_w := \{x \in \mathcal{V} : \langle w, x \rangle = 0\} \text{ for } w \in \mathcal{V},$$

where $w \in \mathcal{V}$ and $x \in \mathcal{V}$ a d-dimensional vectors and w is referred to as a normal vector of the linear hyperplane.

Definition 9. (Affine hyperplane) *An* affine hyperplane *in a d-dimensional vector space $\mathcal{V}$ is an affine subspace of dimension $d-1$ and is characterized by*

$$\mathcal{P}_{(w,b)} := \{x \in \mathcal{V} : \langle w, x\rangle = b\} \text{ for } w \in \mathcal{V},$$

where $w \in \mathcal{V}$ and $x \in \mathcal{V}$ a d-dimensional vectors and w is referred to as a normal vector of the affine hyperplane.

In the machine learning literature it frequently occurs that affine hyperplanes are referred to as linear hyperplanes.

Remark 10. (Distance from a point to a hyperplane) *The signed distance between a point $v \in \mathcal{V}$ and a hyperplane $\mathcal{P}_{(w,b)}$ is given by the length of the projection of a vector from any point $x_0 \in \mathcal{P}_{(w,b)}$ to v, given by $v - x_0$, onto the normal vector of the hyperplane w*

$$d_{(w,b)} : \mathcal{V} \to \mathbb{R} : x \mapsto \frac{\langle w, x\rangle - b}{\|w\|_2}. \tag{2.6}$$

Every hyperplane naturally separates its corresponding vector space into two subspaces.

Remark 11. (Half-spaces) *In a vector space $\mathcal{V}$ over the field of real numbers an affine hyperplane $\mathcal{P}_w$ separates the space into two half-spaces, which are given by*

$$\mathcal{V}_+ := \{x \in \mathcal{V} : \langle w, x\rangle > b\}$$

and

$$\mathcal{V}_- := \{x \in \mathcal{V} : \langle w, x\rangle < b\},$$

where $w, x \in \mathcal{V}$ and $b \in \mathbb{R}$. A hyperplane separating two classes in a classification scenario is called separating hyperplane.

Therefore, to define a linear classifier it is sufficient to find a hyperplane that separates the input space, in such a way that one half contains all the data points with class label one and the other half contains all data points with label minus one.

Definition 12. (Binary linear classifier) *Given an affine hyperplane $\mathcal{P}_{(w,b)} \subset \mathcal{V}$ a binary linear classifier $h : \mathcal{V} \to \{-1, 1\}$ can be obtained by considering*

$$h(x) := sign(\langle w, x\rangle - b) \text{ for } x \in \mathcal{X},$$

which is equal to one if $x \in \mathcal{V}_+$ and minus one if $x \in \mathcal{V}_-$.

If a hyperplane that agrees with the data sample exists, the sample will be linearly separable.

Definition 13. (Linear separability) *Let $\mathcal{X}$ be a Euclidean vector space. A data-set $\mathbf{z} = (\mathbf{x}, \mathbf{y}) \in (\mathcal{X} \times \{-1, 1\})^m$ is called* linearly separable *if a linear classifier h exists that satisfies*

$$\{(x, y) \in \mathbf{z} : h(x) \neq y\} = \varnothing.$$

Meaning that it correctly classifies each item of the training set.

Multi-class Classification

After having introduced binary linear classifiers, we new have the tools to address the binary classification task. However, in practice often more than two classes are of interest.

Definition 14. (Multi-class classification problem) *Given a sample* $\mathbf{z} = (\mathbf{x}, \mathbf{y}) = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$ *of size* $m \in \mathbb{N}$*, where the inputs* $x_i \in \mathcal{X}$ *can have arbitrary structure and the target values* $y_i \in \{1, \ldots, k\}$ *correspond to class labels, the objective is to find a function* $f : \mathcal{X} \to \mathcal{Y} \in \mathcal{Y}^{\mathcal{X}}$ *that for a* $x \in \mathcal{X}$ *assigns the corresponding class label* $y \in \mathcal{Y}$*.*

In the following example we will motivate the choice of linear classifiers and introduce one way to address the multi-class classification problem.

Example 15. (Classification learning example) *Given a sample* $(\mathbf{x}, \mathbf{y}) = ((x_1, y_1), \ldots, (x_m, y_m))$ *of object-label pairs, where* $\mathcal{Y} = \{1, \ldots, k\}$*, we are looking for a function* $h : \mathcal{X} \to \mathcal{Y}$ *that assigns a class label* $y \in \mathcal{Y}$ *to an object* $x \in \mathcal{X}$*. Ideally,* h *should assign identical class labels to objects that are very similar. When talking about similarity between objects, it is useful to work with metric spaces. In this example* $\mathcal{X}$ *is assumed to be a Euclidean vector space. Arbitrary input spaces can be handled by mapping them into metric spaces. One simple classifier showing the desired behavior is the nearest neighbor classifier*

$$h_{NN} : \mathcal{X} \to \mathcal{Y} : x \mapsto y_{nn}, \text{ where } nn = \underset{i \in \{1,\ldots,m\}}{\arg\min} \|x - x_i\|, \tag{2.7}$$

which assigns the label of the closest training point to the point of interest. Unfortunately using a nearest neighbor classifier requires the storage of the whole training set, which can require a significant amount of storage. Therefore, it would be favorable to use a parametric function to model the classifier in order to overcome this drawback. The simplest parametric functions with the desired behavior are linear ones

$$f(\cdot; w) : \mathcal{X} \to \mathbb{R} : x \mapsto \langle w, x \rangle = w' x. \tag{2.8}$$

The fact that linear functions map similar points to similar function values can be easily derived by considering

$$\begin{aligned} |f(x) - f(\hat{x})| &= |\langle w, x \rangle - \langle w, \hat{x} \rangle| \\ &= |\langle w, x - \hat{x} \rangle| \\ &\leq \|w\| \|x - \hat{x}\|, \end{aligned}$$

where the last inequality is the Cauchy-Schwarz inequality. The difference between the function values evaluated at two points is proportional to the distance between the points with the constant factor $\|w\|$*. A linear binary classifier can be obtained by taking the sign of a linear function*

$$h_{lin}(\cdot; w) : \mathcal{X} \to \mathcal{Y} : x \mapsto sign(f(x; w)).$$

In order to build a classifier for more than two classes, as required in our case, a simple construction is to first learn k *1-vs-all classifiers* $h_1, \ldots, h_k$*, where a positive sign of* $h_i(x)$ *corresponds to "*x *is member of class* i*". The linear functions learned can be used to construct a multiclass classifier*

$$h_{multi} : \mathcal{X} \to \mathcal{Y} : x \mapsto \underset{i \in \{1,\ldots,k\}}{\arg\max} f_i(x).$$

Therefore, using parametric linear classifiers enables to drastically reduce the amount of storage instead of storing the whole training set. That way, only the storage of k *parameter vectors is required, while the property that similar points are mapped to similar class labels is preserved.*

In Chapter 5, a more sophisticated learning framework with the ability to address the multi-class classification problem is introduced.

2.2.2 Feature Space and Hypothesis Space

Unfortunately, real world problems are often more complex. This occurs, for instance, when the data is not linearly separable in the input space or when the input space $\mathcal{X}$ is an arbitrary set without the notion of an inner product and the other nice properties of Euclidean vector spaces. When working with not linearly separable data the classifiers introduced so far are likely to perform poorly. Even more so, if $\mathcal{X}$ is an arbitrary set they might not be able to be used directly. Therefore, it is common to map the input space to a Euclidean space or - more generally - to a Hilbert space. In a Hilbert space, we have an inner product and therefore are able to work with linear forms the same way we did in Euclidean vector spaces.

Definition 16. (Hilbert space) *A Hilbert space is a vector space $\mathcal{H}$ over the field $\mathbb{K}$ together with an inner product $\langle \cdot, \cdot \rangle : \mathcal{H} \times \mathcal{H} \to \mathbb{K}$ that for all $x, y, z \in \mathcal{H}$ and $a \in \mathbb{K}$ satisfies*

1. *Conjugate symmetry:*
$$\langle x, y \rangle = \overline{\langle y, x \rangle}$$
2. *Linearity in the first argument:*
$$\langle ax, y \rangle = a \langle x, y \rangle \text{ and } \langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle$$
3. *Positive-definiteness:*
$$\langle x, x \rangle \geq 0 \text{ and } \langle x, x \rangle = 0 \Rightarrow x = 0.$$

Note that if $\mathbb{K} = \mathbb{R}$ conjugate symmetry is equivalent to symmetry. Thus, the linearity in the first argument implies bilinearity. Additionally a Hilbert space $\mathcal{H}$ is a complete metric space with respect to the metric $D_{\mathcal{H}}$ induced by the inner product

$$D_{\mathcal{H}} : \mathcal{H} \times \mathcal{H} \to [0, \infty) : (x, y) \mapsto \|x - y\| := \sqrt{\langle x - y, x - y \rangle}. \tag{2.9}$$

We don't need the concept of completeness in the scope of this chapter, but it will be detailed in Chapter 4.

Furthermore, the notion of Hilbert spaces allows us to consider diverse hypothesis spaces that are easy to handle, for example the space of polynomials.

Definition 17. (Basis function) *A* basis function *is an element of a basis of a function space. Analogously to the representations of vectors in a vector space in terms of a linear combination of basis vectors, it is possible to represent every function in a function space by a linear combination of the basis functions of that space.*

It is not trivially possible to work with arbitrary shaped data points; still, one way to do so is to perform a mapping of the data into a Hilbert space, referred to as feature space.

Definition 18. (Feature space mapping) *We call a mapping ϕ from the input space $\mathcal{X}$ to a Hilbert space $\mathcal{H}_\phi$ a feature space mapping and $\mathcal{H}_\phi$ a feature space. One way of defining such a mapping is using a set of basis functions $\phi_1, \dots, \phi_i, \dots$ resulting in*

$$\phi : \mathcal{X} \to \mathcal{H}_\phi : x \mapsto (\phi_1(x), \dots, \phi_i(x), \dots)'. \tag{2.10}$$

The feature space can be infinite dimensional which is indicated by the dots $(\phi_1(x), \dots, \phi_i(x), \dots)$. The image $\phi(x)$ of an input $x \in \mathcal{X}$ under ϕ is often referred to as feature vector, of which the components are called features. The term basis function for the component functions of the feature space mapping relates to the fact, that the dual space of the feature space, namely the space of linear forms $f : \mathcal{H}_\phi \to \mathbb{R}$, is isomorphic to the function space spanned by the basis functions. A linear form in the feature space corresponds to a possibly nonlinear function in the input space.

Remark 19. (A new hypothesis space) *Feature space mappings allow us to work with a powerful family hypothesis spaces, namely the ones obtained by considering linear forms from feature spaces to* $\mathbb{R}$

$$\mathcal{H}_\phi^* = \{f : \mathcal{H}_\phi \to \mathbb{R} : f \text{ is linear}\}. \tag{2.11}$$

By the composition of the corresponding feature space mapping and those linear forms we obtain a hypothesis space of possibly non-linear functions

$$\mathscr{H} = \{g : \mathcal{X} \to \mathbb{R} : g = f \circ \phi, f \in \mathcal{H}_\phi^* \text{ and feature space mapping } \phi\}. \tag{2.12}$$

Remark 20. (Convenient notation) *Additionally, the notion of feature space mapping allows us to omit the bias term when working with linear models, since it can be assumed that one of the basis functions* ϕ_i *is equal to one. Therefore,* $\langle w, x\rangle - b$ *can be written as* $\langle\langle \hat{w}, \phi(x)\rangle\rangle$*, where* $\hat{w} := (w', -b)'$ *is the concatenation of the old parameter vector* w *and the bias term and* $\phi(x) := (x, 1)'$*.*

Non-linearly Separable Data

If the input space $\mathcal{X}$ is already a Hilbert space but the given data is not linearly separable, it is possible to improve the separability of the data by wisely choosing a feature mapping. Recall that binary linear classifiers were obtained by considering the signs of linear forms on the input space.

After finding a mapping from the input space to a Hilbert space $\phi : \mathcal{X} \to \mathcal{H}_\phi$ it is possible to define binary linear classifiers in exactly the same fashion

$$h_w : \mathcal{X} \to \{-1, 1\} : x \mapsto sign(\langle w, \phi(x)\rangle_{\mathcal{H}_\phi}) \text{ where } w \in \mathcal{H}_\phi.$$

The pre-image of the separating hyperplane $\mathcal{P}_w$ in the feature space under the feature map ϕ, denoted by $\phi^{-1}(\mathcal{P}_w)$ corresponds to a non-linear decision surface or decision boundary in the input space, where the non-linearity is determined by the choice of the basis functions of the feature mapping.

Definition 21. (Decision surface) *Given an input space* $\mathcal{X}$ *a feature map* ϕ *a binary linear classifier* h_w *and the corresponding separating hyperplane* $\mathcal{P}_w \subset \mathcal{H}_\phi$*, then the pre-image of* $\mathcal{P}_w$ *under the feature map* ϕ

$$\begin{aligned}\phi^{-1}(\mathcal{P}_w) &= \{x \in \mathcal{X} : \phi(x) \in \mathcal{P}_w\} \\ &= \left\{x \in \mathcal{X} : \langle w, \phi(x)\rangle_{\mathcal{H}_\phi} = 0\right\} \\ &= \{x \in \mathcal{X} : h_w(x) = 0\},\end{aligned}$$

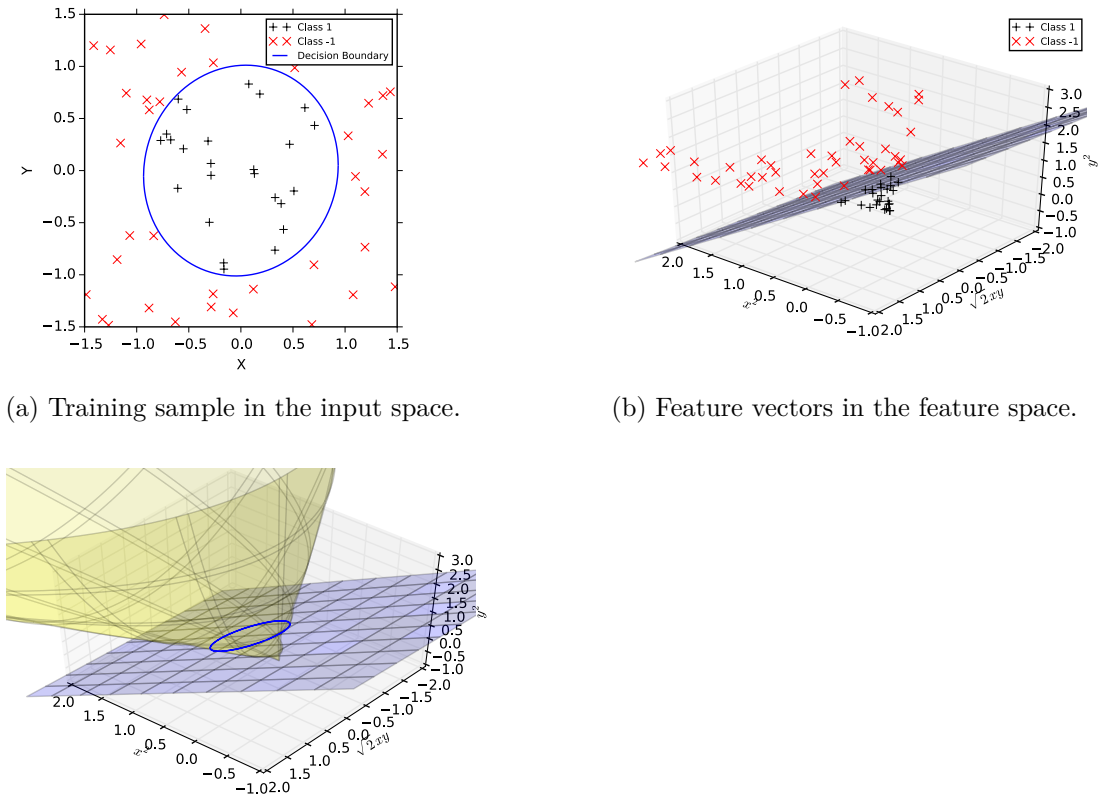
is referred to as decision surface *or* decision boundary*.*

The following example is intended to provide a basic idea about the way in which the choice of basis functions affects the non-linearities used for classification.

Example 22. (Non-linear classification)

Figure 2.2 illustrates a situation as stated previously, where - in the input space - the two classes of points cannot be separated by a linear function. However, they can be separated by a nonlinear function, for example a circle with its center at $(0, 0)$ *and a radius of length one. This observation suggests to use quadratical monomials as non linear basis functions, the feature space mapping given by*

$$\phi : \mathbb{R}^2 \to \mathbb{R}^3 : (x, y) \mapsto (x^2, \sqrt{2}xy, y^2) \tag{2.13}$$



(a) Training sample in the input space.

(b) Feature vectors in the feature space.

(c) The full feature space.

Figure 2.2: Classification of non-linearly separable data by choosing non-linear basis functions. Figure (a) depicts the training sample in the input space, clearly the sample is not linearly separable. Figure (b) depicts feature vectors of the data points, computed by $\phi : \mathbb{R}^2 \to \mathbb{R}^3 : (x, y) \mapsto (x^2, \sqrt{2}xy, y^2)$, and a separating hyperplane. In Figure (c) the image of $\mathbb{R}^2$ under ϕ $\phi(\mathbb{R}^2) \subset \mathbb{R}^3$ is visualized by the yellow cone. Considering planes in $\mathbb{R}^3$ corresponds to considering conic sections in $\mathbb{R}^2$. The conic section corresponding to the separating hyperplane in Figure (b) is the blue ellipse in Figure (a) and (c).

allows to linearly separate the mapped points. An important observation that we can make by considering Figure 2.2 is that the non-linear basis functions chosen for the feature space mapping directly influence the shape of the decision surface, i.e. the inverse image under ϕ of the separating hyperplane, in the input space. For example, if second-degree polynomials are chosen as basis functions the inverse image of the linear hyperplane in the feature space will be a second-degree polynomial surface in the input space. Finding the best non-linear feature mapping to separate the data can be difficult in practice, since a certain degree of prior knowledge is required to support the choice of specific non-linearities. Consequently, it is a common practice to determine an acceptable feature space mapping by trial and error.

Remark 23. (Hypothesis space for classification) *When working with linear classifiers in feature spaces, it is observable that the choice of feature space mapping directly affects the hypothesis space. For a given feature space mapping*

$$\phi : \mathcal{X} \to \mathcal{H}_\phi : x \mapsto (\phi_1(x), \ldots, \phi_n(x))$$

the corresponding hypothesis space $\mathscr{H} \subset \mathcal{Y}^{\mathcal{X}}$ *is*

$$\begin{aligned}\mathscr{H} &= \left\{h \in \mathcal{Y}^{\mathcal{X}} : h(x) = sign(f(x)), x \in \mathcal{X}, f \in \mathcal{H}_\phi^*\right\} \\ &= \left\{h \in \mathcal{Y}^{\mathcal{X}} : h(x) = sign(\langle w, \phi(x)\rangle), x \in \mathcal{X}, w \in \mathcal{H}_\phi\right\} \\ &= \left\{h \in \mathcal{Y}^{\mathcal{X}} : h(x) = sign(\sum_{i=1}^{n} w_i\phi_i(x)), x \in \mathcal{X}, w \in \mathcal{H}_\phi\right\}.\end{aligned}$$

Arbitrary Input Space

If the only requirement for the input space $\mathcal{X}$ is to be a set it will - per definition - not be possible to define a linear form which is a linear function from a vector space to its field of scalars directly on the input space. Therefore, the mapping of the data into a Hilbert space is required in order to work with linear forms or subsequently with linear classifiers. Since linear classifiers are similarity based, one desired property for a feature space mapping $\phi : \mathcal{X} \to \mathcal{H}_\phi$ is that the images of similar objects under the feature mapping are close. If the only information available about the input space is that it is a set, no notion of similarity in the input space will exist. Therefore, in that case it is impossible to quantify the goodness of a corresponding feature space mapping. Fortunately, the objects of interest in practice, for example images, texts, DNA sequences, time series and so on, typically have certain additional structure that enables at least an empirical notion of similarity between them. However, by now it should be observable that the choice of a proper feature space mapping can be tricky.

Additionally, the interpretation of the basis functions ϕ_i as non-linearities that can enhance classification performance cannot be used directly, instead the images of inputs $x \in \mathcal{X}$ under the feature space mapping $\phi(x)$ should be thought of as representers of the inputs. In order to resolve remaining unclarities consider the following example, in which the input space is not a Hilbert space.

Example 24. (String classification) *Let* $\mathcal{X} = \Sigma^*$ *be the set of strings of arbitrary length over the alphabet* Σ*, for more information about strings and substrings we refer to Hopcroft and Ullman (1990). Obviously,* Σ^* *is not a Hilbert space, therefore, in order to use linear classifiers it is necessary to find a feature mapping* ϕ *from* Σ^* *to* $\mathcal{H}_\phi$*. As stated above, it would be desirable if similar strings get mapped to similar representations. Intuitively, two strings are similar if they share common sub-strings. Motivated by that notion of similarity a natural choice of a basis function would be an indicator function for a certain substring*

$$\phi_b : \Sigma^* \to \mathbb{R} : v \mapsto \begin{cases} 1, & \text{if } v \text{ contains } b \\ 0, & \text{else} \end{cases}, \text{ where } b \in \Sigma^*. \tag{2.14}$$

Therefore, for a given lexicon $(b_1, \ldots, b_d)$ *of substrings one possible feature space mapping with the desired property could be*

$$\phi : \Sigma^* \to \mathbb{R}^d : v \mapsto (\phi_{b_1}, \ldots, \phi_{b_d}). \tag{2.15}$$

Of course, there are more sophisticated ways to represent strings, as in Lodhi et al. (2002) where a feature space generated by considering the number of occurrences of all subsequences of length k *weighted by their length is used.*

2.2.3 Learning Linear Classifiers

After defining linear classifiers the only open question remaining is how to find the best one for a given task. Ideally, for a given i.i.d. sample $\mathbf{z} = (\mathbf{x}, \mathbf{y}) = ((x_1, y_1), \ldots, (x_m, y_m))$ we would like

to find not just any classifier, but the best one. For the sake of simplicity let us consider the binary classification scenario for now. Intuitively, it would make sense to consider the classifier with the least miss-classifications as the best one. More generally, one could define different loss functions, which are supposed to quantify the deviation between two elements of the output space $\mathcal{Y}$. Counting the amount of misclassifications corresponds to using the so-called zero-one loss.

Definition 25. (Zero-one loss) *The zero-one loss is defined as*

$$c_{01} : \mathcal{Y} \times \mathcal{Y} \to [0, \infty) : (\hat{y}, y) \mapsto \begin{cases} 0 & \text{if } y = \hat{y} \\ 1 & \text{else} \end{cases} \tag{2.16}$$

and assigns zero loss if y and $\hat{y}$ are the same and one otherwise. It can be also written as an indicator function $c_{01}(\hat{y}, y) = \mathbb{I}_{y \neq \hat{y}}$.

In order to count the amount of misclassifications we need to evaluate the zero-one loss for every training example and to sum up the results. After choosing a hypothesis space $\mathscr{H}$, which is typically chosen implicitly by choosing a feature space mapping ϕ, the binary classification problem reduces to an optimization problem of the form

$$\boxed{\begin{array}{ll} \min & \sum_{i=1}^{m} c(y_i, h(x_i)) \\ \text{w.r.t.} & h \in \mathscr{H}, \end{array}} \tag{2.17}$$

or equivalently

$$\boxed{\begin{array}{ll} \min & \sum_{i=1}^{m} c(y_i, sign(\langle w, \phi(x_i) \rangle)) \\ \text{w.r.t.} & w \in \mathcal{H}_\phi. \end{array}} \tag{2.18}$$

The zero-one loss in the above optimization problem can be substituted with a different loss function if required. A learning algorithm for the binary classification task would return the minimizer $h^* \in \mathscr{H}$ of the above optimization problem. The alert reader might have noticed, that the optimal solution to this optimization problem is not necessarily unique especially when using the zero-one loss and when the sample is separable in the feature space. In the separable case the set of all classifiers that are consistent with the sample is referred to as version space.

Definition 26. *(*Version space*) Given a training sample $\mathbf{z} = (\mathbf{x}, \mathbf{y}) \in (\mathcal{X} \times \mathcal{Y})^m$ and a hypothesis space $\mathscr{H}$ the set*

$$\mathcal{V}_{\mathbf{z}} := \{h \in \mathscr{H} : h(x_i) = y_i, \forall i \in \{1, \ldots, m\}\}$$

is called version space.

Later we will see that by making further assumptions it is possible to select the best solution from the version space.

2.2.4 Linear Regression

In contrast to classification learning, where the output space has no structure, in function learning the output space $\mathcal{Y}$ is a metric space. In simplest case the output space is the space of real numbers $\mathcal{Y} = \mathbb{R}$ and the task of finding the functional relationship between the input space and $\mathbb{R}$ is called regression.

When considering only linear functions we talk about linear regression, again non-linearities can be added by utilizing the notion of feature spaces. Therefore, strictly speaking the functions

of interest are only linear in the feature space. More precisely, given a feature space mapping ϕ, the hypothesis space $\mathscr{H}$ is the space of linear forms from $\mathcal{H}_\phi$ to $\mathbb{R}$, also referred to as dual space $\mathcal{H}_\phi^*$ of $\mathcal{H}_\phi$. Traditionally, the loss function used for regression is the squared loss.

Definition 27. (Squared loss) *The squared loss is defined as*

$$c_{sq} : \mathcal{Y} \times \mathcal{Y} \to [0, \infty) : (y, \hat{y}) \mapsto \|y - \hat{y}\|_2^2,$$

where $\|x\|_2^2$ *is defined as* x^2 *for* $x \in \mathbb{R}$.

This particular choice of loss function can be motivated probabilistically. In the classical regression model the i'th observation is assumed to have the following form

$$y_i = wx_i + b + \epsilon_i, \tag{2.19}$$

where $i \in \{1, \dots, n\}$ and ϵ_i is the realization of a normally, independently and identically distributed sample $E_1, \dots, E_n$, with $\mathbb{E}[E_1] = 0$ and $Var[E_1] = \sigma^2$. Therefore, the random variables $Y_i = wx_i + b + E_i$ are distributed normally $Y_i \sim \mathcal{N}(wx_i + b, \sigma^2)$ with the density

$$f(y; wx_i - b, \sigma^2) = \frac{1}{\sqrt{2\phi\sigma^2}} exp\left(-\frac{1}{2}\left(\frac{y - wx_i - b}{\sigma}\right)^2\right) \tag{2.20}$$

for $i \in \{1, \dots, n\}$. In statistics a common practice for parameter estimation is to maximize the likelihood function. Since our random variables are independent the likelihood function $L(w, b)$ is

$$\begin{aligned} L(w, b) &= \prod_{i=1}^{n} f(y_i; wx_i - b, \sigma^2) \\ &= \left(\frac{1}{\sqrt{2\phi\sigma^2}}\right)^n exp\left(\sum_{i=1}^{n} -\frac{1}{2}\left(\frac{y_i - wx_i - b}{\sigma}\right)^2\right). \end{aligned} \tag{2.21}$$

When working with normal distributions the maximization of the likelihood function can be simplified by considering the logarithm of the likelihood function

$$ln(L(w, b)) = -\frac{n}{2} ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - wx_i - b)^2 . \tag{2.22}$$

The maximization of $ln(L(w, b))$ with respect to w and b is achieved, when the sum of squared losses $\sum_{i=1}^{n} (y_i - wx_i - b)^2$ is minimized. Therefore, after choosing a reasonable feature space $\mathcal{H}_\phi$, the optimization problem to solve becomes

$$\boxed{\begin{array}{ll} \min & \sum_{i=1}^{m} c_{sq}(y_i, \langle w, \phi(x_i)\rangle) \\ \text{w.r.t.} & w \in \mathcal{H}_\phi. \end{array}} \tag{2.23}$$

Since the squared loss function is differentiable and convex, the sum of convex functions is convex and the sum of differentiable functions is differentiable, a closed form solution for the linear regression problem can be obtained by setting the derivative of the objective function to zero.

Example 28. (Linear regression with a straight line) *Let's find the optimal weight vector for a simple example, where* $\mathcal{X}$ *is* $\mathbb{R}$, $\mathcal{Y}$ *is* $\mathbb{R}$ *and* $\phi(x) = (1, x)'$. *Given an i.i.d. sample of observations* $(x_i, y_i)_{i=1}^m \in (\mathbb{R} \times \mathbb{R})^m$, *we are looking for the linear function that minimizes the sum of squares error* E

$$\min_{w \in \mathcal{H}_\phi} E(w_1, w_2) := \frac{1}{2} \sum_{i=1}^{m} (w_1 + w_2 x_i - y_i)^2, \tag{2.24}$$

where we added the factor $\frac{1}{2}$ *to make the solution prettier. Therefore, we set the derivatives of the error function* E *with respect to* w_1

$$\frac{\partial E}{\partial w_1}(w_1, w_2) = \sum_{i=1}^{m} (w_1 + w_2 x_i - y_i) \stackrel{!}{=} 0 \tag{2.25}$$

and to w_2

$$\frac{\partial E}{\partial w_2}(w_1, w_2) = \sum_{i=1}^{m} (w_1 + w_2 x_i - y_i) x_i \stackrel{!}{=} 0 \tag{2.26}$$

to zero. From Equation 2.25 we obtain

$$w_1 = \frac{1}{m} \sum_{i=1}^{m} y_i - w_2 x_i, \tag{2.27}$$

by splitting up the sum into $\sum_{i=1}^{m} w_1 + \sum_{i=1}^{m} (w_2 x_i - y_i)$ *and solving for* w_1*. Let's denote the average* $\frac{1}{m} x_i$ *by* $\bar{x}$ *and* $\frac{1}{m} y_i$ *by* $\bar{y}$*. Substituting Equation 2.27 into Equation 2.26, yields*

$$\begin{aligned} 0 &= \sum_{i=1}^{m} (\bar{y} - w_2 \bar{x} + w_2 x_i - y_i) x_i \\ &= w_2 \sum_{i=1}^{m} (x_i - \bar{x}) x_i + \sum_{i=1}^{m} (\bar{y} - y_i) x_i, \end{aligned} \tag{2.28}$$

which is equivalent to

$$w_2 = \frac{\sum_{i=1}^{m} (y_i - \bar{y}) x_i}{\sum_{i=1}^{m} (x_i - \bar{x}) x_i}. \tag{2.29}$$

If our goal was only the determination of the optimal parameters we would be done here. However, with slight refinements of this expression a meaningful representation can be derived. In order to do so, let us consider the enumerator and the denominator individually. The enumerator can be rewritten by first adding and subtracting $\bar{x} y_i$ *to every summand*

$$\sum_{i=1}^{m} (y_i x_i - \bar{y} x_i) = \sum_{i=1}^{m} (y_i x_i - \bar{y} x_i - \bar{x} y_i + \bar{x} y_i). \tag{2.30}$$

By pulling $\bar{x}$ *and* $\bar{y}$ *out of the sums and utilizing* $\bar{x} = \frac{1}{m} x_i$ *and* $\bar{y} = \frac{1}{m} y_i$

$$\sum_{i=1}^{m} y_i x_i - m\bar{y}\bar{x} - m\bar{x}\bar{y} + m\bar{x}\bar{y} = \sum_{i=1}^{m} y_i x_i - \sum_{i=1}^{m} \bar{y}\bar{x} - \sum_{i=1}^{m} \bar{x}\bar{y} + \sum_{i=1}^{m} \bar{x}\bar{y} \tag{2.31}$$

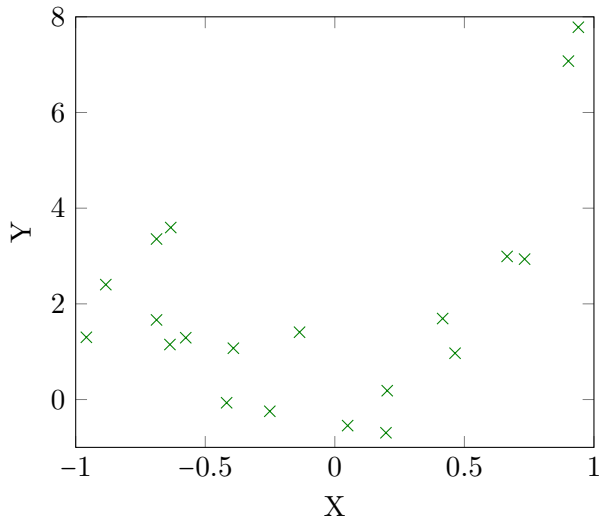
is obtained, which is equivalent to

$$\sum_{i=1}^{m} (y_i x_i - \bar{y} x_i) = \sum_{i=1}^{m} (x_i - \bar{x})(y_i - \bar{y}) \tag{2.32}$$

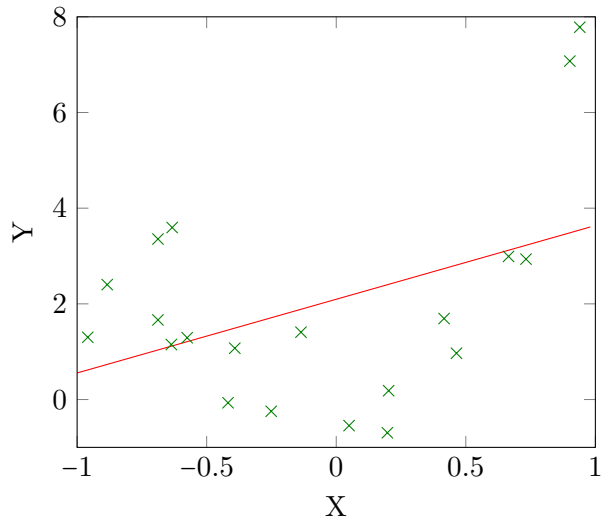
Analogously the denominator can be transformed into $\sum_{i=1}^{m} (x_i - \bar{x})^2$*. Therefore we get the following expression for* w_2

$$w_2 = \frac{\sum_{i=1}^{m} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{m} (x_i - \bar{x})^2}, \tag{2.33}$$

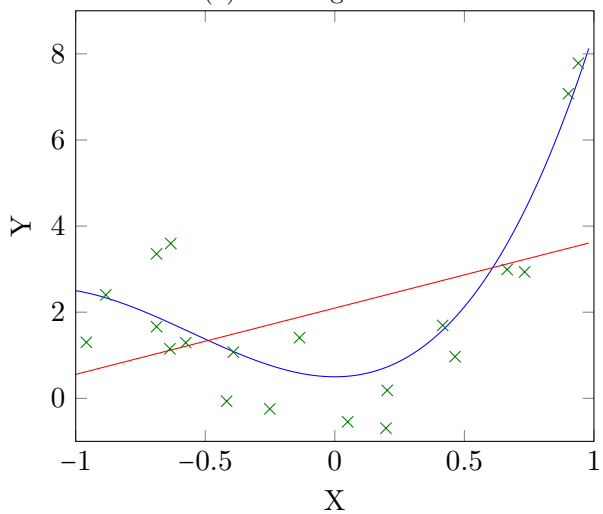
which is closely related to the Pearson correlation, see Pearson (1895) for further details. The expression in the enumerator of w_2 *is called empirical covariance, since the denominator is always positive, the empirical covariance alone determines the sign of the slope of the regression line. Figure 2.3 shows the type of regression line we just derived in a toy example. The training data was generated by perturbing point evaluations of a polynomial function with normally distributed noise in the target component. The Figure contains plots of the training data, the regression line and also the ground truth polynomial.*



(a) Training data



(b) Regression line



(c) Ground truth

Figure 2.3: Linear regression using a line. In the upper left corner there is the training data, in the upper right corner the minimizer of the least squares error (red) and in the lower left corner there is the function (blue) used to generate the training data. The training data was generated by evaluating a polynomial function and adding Gaussian noise.

It's observable that the regression line in Figure 2.3 (b) does not look like a very good estimate of the polynomial function. In order to fit the underlying polynomial function better it could be beneficial to increase the degree of the regression polynomial that we want to fit. Instead of a polynomial of degree one, which is a line, polynomials of arbitrary degree can be utilized by adjusting the feature space mapping accordingly.

Example 29. (Linear regression with a polynomial) *When $\mathcal{X}$ is $\mathbb{R}$ and $\mathcal{Y}$ is $\mathbb{R}$ like in the previous example it is sufficient to change the feature mapping ϕ to*

$$\phi : \mathbb{R} \to \mathbb{R}^{(n+1)} : x \mapsto (x^0, x^1, \ldots, x^n) \tag{2.34}$$

in order to use polynomials of degree n. The resulting optimization problem is

$$\min_{w \in \mathcal{H}_\phi} E(w) := \frac{1}{2} \sum_{i=1}^{m} (y_i - \sum_{j=0}^{n} w_j x_i^j)^2 \tag{2.35}$$

and can be solved analogously by setting the derivative with respect to w equal to zero. Figure 2.4 illustrates regression polynomials of different degrees. With an increasing degree the regression polynomial converges closer and closer to the training points, in other words, the least squares error becomes smaller the higher the degree of the regression polynomial gets. Nevertheless, the deviation between ground truth and regression polynomials obviously increases with the degree of the polynomials considered. This problem is called overfitting. In the next section we shall see one possibility to deal with overfitting.

2.3 Risk Minimization

As we have seen in the classification and regression task, different trains of thoughts lead us to almost identically looking optimization problems. Considering the resulting optimization problems of both tasks, the only observable differences are to be found in the choice of hypothesis space and loss functions. In this chapter we are going to introduce a framework that is capable of handling all supervised learning problems, namely, the risk minimization framework. So far, we have already worked with several loss functions without describing explicitly what a loss function should look like in general. Clearly, it should be possible to interpret the loss function as a measure of discrepancy in the output space. In further consequence, a loss function should allow us to determine the quality of a prediction. The higher the value of the loss function evaluated at a predicted output and the corresponding true output the worse the quality of the prediction.

Definition 30. (Loss function) *Let $\mathcal{Y}$ be the output space, then a loss function is a function that assigns a positive real number to every pair of output values*

$$c : \mathcal{Y} \times \mathcal{Y} \to [0, \infty) : (\hat{y}, y) \mapsto c(\hat{y}, y). \tag{2.36}$$

We interpret the first input variable $\hat{y}$ as the predicted value and the second as the true value. The loss function is intended to measure the discrepancy between the predicted and the true value. Therefore, for two elements $\hat{y}, y \in \mathcal{Y}$ the loss $c(\hat{y}, y)$ should be zero if $\hat{y} = y$ and greater than zero otherwise.

Remark 31. (Practical loss function) *Sometimes it makes sense to loosen the definition of loss function in order to be more flexible in its design. For instance, in linear classification it might be useful to consider the exact output of $\langle w, \phi(x) \rangle$, which is proportional to the distance between the separating hyperplane and the point of interest $\phi(x)$, instead of only its sign.*

If $\mathbf{P}_\mathcal{Z}$ was known, the expected value of a cost function $\mathbb{E}_\mathcal{Z}[c(h(\cdot), \cdot)]$ would be a meaningful measurement for the overall risk of a given hypothesis h. This observation brings us to the definition of a so-called risk functional.

Definition 32. (Risk functional) *Given a loss function $c : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ and the joint probability distribution $\mathbf{P}_\mathcal{Z} = \mathbf{P}_{(\mathcal{X}, \mathcal{Y})}$ of inputs and outputs, the risk functional is given by the expected value of the loss function*

$$R : \mathcal{Y}^\mathcal{X} \to [0, \infty) : f \mapsto \mathbb{E}_{(\mathcal{X}, \mathcal{Y})}[c(f(\cdot), \cdot)] = \int_{\mathcal{X} \times \mathcal{Y}} c(f(x), y) d\mathbf{P}_{(\mathcal{X}, \mathcal{Y})}(x, y), \tag{2.37}$$

where $\int \cdot d\mu$ denotes the Lebesgue integral with respect to the measure μ. In this case μ is defined as the joint probability distribution of the input-output space $\mathbf{P}_{(\mathcal{X},\mathcal{Y})}$. For the construction and properties of the Lebesgue integral we refer to Geiss and Geiss (2014). Note that for evaluations of functionals we use square brackets $R[f]$.

After choosing a hypothesis space and a cost function, learning reduces to an optimization problem of the form

$$\boxed{\begin{array}{ll} \min & R[h] \\ \text{w.r.t.} & h \in \mathscr{H}. \end{array}} \tag{2.38}$$

Unfortunately, this elegant approach is not directly applicable in most real world scenarios, since the probability distribution of the data is typically unknown. Instead, the only information available is an independent and identically distributed sample of the form $((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$.

2.3.1 Empirical Risk Minimization

The question that demands to be answered now is how to estimate $\mathbf{P}_{(\mathcal{X},\mathcal{Y})}$ given an i.i.d. sample $((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$. For simplicity reasons let's assume that $\mathcal{X} \subseteq \mathbb{R}^d$ and that $\mathcal{Y} \subseteq \mathbb{R}^k$. If the sample is sufficiently large the obvious answer to that question will be to use the empirical distribution

$$\mathbf{P}_m(\cdot; (\mathbf{x}, \mathbf{y})) : \mathcal{B}(\mathbb{R}^d \times \mathbb{R}^k) \to [0,1] : (A, B) \mapsto \frac{1}{m} \sum_{i=1}^{m} \delta_{(x_i, y_i)}(A, B), \tag{2.39}$$

where

$$\delta_{(\hat{x},\hat{y})}(A, B) = \begin{cases} 1 & \text{if } \hat{x} \in A \text{ and } \hat{y} \in B \\ 0 & \text{else} \end{cases} \tag{2.40}$$

is a Dirac measure, $(A, B) \in \mathcal{B}(\mathbb{R}^d \times \mathbb{R}^k)$ are Borel sets, i.e. sets that can be formed from open sets through countable unions and intersections, and $\mathcal{B}(\mathbb{R}^d \times \mathbb{R}^k)$ denotes the Borel σ-algebra of $\mathbb{R}^d \times \mathbb{R}^k$, which is the smallest σ-algebra containing all open sets in $\mathbb{R}^d \times \mathbb{R}^k$.

According to the Glivenko-Cantelli theorem, see Glivenko and Cantelli (1933), the empirical distribution converges to the real probability distribution with an increasing amount of sample items almost certainly. The convergence towards the real probability distribution suggests that for large training sets the empirical distribution is a good estimate of the real distribution, which motivates the definition of the empirical risk functional. By estimating the joint probability distribution with the empirical distribution, the integral of the cost function reduces to a sum of cost function evaluations

$$\begin{aligned} R_{emp}[f; \mathbf{z}] := \mathbb{E}_m[c(f(\cdot), \cdot)] &= \int_{\mathcal{X}\times\mathcal{Y}} c(f(x), y) d\mathbf{P}_m(x, y; (\mathbf{x}, \mathbf{y})) \\ &= \int_{\mathcal{X}\times\mathcal{Y}} c(f(x), y) d(\frac{1}{m} \sum_{i=1}^{m} \delta_{(x_i, y_i)}(x, y)) && \text{(by definition of } \mathbf{P}_m) \\ &= \frac{1}{m} \sum_{i=1}^{m} \int_{\mathcal{X}\times\mathcal{Y}} c(f(x), y) d\delta_{(x_i, y_i)}(x, y) && \text{(by definition of } \textstyle\int) \\ &= \frac{1}{m} \sum_{i=1}^{m} c(f(x_i), y_i) && \text{(property of the Dirac measure).} \end{aligned} \tag{2.41}$$

This leads to the following definition of the empirical risk functional.

Definition 33. (Empirical risk functional) *Given a loss function* $c : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ *and a training sample* $\mathbf{z} \in (\mathcal{X} \times \mathcal{Y})^m$ *the empirical risk functional is given by*

$$R_{emp}[\cdot; \mathbf{z}] : \mathcal{Y}^{\mathcal{X}} \to [0, \infty) : f \mapsto \frac{1}{m} \sum_{i=1}^{m} c(f(x_i), y_i). \tag{2.42}$$

Therefore, in practice the learning problem leads to the following optimization problem

$$\boxed{\begin{array}{ll} \min & R_{emp}[h; \mathbf{z}] \\ \text{w.r.t.} & h \in \mathscr{H}, \end{array}} \tag{2.43}$$

which captures all classification and regression examples that we have seen so far. Unfortunately, the empirical risk minimization problem is ill-posed when the hypothesis space $\mathscr{H}$ is sufficiently large. Before we continue let's have a brief look at Hadamard's definition of ill-posedness.

Definition 34. (Well-posed and ill-posed problems) *Hadamard (1902) characterizes a mathematical problem as well-posed if it satisfies the following properties:*

- *A solution exists*
- *The solution is unique*
- *The solution is stable, i.e. the solution's behavior changes continuously with the initial conditions.*

Consequently, a mathematical problem is ill-posed if it is not well-posed.

In order to get an intuition about the meaning of ill-posed optimization problems please recall the classification and polynomial regression example having been discussed earlier. The optimization problem obtained in the binary classification example using the zero-one loss is ill-posed because its solution is not unique. In the linearly separable case infinitely many separating hyperplanes exist. In the polynomial regression example the solutions become unstable when allowing polynomials of larger degrees, in between the training points they are wiggly and at the training points very accurate, meaning that slight variations in $\mathbf{z}$ would result in significantly different solutions of the optimization problem. Figure 2.5 highlights that issue, in which polynomial regression was performed on two slightly different training sets.

Note that stability would be a particularly desirable property for a machine learning algorithm, since in real world applications most measurements are perturbed. Using an unstable algorithm the ability to generalize to unseen data is not given. The polynomials of lower degree improved the stability of the optimization problem, meaning that the solutions behave more smoothly between observations, see Figure 2.4. This observation suggests that there is a connection between well-posedness and the ability to generalize.

Definition 35. (Generalization and consistency) *According to Poggio et al. (2004) an algorithm* $\mathscr{A}$ *will generalize if the function selected by it* $f^* = \mathscr{A}(\mathbf{z})$ *satisfies for all training sets* $\mathbf{z}$ *of size* n *and uniformly for any probability distribution* μ

$$lim_{n \to \infty} |R[f^*] - R_{emp}[f^*, \mathbf{z}]| = 0 \text{ in probability.} \tag{2.44}$$

Poggio et al. (2004) prove a *necessary and sufficient condition for generalization*, namely, if the hypothesis space $\mathscr{H}$ is a uniform Glivenko-Cantelli class, the empirical risk minimization will generalize. Without going into further details about this important theoretical result from learning theory, for us the take-home message is that the empirical risk minimization is not doomed to failure, as long as the hypothesis space is selected or restrained properly. The hypothesis space can be restrained by adding a regularizing term to the objective function.

2.3.2 Regularization

If the hypothesis space in empirical risk minimization is sufficiently discriminative, an unavoidable problem that occurs is the problem of overfitting. In order to fully understand the magnitude of this problem, think of a training sample of the following shape

$$((x_1, y_1), \dots, (x_m, y_m)) \in (\mathbb{R} \times \mathbb{R})^m \text{ with } x_i \neq x_j \text{ for } i \neq j \in \{1, \dots, m\}. \tag{2.45}$$

If the hypothesis space contains polynomials of degree $m-1$, it will contain at least one function that minimizes every reasonable loss function evaluated at the training set, namely, the interpolating polynomial given by the Lagrange formula

$$p(x) := \sum_{i=1}^{m} y_i \prod_{k=1, k\neq i}^{m} \frac{x - x_k}{x_i - x_k}. \tag{2.46}$$

However, polynomials of high degree are known to be rather poor interpolants in terms of their behavior between interpolated points, where they are typically wiggly. Additionally, a small change in the training sample can have a big impact on the interpolating function. Similarly, if the hypothesis space is complex enough there will always be a minimizer strongly dependent on the training sample. When doing interpolation one way around this is to consider more sophisticated interpolation methods like the spline interpolation.

Another more practical way in machine learning is the method of regularization, i.e. to constrain the solution to be less complex. Remember that in machine learning problems we don't want to interpolate, instead, we want to learn a function that generalizes to the whole input space. When learning polynomials this would mean to prefer polynomials of smaller degree. More generally, instead of minimizing the empirical risk functional, given by Equation 2.41, the regularized risk functional is minimized.

Definition 36. (Regularized risk functional) *Given a hypothesis space $\mathscr{H}$ and a training sample $\mathbf{z} \in (\mathcal{X} \times \mathcal{Y})^m$ the regularized risk functional is given by*

$$R_{reg}[.; \mathbf{z}] : \mathscr{H} \to [0, \infty) : f \mapsto R_{emp}[f; \mathbf{z}] + \lambda \Omega(\|f\|_{\mathscr{H}}),$$

where $\lambda \in [0, \infty)$ can be thought of as a trade-off parameter that controls the impact of the regularization functional $\Omega \circ \|\cdot\|_{\mathscr{H}} : \mathscr{H} \to \mathbb{R}$, where Ω is a strictly monotonic increasing function. The idea of the regularization is to restrict the space of solutions to a compact subset of the hypothesis space. Therefore, the essential requirement for any regularization functional Ω is that $\{f \in \mathscr{H} : \Omega(\|f\|_{\mathscr{H}}) \leq \epsilon\} \subseteq \mathscr{H}$ is compact for each positive number $\epsilon > 0$, see Herbrich (2001). When using $\Omega(\|f\|_{\mathscr{H}}) = \|f\|^2_{\mathscr{H}}$ we talk about the well-known Tikhonov regularization introduced by Tikhonov and Arsenin (1977).

Resulting in the following optimization problem

$$\boxed{\begin{array}{ll} \min & R_{reg}[f, \mathbf{z}] \\ \text{w.r.t.} & f \in \mathscr{H}. \end{array}} \tag{2.47}$$

As a concluding example of this section the so-called Tikhonov regularization is applied to the linear regression task using polynomial basis functions.

Example 37. (Regularized polynomial regression) *Recall that despite decreasing the empirical risk, increasing the degree of the regression polynomial resulted in rather poor regression polynomials. When considering Figure 2.4 it is observable that with an increasing degree the regression*

polynomial wiggles between the training instances. Let's examine how the extension of the objective function by a regularizing term affects the solution of the regression problem. Let $\mathcal{X}$ *be* $\mathbb{R}$, $\mathcal{Y}$ *be* $\mathbb{R}$ *and the feature space mapping* $\phi(x)$ *be* $(1, x, x^2, \ldots, x^d)$ *like in the previous example. Hypotheses can be represented by an inner product in the feature space* $f_w(\cdot) := \langle w, \cdot \rangle$. *For regression one of the most popular forms of regularization is the so-called Tikhonov regularization, named after Tikhonov and Arsenin (1977), which is also known as ridge regression in statistics. In Tikhonov regularization the regularizing term takes the form*

$$\Omega \circ \|\Gamma \cdot\|_2 : \mathscr{H} \to [0, \infty) : f_w \mapsto \|\Gamma w\|_2^2, \tag{2.48}$$

where the squared Euclidean norm of the parameter vector w, *transformed by the so-called Tikhonov matrix* Γ, *is computed. Originally the Tikhonov regularization objective function takes the following form*

$$\boxed{\begin{array}{ll} \min & \frac{1}{2}\sum_{i=1}^{m}(y_i - \langle w, \phi(x_i)\rangle) + \|\Gamma w\|_2^2 \\ w.r.t. & f_w \in \mathscr{H}, \end{array}} \tag{2.49}$$

where the squared loss is used as loss function and the regularizing term is simply added to the expected risk. For the sake of simplicity we consider diagonal matrices of the form $\Gamma = \lambda I_d$ *in this example. This is referred to as* l^2*-regularization in the literature. Using the absolute homogeneity of the norm and considering only* $\lambda > 0$ *the objective function becomes*

$$\boxed{\begin{array}{ll} \min & \frac{1}{2}\sum_{i=1}^{m}(y_i - \sum_{j=0}^{d} w_j x_i^j)^2 + \lambda\|w\|_2^2 \\ w.r.t. & f_w \in \mathscr{H}, \end{array}} \tag{2.50}$$

where the regularization term prefers weight vectors with low coefficients or in other words polynomials of low degree. The factor λ *can be thought of as a trade-off parameter, which steers the amount of regularization. Obviously, the old objective function can be restored by setting* $\lambda = 0$ *and the larger* λ *becomes the less is the relevance of the training data. Figure 2.6 depicts how different choices of lambda influence the solution of the optimization problem. It is observable that the solutions obtained with reasonable choices of* λ, *- see Figure 2.6 (b) and (c) - generalize better to unseen data points than the solution obtained without regularization, Figure 2.6 (a). Applied in real world problems the trade-off parameter* λ *can be estimated by cross-validation.*

The ridge regression example empirically showed that regularization can improve the quality of the solution by restricting the hypothesis space in a proper way.

2.4 The Support Vector Machine

The probably most important machine learning method for classification is the Support Vector Machine (SVM) introduced by Cortes and Vapnik (1995). Recall that the classification problem using the zero-one loss is ill-posed, since there are infinitely many indistinguishable solutions in the linear separable case. We called the set of all classifiers agreeing with the training set the version space. The question is which hypothesis to select from the version space. To find an answer to this question consider Figure 2.7, where a subset of the version space is illustrated for an example dataset. Based on the zero-one loss all the hypotheses are the same, despite the fact that we would probably choose one of them with a large margin to the training instances of both classes. Cortes and Vapnik (1995) utilized that simple idea, referred to as maximum margin principle, to determine which hypothesis in the version space is the best one. To sum up, the solution with the maximum margin to the instances of both classes is assumed to be the best one. We define the margin for a given training set the following way.

Definition 38. (Margin) *For a set of points $X := \{x_1, \dots, x_n\}$ living in a Hilbert space $\mathcal{H}$ and a hyperplane $\mathcal{P}_{(w,b)} \subset \mathcal{H}$ the margin $\hat{d}$ is defined as the distance from the hyperplane to it's closest point $x \in X$*

$$\hat{d}(w, X) := \min_{x \in X} \frac{|\langle w, x \rangle|}{\|w\|_2}, \tag{2.51}$$

where $\frac{|\langle w,x \rangle|}{\|w\|_2}$ is the projection of x to the normal vector of the hyperplane w.

2.4.1 Linearly Separable Case

Let's for the sake of simplicity assume that the training set $\mathbf{z} = (\mathbf{x}, \mathbf{y})$ of size n is linearly separable in the feature space $\mathcal{H}_\phi$ induced by ϕ and that $\mathcal{Y}$ is $\{-1, 1\}$. In the linearly separable case the inequation $y_i \langle w, \phi(x_i) \rangle \geq 0$ holds for all $i \in \{1, \dots, n\}$. Therefore, we can get rid of the modulus in $|\langle w, \phi(x_i) \rangle|$ by multiplying $\langle w, \phi(x_i) \rangle$ with y_i. The problem of finding the separating hyperplane with the largest margin can be written as a constrained optimization problem

$$\boxed{\begin{array}{ll} \max_w & \min_i \frac{y_i \langle w, \phi(x_i) \rangle}{\|w\|_2} \\ \text{w.r.t.} & w \in \mathcal{H}_\phi, x_i \in \mathbf{x}, \text{ for } i \in \{1, \dots, n\} \\ \text{s.t.} & y_i \langle w, \phi(x_i) \rangle \geq 0, \text{ for } i \in \{1, \dots, n\}, \end{array}} \tag{2.52}$$

where the objective is to maximize the margin between the hyperplane and the training set, in such a way that there is no disagreement. At first glance, the optimization problem given by Equation 2.52 seems hard, since for every choice of w the closest training point to the hyperplane might be different.

The problem can be significantly simplified by defining $y_i \langle w, \phi(x_i) \rangle$ as one for points $\phi(x) \in \mathcal{H}_\phi$ that lie on the boundaries of the margin. Figure 2.8 visualizes the resulting situation. As a consequence for all training points x_i, with $i \in \{1, \dots, n\}$, the equation $y_i \langle w, \phi(x_i) \rangle \geq 1$ is satisfied. This can be achieved without loss of generality, since it is always possible to adjust the feature space mapping in such a way that the boundary equations are satisfied. Therefore, the optimization problem to solve changes to

$$\boxed{\begin{array}{ll} \max & \frac{1}{\|w\|_2} \\ \text{w.r.t.} & w \in \mathcal{H}_\phi, \\ \text{s.t.} & y_i \langle w, \phi(x_i) \rangle \geq 1, \text{ for } i \in \{1, \dots, n\}, \end{array}} \tag{2.53}$$

which is equivalent to

$$\boxed{\begin{array}{ll} \min & \frac{1}{2} \|w\|_2^2 \\ \text{w.r.t.} & w \in \mathcal{H}_\phi, \\ \text{s.t.} & y_i \langle w, \phi(x_i) \rangle \geq 1, \text{ for } i \in \{1, \dots, n\}. \end{array}} \tag{2.54}$$

The resulting optimization problem is a quadratic optimization problem with linear constraints and is solvable with the use of so-called Lagrangian multipliers, which are introduced in Appendix 2.A. Before considering the Lagrangian dual problem let's think about the non-linearly separable case.

2.4.2 Non-linearly Separable Case

If the training set $\mathbf{z} = (\mathbf{x}, \mathbf{y})$ is not linearly separable in the feature space $\mathcal{H}_\phi$, training points that lie on the wrong side of the margin, i.e.

$$\exists (x, y) \in \mathbf{z} : y \langle w, x \rangle < 1, \tag{2.55}$$

will exist.

Therefore, the optimization problem needs to be adjusted to that situation in order to make it solvable. One popular method to account for points that are possibly on the wrong side of the margin, namely the usage of so-called slack variables. Cristianini and Shawe-Taylor (2000) outline the usage of slack variables in the context of the SVM and Tsochantaridis et al. (2005a) discuss a variety of different types of slack variables linked to specific tasks.

In the following, we will introduce a type of slack variables, that is used in the "1-norm soft margin"-SVM formulation by Cristianini and Shawe-Taylor (2000). For every training point x_i we add a slack variable ξ_i, with $i \in 1, \dots, n$, to the objective function. The slack variable ξ_i measures the distance between the point x_i and the correct boundary or in other words the slack ξ_i measures the wrongness of the point x_i. Since slacks measure the "wrongness" the sum of all slacks is an additional quantity that has to be minimized. Thereby, the optimization problem for the non-linearly separable case is

$$\boxed{\begin{array}{ll} \min & \frac{1}{2}\|w\|_2^2 + \frac{C}{n}\sum_{i=1}^n \xi_i \\ \text{w.r.t.} & w \in \mathcal{H}_\phi, \xi_i \in \mathbb{R}, \text{ for } i \in \{1, \dots, n\} \\ \text{s.t.} & y_i \langle w, \phi(x_i)\rangle \geq 1 - \xi_i, \\ & \xi_i \geq 0 \text{ for } i \in \{1, \dots, n\}, \end{array}} \tag{2.56}$$

where $y_i \langle w, \phi(x_i)\rangle$ is greater or equal to one for points that are on the correct side of the margin and greater or equal to $1 - \xi_i$ for points on the wrong side of the margin, meaning that shifting the point x_i into the direction $y_i w$ by $\frac{\xi_i}{\|w\|_2}$ would put it to the correct side of the margin. The above optimization problem given by equation 2.56 can be cast into the regularized risk minimization framework by multiplying it with $1/C$ and interpreting the slack ξ_i as loss for the training point x_i, a loss function also known as the hinge loss.

Definition 39. (Hinge loss - binary case) *For $\mathcal{Y} = \{-1, 1\}$ the hinge loss c_{hinge} is defined as*

$$c_{hinge} : \mathbb{R} \times \mathcal{Y} \to [0, \infty) : (\hat{y}, y) \mapsto \max(0, 1 - y\hat{y}), \tag{2.57}$$

where we interpret the first argument as prediction and the second one as ground truth.

The hinge loss is given by the optimal value of ξ_i for a fixed w. At this point we can only observe that it is greater or equal to zero and greater or equal to $1 - y_i \langle w, \phi(x_i)\rangle$, which is obtained by rearranging the constraint $y_i \langle w, \phi(x_i)\rangle \geq 1 - \xi_i$. In the following, it will become clear why $\max(0, 1 - y_i \langle w, \phi(x_i)\rangle)$ is really the optimal value of the i-th slack variable.

2.4.3 Solving the Constrained Optimization Problem

The primal problem of the binary support vector machine, given by Equation 2.56, is a quadratic optimization problem with linear constraints. Therefore according to Appendix 2.A it can be solved in it's dual formulation. In order to derive the Lagrangian dual form, we first need to examine the Lagrangian function

$$L(w, \xi, \beta, \alpha) = \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^n \xi_i - \sum_{i=1}^n \beta_i \xi_i - \sum_{i=1}^n \alpha_i (y_i \langle w, \phi(x_i)\rangle - 1 + \xi_i), \tag{2.58}$$

where ξ is a vector containing all the slack variables and β and α are vectors containing the Lagrange multipliers for the inequality constraints. The negative sign at the terms corresponding to the constraints comes from the fact that $a \geq b$ is equivalent to $-a \leq -b$.

The Lagrangian dual function is obtained by minimizing the Lagrangian formula, see Equation 2.58, with respect to the primal variables, ξ and w. Since the objective in Equation 2.58 is

a convex function with respect to w and ξ its minimum can be found by setting the gradients with respect to w and ξ,

$$\frac{\partial L}{\partial w}(w,\xi,\beta,\alpha) = w - \sum_{i=1}^{n} \alpha_i y_i \phi(x_i) \overset{!}{=} 0 \tag{2.59}$$

and

$$\frac{\partial L}{\partial \xi_i}(w,\xi,\beta,\alpha) = C\mathbf{1} - \beta - \alpha \overset{!}{=} 0, \text{ for } i \in \{1,\dots,n\} \tag{2.60}$$

to zero. From Equation 2.59 follows that

$$w = \sum_{i=1}^{n} \alpha_i y_i \phi(x_i) \tag{2.61}$$

and from Equation 2.60 follows that

$$\beta = C\mathbf{1} - \alpha. \tag{2.62}$$

Additionally, the remaining Karush-Kuhn-Tucker (KKT) conditions, namely the KKT complementarity conditions

$$\begin{aligned} &\alpha_i(y_i \langle w, \phi(x_i)\rangle - 1 + \xi_i) = 0 \\ &(\alpha_i - C)\xi_i = 0 \text{ (using } \beta_i = C - \alpha_i) \end{aligned} \tag{2.63}$$

must be satisfied.

Remark 40. (Hinge loss) *The KKT conditions legitimate the definition of the hinge loss, see Equation 2.57. When the i-th boundary constraint, $y_i \langle w, \phi(x_i)\rangle - 1 + \xi_i \geq 0$, is active, i.e. $\alpha_i > 0$, then ξ_i must be equal to $1 - y_i \langle w, \phi(x_i)\rangle$ and α_i must be equal to C and when the i-th boundary constraint is inactive, i.e. $\alpha_i = 0$, then ξ must be equal to zero in order to satisfy the complementarity conditions, given by Equation 2.63.*

By substituting Equation 2.61 and Equation 2.62 back into the Lagrange formula 2.58 the Lagrangian dual function g

$$g(\alpha) = \frac{1}{2}\|\sum_{i=1}^{n} \alpha_i y_i \phi(x_i)\|_2^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}(C-\alpha_i)\xi_i - \sum_{i=1}^{n}\alpha_i(y_i\left\langle \sum_{j=1}^{n}\alpha_j y_j \phi(x_j), \phi(x_i)\right\rangle - 1 + \xi_i) \tag{2.64}$$

is obtained. The expression for g can be simplified by using the identity $\|a\|_2^2 = \langle a, a\rangle$, for $a \in \mathcal{H}_\phi$, and the bilinearity of the inner product, resulting in

$$g(\alpha) = \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \langle \phi(x_i), \phi(x_j)\rangle - \sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \langle \phi(x_i), \phi(x_j)\rangle + \sum_{i=1}^{n}\alpha_i + \sum_{i=1}^{n}(C - \alpha_i - C + \alpha_i)\xi_i. \tag{2.65}$$

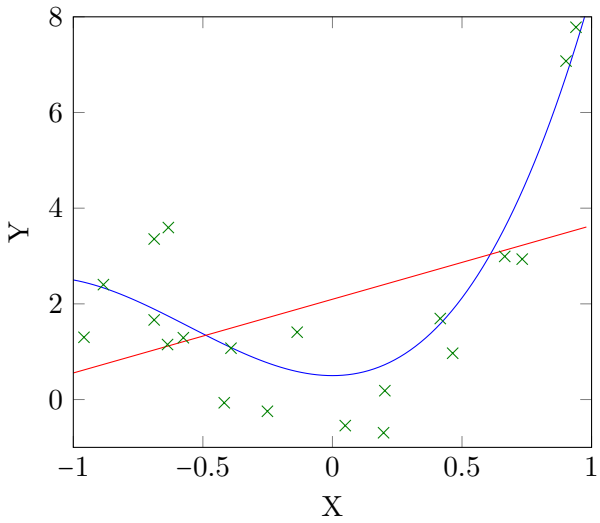
After grouping the terms we get

$$g(\alpha) = -\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \langle \phi(x_i), \phi(x_j)\rangle + \sum_{i=1}^{n}\alpha_i. \tag{2.66}$$

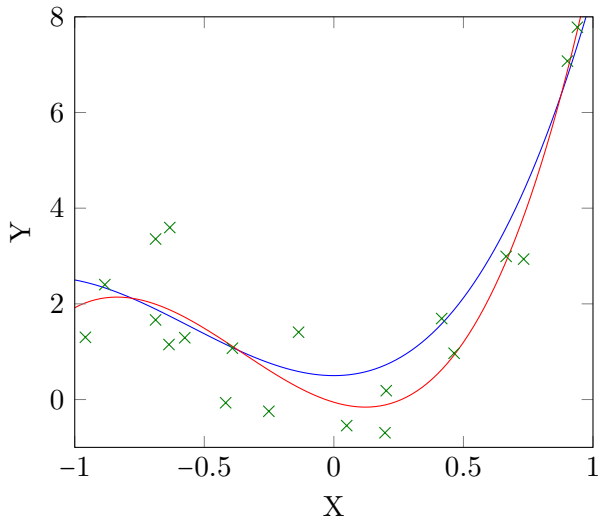
Since the Lagrange dual function g, given by Equation 2.66, provides a lower bound on the optimal value of the optimization problem it needs to be maximized in order to find the best possible lower bound. Maximizing the dual function is equivalent to minimizing its negative

$$\boxed{\begin{array}{ll} \min & \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \langle \phi(x_i), \phi(x_j)\rangle - \sum_{i=1}^{n}\alpha_i \\ \text{w.r.t.} & \alpha_i \in \mathbb{R}, \text{ for } i \in \{1,\dots,n\} \\ \text{s.t.} & 0 \leq \alpha_i \leq C, \text{ for } i \in \{1,\dots,n\}, \end{array}} \tag{2.67}$$

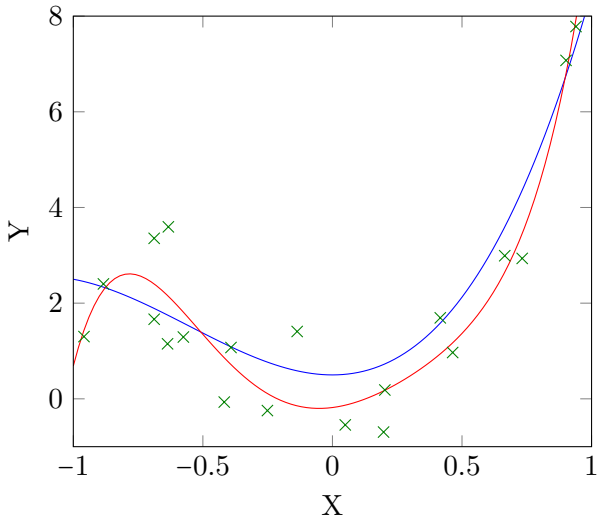
where the box-constraints for α_i come from the KKT conditions and Equation 5.70, since $\alpha_i \geq 0$, $\beta_i \geq 0$ and $\alpha_i = C - \beta_i$ imply that $\alpha_i \leq C$ for $i \in \{1,\dots,n\}$.



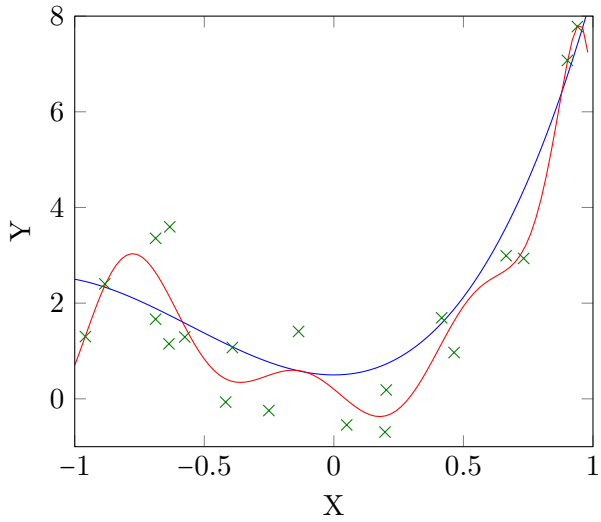
(a) Polynomial of degree one



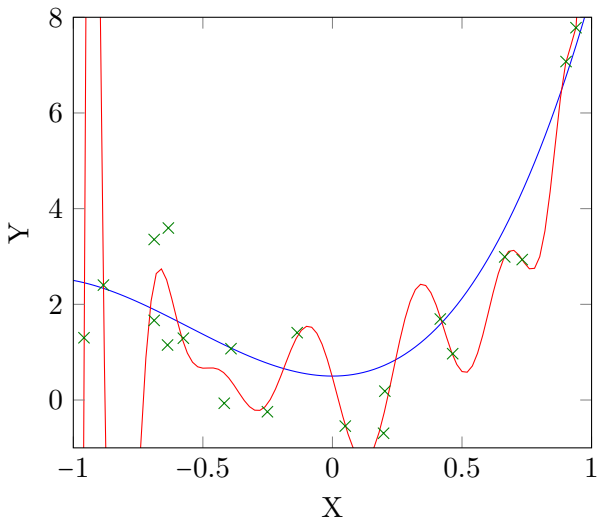
(b) Polynomial of degree three



(c) Polynomial of degree five

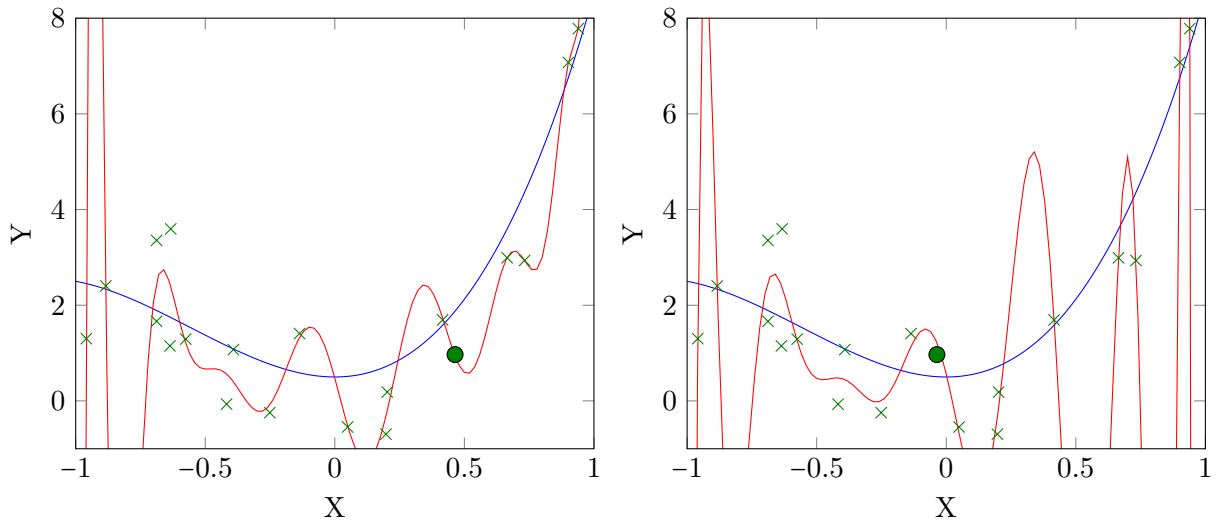


(d) Polynomial of degree ten



(e) Polynomial of degree 15

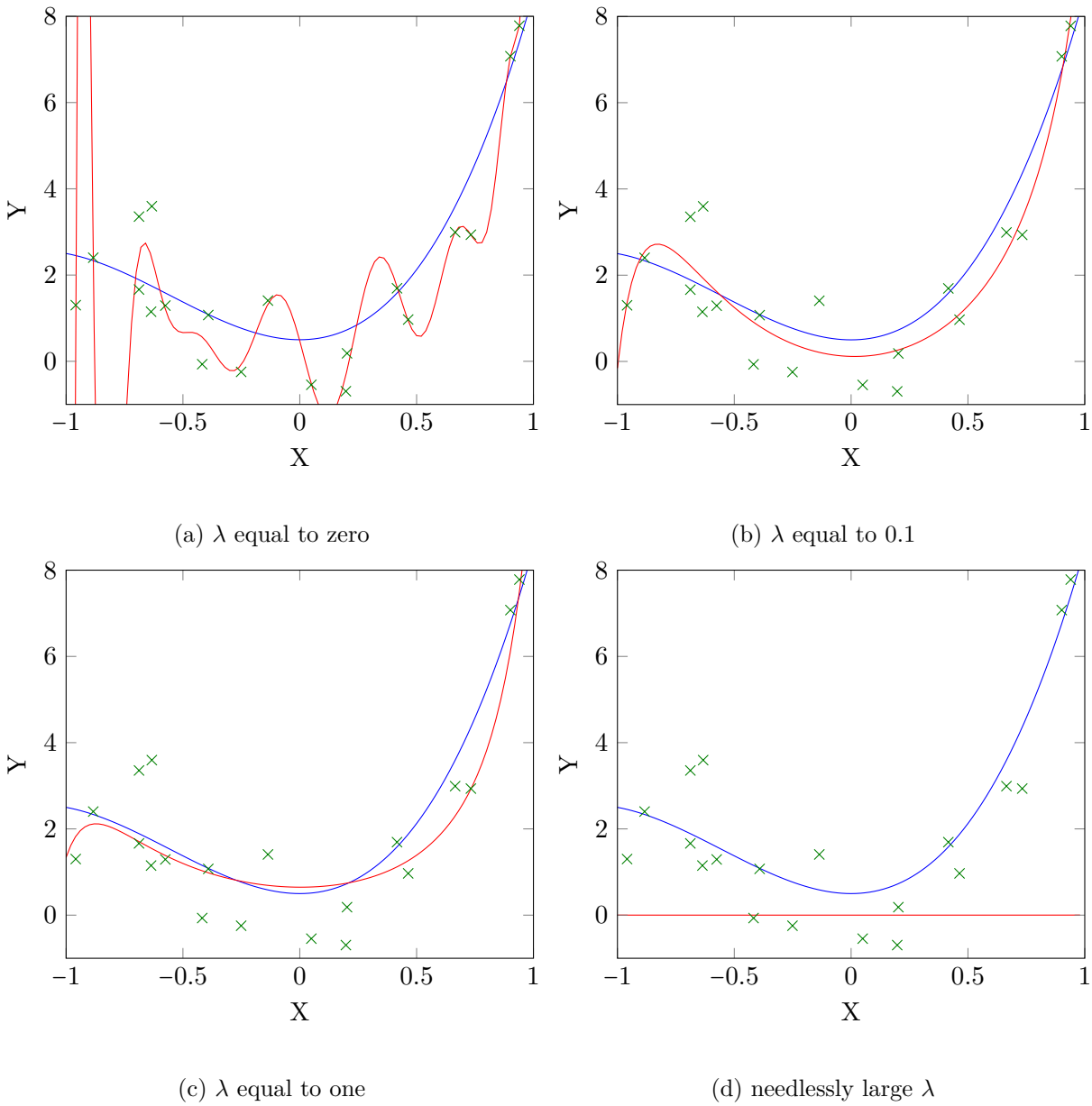
Figure 2.4: Linear regression using polynomials of increasing degree (red). The training data points (green) were generated by evaluating a polynomial function (blue) and adding Gaussian noise.



(a) Polynomial of degree 15

(b) Polynomial of degree 15

Figure 2.5: Linear regression using polynomials of degree 15 (red). The training data points (green) were generated by evaluating a polynomial function (blue) and adding Gaussian noise. The only difference between the training data in the left figure and in the right figure is that the point indicated as a dot in both figures doesn't correspond.



(a) λ equal to zero

(b) λ equal to 0.1

(c) λ equal to one

(d) needlessly large λ

Figure 2.6: Ridge regression using polynomials of degree 15 (red) with different trade-off parameters λ. The training data points (green) were generated by evaluating a polynomial function (blue) and adding Gaussian noise.

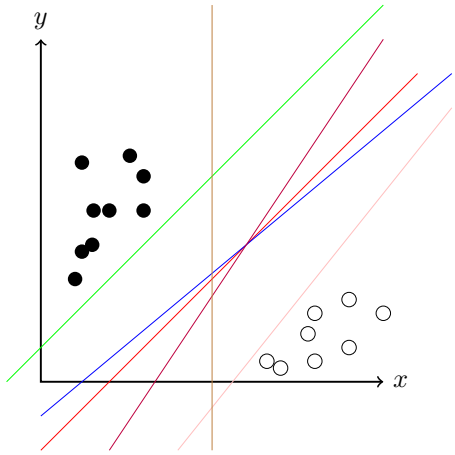


Figure 2.7: Several elements of the version space are illustrated in different colors. All of them minimize the empirical risk with the zero-one loss, however intuitively we would tend to choose a hypothesis similar to the red, blue or purple one. The red line is the one that satisfies the maximum margin property. The illustration is derived from an illustration by Yifan.

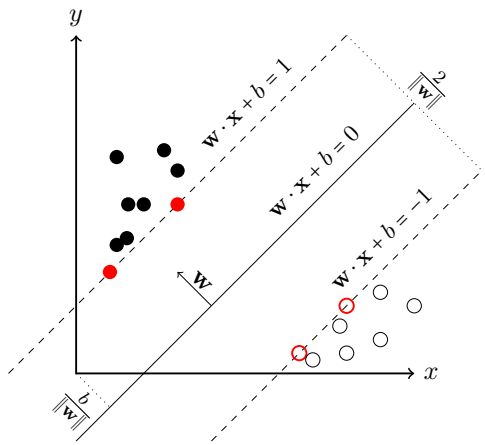


Figure 2.8: The hyperplane with the maximum margin in a two dimensional example. In two dimensions the hyperplane corresponds to a line. For simplicity reasons the feature space mapping $\phi(x) = (x, 1)'$ and the weight vector $\hat{w} = (w, b)'$ resulting in $\langle(\hat{w}, \phi(x))\rangle = wx + b$ are used. The dotted lines illustrate the boundaries of the margin, which are set to one and minus one, respectively. The illustration is taken from Yifan.

Appendix

2.A Constrained Optimization

In this section the Karush-Kuhn-Tucker theory is briefly summarized, the material mainly is taken from the Convex Optimization textbook of Boyd and Vandenberghe (2004), where further details and proofs can be found.

2.A.1 The Problem

The goal of this chapter is to solve an optimization problem with respect to certain constraints

$$\begin{array}{ll} \min & f(x) \\ \text{w.r.t.} & x \in \mathbb{R}^n \\ \text{s.t.} & g_i(x) \leq 0, i = 1, \ldots, m, \\ & h_j(x) = 0, j = 1, \ldots, p. \end{array} \tag{2.68}$$

Example 41. *Problems of this shape naturally occur in many scenarios, imagine for example that you would like to build a windmill at a given contour line of a mountain. Obviously the strength of the wind can vary at different locations and therefore it would be beneficial to choose the location on the contour line with the largest wind strength. In this scenario the variable of interest $x \in \mathbb{R}^3$ would correspond to a position in space, the objective function would be a function $f : \mathbb{R}^3 \to \mathbb{R}$, that gives the wind strength at a given position and the constraint could be another function $h : \mathbb{R}^3 \to \mathbb{R}$,that is zero if and only if the current position is located on the contour line of interest. Since Optimization Problem 2.68 is stated as a minimization problem it the sign of the objective function has to be reversed in order to maximize the wind strength, resulting in*

$$\begin{array}{ll} \min & -f(x) \\ w.r.t. & x \in \mathbb{R}^3 \\ s.t. & h(x) = 0. \end{array} .$$

2.A.2 The Lagrangian Function

Definition 42. (Lagrange function) *For Optimization Problem 2.68, the Lagrange function*

$$L(x, \lambda, \nu) := f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + \sum_{j=1}^{p} \nu_j h_j(x) \tag{2.69}$$

is obtained by adding a linear combination of the constraints to the objective function. The linear factors are often referred to as Lagrange multipliers in honor of "Joseph-Louis de Lagrange".

When having a look at the gradient of the Lagrange function

$$\nabla L(x,\lambda,\nu) = \begin{pmatrix} \frac{\partial L}{\partial x}(x,\lambda,\nu) \\ g(x) \\ h(x) \end{pmatrix}, \tag{2.70}$$

where $g : \mathbb{R}^n \to \mathbb{R}^m$ is a vector valued function with the inequality constraints as component functions and $h : \mathbb{R}^n \to \mathbb{R}^p$ is a vector valued function with the equality constraints as component functions. By setting the gradient of the Lagrange function to zero it is guarantied, that the equality constraints are fulfilled. To ensure the validity of the inequality constraints additional conditions, referred to as Karush-Kuhn-Tucker (KKT) conditions, are necessary. The KKT conditions are named after Kuhn (1982) and Kuhn and Tucker (1951) and are used to generalize the Lagrange multiplier formalism to constraint optimization problems with inequality constraints. After adding the KKT conditions, the optimization problem is

$$\begin{array}{ll} \max_{\lambda,\nu} & \min_x L(x,\lambda,\nu) \\ \text{s.t.} & \lambda_i \geq 0, i = 1,\dots,m, \\ & \lambda_i g_i(x) = 0, i = 1,\dots,m, \end{array} \tag{2.71}$$

which is referred to as the dual problem. We shall see in the remainder of this chapter why the dual problem takes this form.

2.A.3 The Lagrangian dual function

Definition 43. (Lagrange dual function) *The Lagrange dual function is defined as the infimum of the Lagrange function with respect to the primal variable*

$$D : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R} : (\lambda,\nu) \mapsto \inf_x L(x,\lambda,\nu). \tag{2.72}$$

As we shall see in the next section the Lagrange dual function provides a lower bound of the optimal solution.

2.A.4 Linear approximation interpretation

If the optimization problem would be unconstrained and the objective function differentiable it would be straightforward to examine the function's extrema, by setting the gradient to zero and analyzing the definiteness of the hessian matrix. Motivated by that thought Optimization Problem 2.68 can be transformed to an unconstrained optimization problem,

$$\begin{array}{ll} \min & f(x) + \sum_{i=1}^m I_-(g_i(x)) + \sum_{j=1}^p I_0(h_j(x)) \\ \text{w.r.t.} & x \in \mathbb{R}^n, \end{array} \tag{2.73}$$

where the indicator functions

$$I_0 : \mathbb{R} \to \mathbb{R} : u \mapsto \begin{cases} 0 & \text{if } u = 0 \\ \infty & \text{else} \end{cases} \tag{2.74}$$

and

$$I_- : \mathbb{R} \to \mathbb{R} : u \mapsto \begin{cases} 0 & \text{if } u \leq 0 \\ \infty & \text{else} \end{cases} \tag{2.75}$$

are used to exclude unwanted results, by punishing them infinitely hard. If the constraints are fulfilled the indicator functions output zero, and therefore are not affecting the objective

function at all. However if a point violates a constraint the corresponding indicator function adds a penalty ($+\infty$) to the objective. As a consequence points violating the constraints are not considered when looking for the minimum. Linearly approximating the indicator functions yields the *Lagrangian function*

$$L(x, \lambda, \nu) = f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + \sum_{j=1}^{p} \nu_j h_j(x). \tag{2.76}$$

In a certain sense the previously hard punishment of unwanted solutions now turned into a soft one. Assuming $\lambda > 0$ for inequality constraints the punishment is zero when $g_i(x) = 0$, larger than zero when $g_i(x) > 0$ and smaller than zero when $g_i(x) < 0$. Consequently solutions inside of the compactum, parametrized by g_i, with a margin to the boundary are preferred using the soft formulation. Similar considerations can be made for equality constraints, which can be rewritten in terms of inequality constraints

$$h_i(x) = 0 \iff h_i(x) \leq 0 \wedge h_i(x) \geq 0. \tag{2.77}$$

Therefore, the Lagrange function can be written as

$$f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + \sum_{j=1}^{p} \alpha_j h_j(x) - \sum_{j=1}^{p} \beta_j h_j(x), \tag{2.78}$$

where $\alpha > 0$ and $\beta > 0$ are the Lagrange multipliers for the inequality constraints corresponding to the equality constrain. Obviously the equation $\nu = \alpha - \beta$ holds, meaning that ν can be any real number, which is not particularly satisfying when thinking of $\nu_j h_j(x)$ as approximations of the penalty terms. Nevertheless the linear approximations underestimate the indicator functions, since $\lambda_i u \leq I_-(u)$ and $\nu_j u \leq I_0(u)$ hold for all u. As a consequence, the expression

$$\max_{\lambda,\nu} \quad \min_x L(x, \lambda, \nu), \tag{2.79}$$

gives the best lower bound on the optimal value of the original problem. The function $D(\lambda, \nu) = \min_x L(x, \lambda, \nu)$ is called the *Lagrange dual function* of the problem.

2.A.5 Weak and Strong Duality

One question that naturally rises is how good the best lower bound obtained from the dual problem can be. Let p^* denote the optimal value of the Optimization Problem 2.68, in this context usually called primal problem, and let d^* denote the best lower bound on p^* that can be obtained from D, then a property which is called *weak duality*, namely

$$d^* \leq p^*, \tag{2.80}$$

always holds. If the equality

$$d^* = p^* \tag{2.81}$$

holds, we speak of *strong duality*. Slater's theorem, introduced by Slater (2014), provides a sufficient condition for strong duality to hold. Namely, if the primal problem is convex and strictly feasible, such that

$$\exists x_0 \in \mathbb{R}^n : g_i(x_0) < 0, h_j(x_0) = 0 \text{ for } i = 1, \ldots, m \text{ and } j = 1, \ldots, p, \tag{2.82}$$

where the inequality constraints $g_1, \ldots, g_m$ are convex and the equality constraints $h_1, \ldots, h_p$ are affine functions, strong duality will hold.

2.A.6 Karush-Kuhn-Tucker Optimality Conditions

For differentiable functions $f, g_1, \ldots, g_m, h_1, \ldots, h_m$ any pair primal and dual optimal points x^* and (λ^*, ν^*) with strong duality, i.e. a duality gap $f(x^*) - D(\lambda^*, \nu^*)$ equal to zero, must satisfy

$$\nabla f(x^*) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x^*) + \sum_{j=1}^{p} \nu_j \nabla h_j(x^*) = 0, \tag{2.83}$$

since x^* minimizes the Lagrange function $L(x, \lambda^*, \nu^*)$ over x. Therefore, following conditions

$$\begin{aligned} g_i(x^*) &\leq 0,\ i = 1, \ldots, m \\ h_i(x^*) &= 0,\ i = 1, \ldots, p \\ \lambda_i &\geq 0,\ i = 1, \ldots, m \\ \lambda_i g_i(x^*) &= 0,\ i = 1, \ldots, m \\ \nabla f(x) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x) + \sum_{j=1}^{p} \nu_j \nabla h_j(x) &= 0, \end{aligned} \tag{2.84}$$

which are called Karush-Kuhn-Tucker conditions, must hold for any pair of primal and dual optimal points.

Chapter 3

Kernel Methods

3.1 Motivation

We have seen that machine learning methods work well in the linear separable case. However, as far as the real world's problems are concerned, we cannot rely on the linear separability of the data. In fact, we cannot even generally assume that the data is represented in a Hilbert space. We have already seen that one way to deal with non-linear separable and abstract data is to embed it into a Hilbert space, usually referred to as feature space. Thereby, implicitly a different *hypothesis space* is utilized, within which the embedded data points are possibly separable. So far, we rarely had to clarify explicitly which particular feature space mapping we used, despite the fact that in practice - for the solution of a problem - this is a mandatory step. Unfortunately, the choice of a feature space mapping can be difficult, particularly when working with abstract objects. Even when working with vector valued data the embedding into the feature space can be a non-linear function and the feature space can be high dimensional. Even when leaving the choice of feature space aside, the embedding into the feature space and the calculations in the feature space are often computationally expensive.

Recently, kernel methods gained a lot of attention since they provide a tool to use the advantages of using high dimensional feature spaces, while avoiding the computational costs of embedding the data. The following example is intended to provide a better understanding of the problem.

Example 44. (Kernel classifier) *Let the input space $\mathcal{X}$ be a non-empty set, $z \in (\mathcal{X} \times \mathcal{Y})^m$ a training sample and $\phi : \mathcal{X} \to \mathcal{H}_\phi$ a feature space mapping. In the previous chapter we saw that linear classifiers depend on the evaluation of a linear form*

$$f : \mathcal{X} \to \mathbb{R} : x \mapsto \langle w, \phi(x) \rangle .$$

Let's assume for now that w has a representation of the form

$$w = \sum_{i=1}^{m} \gamma_i \phi(x_i), \tag{3.1}$$

in which $\gamma \in \mathbb{R}^m$, like in the SVM example in Equation 2.61, where γ_i would be $y_i \alpha_i$. Then, by using the representation of w given by Equation 3.1 and the bilinearity of the inner product, an evaluation of a linear form can be written as a linear combination of inner products between

training points and the point of interest

$$\begin{aligned} f(x) &= \langle w, \phi(x) \rangle \\ &= \left\langle \sum_{i=1}^{m} \gamma_i \phi(x_i), \phi(x) \right\rangle \\ &= \sum_{i=1}^{m} \gamma_i \langle \phi(x_i), \phi(x) \rangle . \end{aligned} \tag{3.2}$$

Therefore, if we knew how to compute the inner product in the feature space for pairs of elements of the input space, i.e. if we knew how to compute $k(\hat{x}, x) := \langle \phi(\hat{x}), \phi(x) \rangle$*, the explicit feature mapping* ϕ *would not be necessary in order to evaluate linear classifiers. The function* $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *is called kernel function.*

As seen in the previous example, a kernel function corresponds to an inner product in a feature/inner-product space. Therefore, if the problem admits a representation where data points only enter via inner products in the feature space and the kernel function of that feature space is known, the utilization of the benefits provided by the feature space by evaluating the inner products directly via the kernel function will be possible. In many cases it is computationally more efficient to evaluate the kernel function directly than to embed the data and consecutively compute the inner product between the embedded data points. The following example is intended to provide a better idea of the magnitudes of the computational benefits.

Example 45. (Polynomial kernel) *To separate the data depicted in Figure 2.2 a feature space with quadratic basis functions was used. Obviously, the last time when we considered this example prior knowledge about the classes influenced the choice of basis functions of the feature space. Typically, we don't have prior knowledge about the classes in real world applications, which is why it would make sense to include additional basis functions in order to solve more general tasks. Intuitively, one could include linear basis functions and also basis functions in the form of polynomials of higher degree. This considerations ultimately lead to the space of polynomials of a certain degree, which we will see later in this chapter. For now consider polynomials of degree two with two dimensional inputs. One possible feature space mapping in order to represent such polynomials is*

$$\phi : \mathbb{R}^2 \to \mathbb{R}^6 : (x, y) \mapsto (x^2, y^2, \sqrt{2}xy, \sqrt{2}x, \sqrt{2}y, 1)', \tag{3.3}$$

where the coefficients of x, y *and* xy *are chosen equal* $\sqrt{2}$ *for mathematical convenience. Even in this small scale example a quick growth of the dimensionality required to represent the data is visible, if the dimensionality of the input data was increased from two to* $n \geq 2$*, i.e.*

$$\begin{aligned} &\phi_n((x_1, \ldots, x_n)') = \\ &(\underbrace{\underbrace{x_1^2, \ldots, x_n^2}_{n}, \underbrace{\sqrt{2}x_1x_2, \ldots, \sqrt{2}x_1x_n}_{n-1}, \underbrace{\sqrt{2}x_2x_3, \ldots, \sqrt{2}x_2x_n}_{n-2}, \ldots, \underbrace{\sqrt{2}x_{n-1}x_n}_{1}}_{\frac{(n+1)n}{2}}, \underbrace{\sqrt{2}x_1, \ldots, \sqrt{2}x_n}_{n}, \underbrace{1}_{1}), \end{aligned} \tag{3.4}$$

the dimensionality of the second-degree polynomial feature space would increase to $1 + n + \frac{(n+1)n}{2}$*, one dimension for the constant term,* n *dimensions for the linear terms and* $\frac{(n+1)n}{2}$ *dimensions for the quadratic terms. This trend continues when increasing the degree of the polynomial, resulting in high dimensional feature spaces, where computations are costly. In contrast to that, the evaluation of the kernel function corresponding to those feature spaces can be performed efficiently. In the example at hand focusing on the second-order polynomial space the corresponding*

kernel function is

$$
\begin{aligned}
k(u,v) &= \langle \phi(u), \phi(v) \rangle \\
&= (x_1^2, y_1^2, \sqrt{2}x_1y_1, \sqrt{2}x_1, \sqrt{2}y_1, 1) \cdot (x_2^2, y_2^2, \sqrt{2}x_2y_2, \sqrt{2}x_2, \sqrt{2}y_2, 1)' \\
&= x_1^2x_2^2 + y_1^2y_2^2 + +2x_1x_2y_1y_2 + 2x_1x_2 + 2y_1y_2 + 1 \\
&= 1 + 2x_1x_2 + 2y_1y_2 + x_1^2x_2^2 + 2x_1x_2y_1y_2 + y_1^2y_2^2 && \textit{(rearrangement of terms)} \\
&= 1 + 2\langle u,v \rangle + \langle u,v \rangle^2 && \textit{(by definition of } u \textit{ \& } v\textit{)} \\
&= (\langle u,v \rangle + 1)^2 && \textit{(binomial equation)},
\end{aligned}
\tag{3.5}
$$

where $u = (x_1, y_1)'$ *and* $v = (x_2, y_2)'$ *and instead of* $1+n+\frac{(n+1)n}{2}$ *multiplications, that are necessary for the computation of the inner product in the feature space, only* $n+1$ *multiplications are required for the direct evaluation of the corresponding kernel function. In our example* n *was equal to two.*

At this point, it is worth pointing out that considering the kernel version of machine learning problems can be seen as a computational trick in a certain sense, which might be one of the reasons why in the literature (Shawe-Taylor and Cristianini, 2004; Herbrich, 2001) the process of transforming an algorithm into its kernel form is known as the kernel trick. Using kernel methods not only high dimensional feature spaces can be utilized but also infinite dimensional ones, as we will see when having a look at Gaussian kernels. To make a long story short, the goal of this chapter is to utilize kernel functions in order to access the hypothesis spaces induced by feature space mappings in a computationally more efficient way.

3.2 The Kernel Trick

As previously mentioned, the kernel function belonging to a non-linear feature space $\mathcal{H}_\phi$ is given by

$$k(x,y) = \phi(x)'\phi(y) = \langle \phi(x), \phi(y) \rangle, \tag{3.6}$$

which corresponds to the inner product in that feature space. If it is possible to rewrite an algorithm in such a way that the input vectors only enter via inner products, the latter will be replaced by arbitrary kernel functions - a process which is known as the kernel trick.

3.2.1 When Can the Kernel Trick Be Applied?

The Real Valued Case

The kernel trick is applicable to quadratic programming problems with linear constraints. Consequently, the optimization problems of many linear parametric models

$$
\boxed{
\begin{array}{ll}
\min & \frac{1}{2}w'Qw + c'd \\
\text{w.r.t.} & w \in \mathcal{H}_\phi, d \in \mathbb{R}^{|\mathcal{I}\cup\mathcal{J}|}, \\
\text{s.t.} & y_i \langle w, \phi(x_i) \rangle - f_i(d) = 0, \text{ for } i \in \mathcal{I} \\
 & y_j \langle w, \phi(x_j) \rangle - g_j(d) \geq 0, \text{ for } j \in \mathcal{J},
\end{array}
}
\tag{3.7}
$$

where a training sample $\mathbf{z}$ of size n is considered, $\mathcal{I}, \mathcal{J}$ are subsets of $\{1, \ldots, n\}$, ϕ is a feature space mapping from $\mathcal{X}$ to $\mathcal{H}_\phi$, Q is symmetric linear operator from $\mathcal{H}_\phi$ to $\mathcal{H}_\phi$, c *and* d are vectors in $\mathbb{R}^{|\mathcal{I}\cup\mathcal{J}|}$ and f_i *for* $i \in \mathcal{I}$ and g_j *for* $j \in \mathcal{J}$ are differentiable functions from $\mathbb{R}^{|\mathcal{I}\cup\mathcal{J}|}$ to $\mathbb{R}$, allow for the application of the kernel trick.

In order to derive the kernel representation of the optimization problem, consider the corresponding Lagrange function

$$L(w,d,\alpha,\beta) = \frac{1}{2}w'Qw + c'd + \sum_{i\in\mathcal{I}} \alpha_i(\langle w,\phi(x_i)\rangle - f_i(d)) - \sum_{j\in\mathcal{J}} \beta_j(\langle w,\phi(x_j)\rangle - g_j(d)). \tag{3.8}$$

Note that there is no co-occurrence of feature vectors $\phi(x)$, *with* $x \in \mathcal{X}$, and the primal parameter d. As a consequence, the optimization for the parameter d has no impact on the applicability of the kernel trick. Therefore, we only need to consider the derivatives with respect to w in order to figure out whether the kernel trick is applicable. When considering the gradient of the Lagrange function with respect to w, the primal variable w occurs only linearly

$$\frac{\partial L}{\partial w}(w,d,\alpha,\beta) = Qw + \sum_{i\in\mathcal{I}} \alpha_i\phi(x_i) - \sum_{j\in\mathcal{J}} \beta_j\phi(x_j). \tag{3.9}$$

Setting the gradient with respect to the primal variable to zero and solving for the primal variable yields

$$\begin{aligned} w &= Q^{-1}(-\sum_{i\in\mathcal{I}} \alpha_i\phi(x_i) + \sum_{j\in\mathcal{J}} \beta_j\phi(x_j)) \\ &=: Q^{-1}\sum_{i\in\mathcal{K}} \gamma_i\phi(x_i), \text{ with } \mathcal{K} := \mathcal{I}\cup\mathcal{J}, \end{aligned} \tag{3.10}$$

and substituting it back into the Lagrange function will yield a Lagrangian dual problem in which the kernel trick is applicable. When substituting Equation 3.10 into $w'Qw$ we get

$$\begin{aligned} w'Qw &= \left(Q^{-1}\sum_{i\in\mathcal{K}} \gamma_i\phi(x_i)\right)' Q \left(Q^{-1}\sum_{i\in\mathcal{K}} \gamma_i\phi(x_i)\right) \\ &= \left(\sum_{i\in\mathcal{K}} \gamma_i\phi(x_i)'(Q^{-1})'\right)\left(\sum_{i\in\mathcal{K}} \gamma_i\phi(x_i)\right) && \text{(transpose and } QQ^{-1} = id) \\ &= \left(\sum_{i\in\mathcal{K}} \gamma_i\phi(x_i)'Q^{-1}\right)\left(\sum_{i\in\mathcal{K}} \gamma_i\phi(x_i)\right) && \text{(symmetry of } Q^{-1}) \end{aligned} \tag{3.11}$$

since the inverse of a symmetric operator Q^{-1} is a symmetric operator. According to the spectral theorem the symmetric operator Q^{-1} can be written as $(Q^{-1})^{\frac{1}{2}}(Q^{-1})^{\frac{1}{2}}$, yielding

$$\begin{aligned} w'Qw &= \left(\sum_{i\in\mathcal{K}} \gamma_i\phi(x_i)'(Q^{-1})^{\frac{1}{2}}\right)\left((Q^{-1})^{\frac{1}{2}}\sum_{i\in\mathcal{K}} \gamma_i\phi(x_i)\right) && \text{(by } Q^{-1} = (Q^{-1})^{\frac{1}{2}}(Q^{-1})^{\frac{1}{2}}) \\ &= \left\langle \sum_{i\in\mathcal{K}} \gamma_i((Q^{-1})^{\frac{1}{2}})'\phi(x_i), \sum_{i\in\mathcal{K}} \gamma_i(Q^{-1})^{\frac{1}{2}}\phi(x_i)\right\rangle && \text{(by } \phi(x_i)'(Q^{-1})^{\frac{1}{2}} = ((Q^{-1})^{\frac{1}{2}})'\phi(x_i)) \\ &= \left\langle \sum_{i\in\mathcal{K}} \gamma_i(Q^{-1})^{\frac{1}{2}}\phi(x_i), \sum_{i\in\mathcal{K}} \gamma_i(Q^{-1})^{\frac{1}{2}}\phi(x_i)\right\rangle && \text{(by } (Q^{-1})^{\frac{1}{2}} = Pdiag(\sqrt{\lambda_1},\ldots,\sqrt{\lambda_d})P') \\ &= \sum_{i,j\in\mathcal{K}} \gamma_i\gamma_j \left\langle (Q^{-1})^{\frac{1}{2}}\phi(x_i), (Q^{-1})^{\frac{1}{2}}\phi(x_j)\right\rangle && \text{(bilinearity of inner product),} \end{aligned} \tag{3.12}$$

which is a linear combination of inner products in the feature space induced by $(Q^{-1})^{\frac{1}{2}} \circ \phi$.

Similarly substituting Equation 3.10 into $\sum_{i\in\mathcal{I}} \alpha_i w'\phi(x_i) - f_i(d)$ yields

$$\begin{aligned}
\sum_{i\in\mathcal{I}} \alpha_i(w'\phi(x_i) - f_i(d)) &= \sum_{i\in\mathcal{I}} \alpha_i((Q^{-1}\sum_{j\in\mathcal{K}} \gamma_j\phi(x_j))'\phi(x_i) - f_i(d)) \\
&= \sum_{i\in\mathcal{I}} \alpha_i((\sum_{j\in\mathcal{K}} \gamma_j\phi(x_j)')(Q^{-1})'\phi(x_i) - f_i(d)) && \text{(transpose)} \\
&= \sum_{i\in\mathcal{I}} \alpha_i((\sum_{j\in\mathcal{K}} \gamma_j\phi(x_j)')Q^{-1}\phi(x_i) - f_i(d)) && \text{(symmetry of } Q^{-1}) \\
&= \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{K}} (\alpha_i\gamma_j\phi(x_j)'Q^{-1}\phi(x_i)) - \sum_{i\in\mathcal{I}} \alpha_i f_i(d) && \text{(distributivity)} \\
&= \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{K}} \alpha_i\gamma_j \left\langle (Q^{-1})^{\frac{1}{2}}\phi(x_i), (Q^{-1})^{\frac{1}{2}}\phi(x_j)\right\rangle - \sum_{i\in\mathcal{I}} \alpha_i f_i(d) && \text{(similar to Equation 3.12),}
\end{aligned} \tag{3.13}$$

which is a linear combination of inner products in the feature space induced by $(Q^{-1})^{\frac{1}{2}} \circ \phi$. The substitution for $\sum_{i\in\mathcal{J}} \beta_i w'\phi(x_i) - g_i(d)$ works analogously.

We have already seen a concrete example clarifying the statements above, namely the dual problem of the binary support vector machine given by Equation 2.67 which depends only on inner products in the feature space and thereby allows the application of the kernel trick. In order to obtain the corresponding primal optimization problem Q is set to the identity mapping, $\mathcal{J}$ is set to $\{1,\dots,n\}$, $\mathcal{I}$ is set to $\varnothing$, c is set to $\mathbf{1}\frac{C}{n}$ and $g_j(d)$ is set to $1 - d_j$. An alternative framework that is capable of transforming general linear programming problems into their kernel form was introduced by Mangasarian (2006).

Beyond the Real Valued Case

Of course, the kernel trick is not limited to the real valued learning problems. Prior to further investigations concerning theoretical details of the kernel formulation of real valued learning problems, a brief look at the kernel version of the regularized multivariate linear regression will shed more light on the issue. In the following we will encounter various additional examples with more complex output spaces.

Example 46. (Linear Kernel Regression[1])*: In the multivariate case of the linear regression problem with regularization the input space $\mathcal{X}$ is a d-dimensional $\mathbb{R}$-vector space and the output space $\mathcal{Y}$ is a k-dimensional $\mathbb{R}$-vector space. Given a set of observations*

$$\{(x_i, y_i)\}_{i=1}^N \in (\mathcal{X}\times\mathcal{Y})^N \tag{3.14}$$

we are looking for a linear function

$$f : \mathcal{X} \to \mathcal{Y} : x \mapsto \mathbf{W}x, \tag{3.15}$$

parametrized by a the $k \times d$ matrix $\mathbf{W}$, that minimizes the sum-of-squares error

$$\min_{\mathbf{W}} \frac{1}{2}\sum_{i=1}^N \|y_i - \mathbf{W}x_i\|^2 + \frac{C}{2}\|\mathbf{W}\|^2_{Frobenius}. \tag{3.16}$$

Looking at the objective function at hand it is not trivial to formulate the Lagrangian dual. However, substituting $y_i - \mathbf{W}x_i$ with ξ_i and adding equality constraints of the form $\xi_i = y_i - \mathbf{W}x_i$ to the optimization problem

$$\boxed{\begin{array}{ll} \min & \frac{1}{2}\sum_{i=1}^N \|\xi_i\|^2 + \frac{C}{2}\|\mathbf{W}\|^2_{Frobenius} \\ w.r.t. & \mathbf{W} : \mathcal{X} \to \mathcal{Y}, \xi \in \mathcal{Y} \\ s.t. & \xi_i = y_i - \mathbf{W}x_i, i \in \{1\dots N\} \end{array}} \tag{3.17}$$

[1]Similar examples are considered by Saunders et al. (1998) and Cortes et al. (2006).

does the trick. As a consequence, the linear regression problem can be considered from a constrained optimization perspective, where the Lagrangian dual form can be derived. The Lagrange function of the constrained optimization problem is

$$L(\mathbf{W}, \xi, \mathbf{A}) = \frac{1}{2}\sum_{i=1}^{N}\|\xi_i\|^2 + \frac{C}{2}\|\mathbf{W}\|^2_{Frobenius} + \sum_{i=1}^{N}\alpha_i'(\mathbf{W}x_i - y_i + \xi_i), \tag{3.18}$$

where $\mathbf{A}$ *is a* $k \times N$ *matrix with columns* $\alpha_1, \ldots, \alpha_N$*. In order to obtain the Lagrangian dual function the gradients of the Lagrangian with respect to the primal variables*

$$\frac{\partial L}{\partial \mathbf{W}}(\mathbf{W}, \xi, \mathbf{A}) = C\mathbf{W} + \sum_{i=1}^{N}\alpha_i x_i' \overset{!}{=} 0 \tag{3.19}$$

and

$$\frac{\partial L}{\partial \xi_i}(\mathbf{W}, \xi, \mathbf{A}) = \xi_i + \alpha_i \overset{!}{=} 0 \tag{3.20}$$

are set to zero. From Equation 3.19 follows that

$$\mathbf{W} = -\frac{1}{C}\sum_{i=1}^{N}\alpha_i x_i' \tag{3.21}$$

and from Equation 3.20 follows that

$$\xi_i = -\alpha_i. \tag{3.22}$$

By substituting Equation 3.21 and Equation 3.22 back into the Lagrange formula, given by Equation 3.18, the Lagrangian dual function is obtained. The substitution of the expression for $\mathbf{W}$ *into the Frobenius norm is considered separately before performing the substitution into the Lagrange function and yields*

$$\begin{aligned}
\|-\frac{1}{C}\sum_{i=1}^{N}\alpha_i x_i'\|^2_{Frobenius} &= \left\langle -\frac{1}{C}\sum_{i=1}^{N}\alpha_i x_i', -\frac{1}{C}\sum_{j=1}^{N}\alpha_j x_j' \right\rangle_{Frobenius} \\
&= \frac{1}{C^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left\langle \alpha_i x_i', \alpha_j x_j' \right\rangle_{Frobenius} && \textit{(bilinearity of inner product)} \\
&= \frac{1}{C^2}\sum_{i=1}^{N}\sum_{j=1}^{N} tr((\alpha_i x_i')'\alpha_j x_j') && \textit{(definition of Frobenius inner product)} \\
&= \frac{1}{C^2}\sum_{i=1}^{N}\sum_{j=1}^{N} tr(x_i\alpha_i'\alpha_j x_j') && \textit{(transpose)} \\
&= \frac{1}{C^2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i'\alpha_j tr(x_i x_j') && (\alpha_i' \cdot \alpha_j \textit{ is a scalar \& linearity of trace}) \\
&= \frac{1}{C^2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i'\alpha_j \langle x_i, x_j \rangle && \textit{(tr(xy') = tr(x'y))},
\end{aligned} \tag{3.23}$$

in which $\|A\|_{Frobenius} = tr(A'A)$ *denotes the Frobenius norm and* $\langle A, B\rangle_{Frobenius} := \sum_{i,j} A_{ij}B_{ij}$ *the Frobenius inner product.*

Similarly by substituting the new expression for $\mathbf{W}$ *into* $\sum_{i=1}^{N} \alpha_i' \mathbf{W} x_i$

$$\begin{aligned}\sum_{i=1}^{N} \alpha_i'(-\frac{1}{C}\sum_{j=1}^{N} \alpha_j x_j')x_i &= -\frac{1}{C}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i'\alpha_j x_j' x_i \\ &= -\frac{1}{C}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i'\alpha_j \langle x_i, x_j\rangle\end{aligned} \tag{3.24}$$

is obtained. Substituting everything back into the Lagrange formula given by Equation 3.18 yields

$$\begin{aligned}g(\mathbf{A}) &= -\frac{1}{2}\sum_{i=1}^{N} \|\alpha_i\|^2 + \frac{1}{2C}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i'\alpha_j \langle x_i, x_j\rangle - \frac{1}{C}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i'\alpha_j \langle x_i, x_j\rangle \\ &= -\frac{1}{2}\sum_{i=1}^{N} \|\alpha_i\|^2 - \frac{1}{2C}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i'\alpha_j \langle x_i, x_j\rangle\,,\end{aligned} \tag{3.25}$$

where the kernel trick can be applied. As a result, we end up the dual function

$$g(\mathbf{A}) = -\frac{1}{2}\sum_{i=1}^{N} \|\alpha_i\|^2 - \frac{1}{2C}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i'\alpha_j k(x_i, x_j). \tag{3.26}$$

Since the Lagrange dual function g, given by Equation 3.25, provides a lower bound on the optimal value of the optimization problem, it needs to be maximized in order to find the best possible lower bound. Maximizing the dual function is equivalent to minimizing its negative

$$\boxed{\begin{array}{ll}\min & \frac{1}{2}\sum_{i=1}^{N} \|\alpha_i\|^2 + \frac{1}{2C}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i'\alpha_j k(x_i, x_j) \\ w.r.t. & \alpha_i \in \mathcal{Y}, i \in \{1 \dots N\}\,.\end{array}} \tag{3.27}$$

3.2.2 Summary and Outlook

For many machine learning algorithms the kernel trick is applicable. It allows for the access to the hypothesis space given by the composition of the feature space mapping $\phi : \mathcal{X} \to \mathcal{H}_\phi$ and linear forms from $\mathcal{H}_\phi$ to $\mathbb{R}$,

$$\mathscr{H}_\phi := \{g : \mathcal{X} \to \mathbb{R} : g = f \circ \phi, f \in \mathcal{H}_\phi^* \text{ and feature space mapping } \phi\}, \tag{3.28}$$

without the need of an explicit representation of the feature space mapping, namely by considering

$$\mathscr{H}_k := \{g : \mathcal{X} \to \mathbb{R} : g = \sum_{i=1}^{n} \alpha_i k(x_i, \cdot),\ \text{for } n \in \mathbb{N}, x_1, \dots, x_n \in \mathcal{X}, \alpha_1, \dots \alpha_n \in \mathbb{R} \text{ and } k \text{ is a kernel function}\}, \tag{3.29}$$

where $k(x, y) := \langle \phi(x), \phi(y)\rangle$. It is to be noted that this is a non-trivial statement, which has so far only been motivated by examples. In the following, additional work will be put into showing that the equivalence

$$\mathscr{H}_\phi \Leftrightarrow \mathscr{H}_k \tag{3.30}$$

holds.

Chapter 4

A Glance at Kernel Theory

This chapter is meant to introduce some of the theorems that build the theoretic foundation of kernel methods, how they are used in the machine learning context, respectively. We revise the theoretical foundation of the kernel trick and the usage of the resulting hypothesis spaces in the scope of the regularized risk minimization framework. Most of the material is based on reviews about reproducing kernel Hilbert spaces by Hofmann et al. (2006) and Wahba (2003) and the statistical learning course notes of Poggio and Rosasco (2015).

4.1 Terminology - Kernel

The type of functions that are referred to as kernel functions in the context of machine learning were initially examined by mathematicians, like Hilbert (1904), Mercer (1909) and Aronszajn (1950), in the first half of the 20th century in the scope of integral calculus. Hilbert (1989) used the German term 'Kern' for complete quadratic forms that characterize linear integral equations of the form

$$\phi(s) = f(s) + \int_a^b k(s,t)\phi(t)dt, \tag{4.1}$$

where the goal is to find the unknown function ϕ given an inhomogeneity function f and a 'Kern' k. In the German language the term 'Kern' - comparable to the English word core - refers to the innermost part of an object or organism such as a fruit. The 'Kern' is the most central and essential part of the fruit since it is fully determined by its 'Kern' carrying the necessary information and ability for reproduction. Similarly, the 'Kern' of an integral equation represents its most central part, leaving the inhomogeneity aside - the 'Kern' fully determines its solution. In the machine learning context the kernel function ('Kern') can be thought of as the most central part of a kernel method as well, since the choice of kernel function determines the hypothesis space used. Despite the fact that the 'Kern' serves different purposes in the fruit, the integral equation and the kernel method, it determines the whole system.

4.2 Reproducing Kernel Hilbert Spaces (RKHS)

4.2.1 Outline

In the following, different perspectives on a rich family of hypothesis spaces, so-called reproducing kernel Hilbert spaces (RKHS) as illustrated in Figure 4.1, are going to be studied. First a mathematical definition will be introduced followed by a demonstration showing the one to one correspondence between RKHS and positive definite kernels. In addition, the equivalence of $\mathscr{H}_\phi$ and $\mathscr{H}_k$, given by Equation 3.28 and Equation 3.29, is going to be shown.

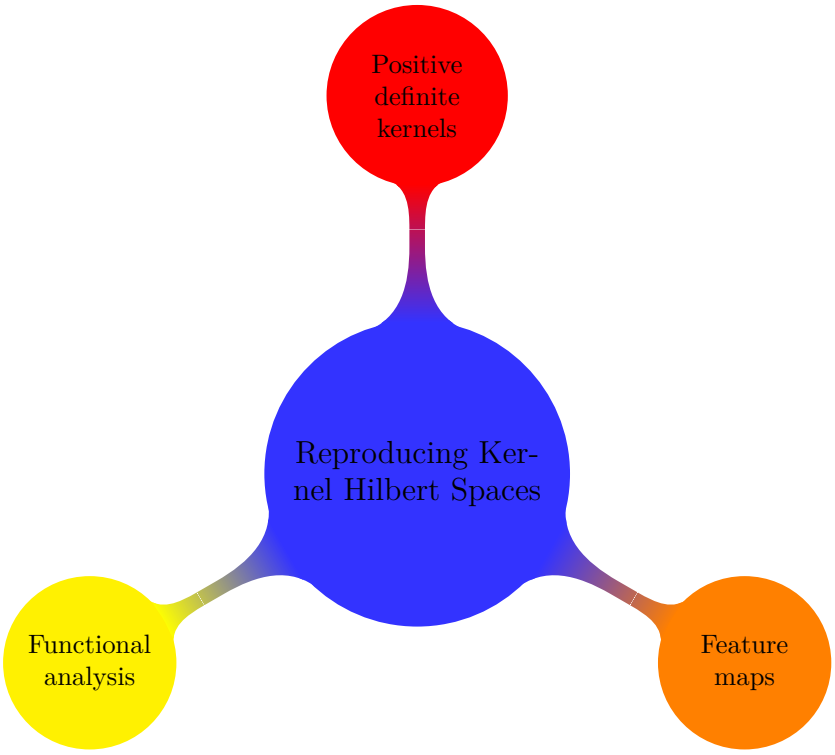


Figure 4.1: Different perspectives on reproducing kernel Hilbert spaces.

4.2.2 Recap & Important Properties of Hilbert Spaces

Before looking at the theorems that provide theoretical justification for the kernel trick, some mathematical definitions and results of relevance, taken from Rudin (2006), Wagner (2004) and Hell and Neumann (2012), are going to be revised.

Function space In Chapter 2 we introduced the notion of *function space* as a set of functions from set $\mathcal{X}$ to set $\mathcal{Y}$, see Definition 3. Function spaces are often also vector spaces.

Hypothesis space The function space considered when solving a learning task is referred to as *hypothesis space*, see Definition 4.

Hilbert space $\mathbb{K}$-Vector spaces with an inner product, that are complete with respect to the metric induced by their inner product, are so-called *Hilbert spaces*, see Definition 16. For simplicity it is to be assumed that the field of real numbers, i.e. $\mathbb{K} := \mathbb{R}$, will be used without exception.

Dual space So far function spaces spanned by linear forms on Hilbert spaces called *feature spaces* were used, see Definition 18. The space spanned by the linear forms from a Hilbert space $\mathcal{H}$ to $\mathbb{R}$

$$\mathcal{H}^* = \{f : \mathcal{H} \to \mathbb{R}\}, \tag{4.2}$$

is called *dual space*. We will see that for the topological dual space $\mathcal{H}'$ of a Hilbert space is a Hilbert space with point-wise addition, point-wise scalar multiplication and an inner product.

In a Hilbert space of functions we are allowed to add functions, multiply them with scalars without leaving the Hilbert space. Additionally, thanks to the inner product associated with the

Hilbert space, we can talk about both orthogonality and distances, since angles and distances are closely related to inner products. In order to study the connections between Hilbert spaces, positive definite kernels and feature spaces, some additional properties of Hilbert spaces are required, for instance the that Hilbert spaces are topological spaces.

Topological spaces

Earlier, the choice of linear functions on finite dimensional vector spaces as hypothesis functions was partially motivated by the fact that linear functions on vector spaces are always continuous. In the infinite dimensional spaces a more general concept of continuity is required.

Definition 47. (Topological space) *The pair $(\mathcal{X}, \mathscr{T})$, where $\mathcal{X}$ is a set and $\mathscr{T}$ is a collection of subsets of $\mathcal{X}$ satisfying the axioms:*

1. *$\mathcal{X} \in \mathscr{T}$ and $\varnothing \in \mathscr{T}$,*
2. *for all sequences $(\mathcal{A}_i)_{i\in\mathbb{N}} \in \mathscr{T}^{\mathbb{N}}$ holds that $\bigcup_{i\in\mathbb{N}} \mathcal{A}_i \in \mathscr{T}$,*
3. *for all finite sequences $(\mathcal{A}_i)_{i=1}^N$ holds that $\bigcap_{i=1}^N \mathcal{A}_i \in \mathscr{T}$,*

is called topological space*. The collection of sets $\mathscr{T}$ is called* topology *and defines which subsets are open or closed.*

Definition 48. (Open subset & closed subset) *A subset $\mathcal{S} \subset \mathcal{X}$ of a topological space $(\mathcal{X}, \mathscr{T})$ is open if $\mathcal{S} \in \mathscr{T}$ and closed if $\mathcal{X} \backslash \mathcal{S} \in \mathscr{T}$.*

Definition 49. (Topological subspace) *A subset $\mathcal{S}$ of a topological space $(\mathcal{X}, \mathscr{T})$ is a topological space equipped with the subspace topology*

$$\mathscr{T}_{\mathcal{S}} := \{\mathcal{S} \cap \mathcal{O} : \mathcal{O} \in \mathscr{T}\}. \tag{4.3}$$

This abstract definition of open and closed subsets leads to a more abstract definition of continuity.

Definition 50. (Continuous function) *Let $(\mathcal{X}, \mathscr{T}_{\mathcal{X}})$ and $(\mathcal{Y}, \mathscr{T}_{\mathcal{Y}})$ be topological spaces, then a function f*

$$f : \mathcal{X} \to \mathcal{Y} \tag{4.4}$$

is continuous iff pre-images of open subsets of $\mathcal{Y}$ under f are open subsets of $\mathcal{X}$, i.e.

$$\forall \mathcal{O}_{\mathcal{Y}} \in \mathscr{T}_{\mathcal{Y}} : f^{-1}(\mathcal{O}_{\mathcal{Y}}) \in \mathscr{T}_{\mathcal{X}}. \tag{4.5}$$

Example 51. (Standard topology on $\mathbb{R}^d$) *The standard topology $\mathscr{T}$ on $\mathbb{R}^d$ is defined utilizing the notion of open balls. An open ball of radius r around the point $x \in \mathcal{X}$ is defined as*

$$B_r(x) := \{y \in \mathbb{R}^d : \|x - y\| < r\}, \tag{4.6}$$

where $\|\cdot\|$ is a norm on $\mathbb{R}^d$. Consequently, an open subset $\mathcal{O}$ of $\mathbb{R}^d$ is characterized by the fact that for every point $x \in \mathcal{O}$ there exists an $\epsilon > 0$ such that the ball $B_\epsilon(x)$ is a subset of $\mathcal{O}$. The set of all open subsets, defined this way, is called the standard topology *on $\mathbb{R}^d$.*

Metric spaces

Definition 52. (Metric space) *The pair* $(\mathcal{X}, d)$*, where* $\mathcal{X}$ *is a set and* d *is a metric, i.e. a function from* $\mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *satisfying*

1. $d(x, y) \geq 0$ *(non-negativity),*
2. $d(x, y) = 0 \Leftrightarrow x = y$ *(identity of indiscernibles),*
3. $d(x, y) = d(y, x)$ *(symmetry),*
4. *and* $d(x, z) \leq d(x, y) + d(y, z)$ *(triangle inequality),*

is called metric space.

The reasoning applied in Example 51 can be extended to metric spaces $(\mathcal{X}, d)$, where open balls are defined using the metric d

$$B_r(x) := \{y \in \mathcal{X} : d(x, y) < r\}. \tag{4.7}$$

Accordingly, the metric of a metric space induces a topology. As a consequence, Hilbert spaces are topological spaces as well, since the *inner-product* induces a *norm* and this *norm* induces a *metric*, which induces a topology.

Definition 53. (Complete metric space) *A metric space* $(\mathcal{X}, d)$ *is complete iff the limit* $x := \lim_{n\to\infty} x_n$ *of every Cauchy sequence* $(x_i)_{i\in\mathbb{N}} \in \mathcal{X}^{\mathbb{N}}$*, i.e. every sequence* $(x_i)_{i\in\mathbb{N}} \in \mathcal{X}^{\mathbb{N}}$ *satisfying*

$$\forall \epsilon > 0 : \exists N \in \mathbb{N} \text{ such that } \forall n, m \in \mathbb{N} \text{ with } m > N \text{ and } n > N : d(x_m, x_n) < \epsilon, \tag{4.8}$$

is contained in $\mathcal{X}$*.*

Theorem 54. (Complete subspace) *A closed subspace of a complete space is a complete subspace.*

Proof. The interested reader is referred to Rudin (2006). □

Hilbert spaces

The notion of Hilbert spaces combines the topological structure with the structure of inner product spaces. At this point it is necessary to recall the definition of Hilbert spaces.

Definition 55. (Hilbert space) *A Hilbert space* $\mathcal{H}$ *is an inner-product space, which is complete with respect to the metric induced by it's inner-product.*

Example 56. (Examples of Hilbert spaces) *The following spaces are Hilbert spaces:*

1. l^2*, the space of square-summable sequences*

$$l^2(\mathbb{R}) := \{(x_i)_{i\in\mathbb{N}} \in \mathbb{R}^{\mathbb{N}} : \sum_{i=1}^{\infty} |x_i|^2 < \infty\} \tag{4.9}$$

with the inner product

$$\langle x, y\rangle_{l^2} := \sum_{i=1}^{\infty} x_i y_i. \tag{4.10}$$

2. $\mathbf{L}^2$, *the space of square-integrable functions*

$$\mathbf{L}^2(\mathbb{R}) := \{f \in \mathbb{R}^{\mathbb{R}} : \int_{-\infty}^{\infty} |f(x)|^2 dx < \infty\} / \{g \in \mathbb{R}^{\mathbb{R}} : \int_{-\infty}^{\infty} |g(x)|^2 dx = 0\} \tag{4.11}$$

with the inner product

$$\langle f, g \rangle_{\mathbf{L}^2} := \int_{-\infty}^{\infty} f(x) g(x) dx. \tag{4.12}$$

3. *Any finite dimensional Euclidean vector space with the corresponding scalar product.*

The interested reader may verify that the listed examples are Hilbert spaces, by checking the Hilbert space properties.

Definition 57. (Topological dual space) *The space spanned by linear and continuous froms on a Hilbert space $\mathcal{H}$ is referred to as the topological dual space $\mathcal{H}'$. Clearly, the relation $\mathcal{H}' \subset \mathcal{H}^*$ holds.*

With the properties of the inner product the complement of a subspace can be uniquely defined.

Definition 58. (Orthogonal complement) *For a subspace $\mathcal{S}$ of a Hilbert space $\mathcal{H}$ the orthogonal complement is defined as*

$$\mathcal{S}^{\perp} := \{x \in \mathcal{H} : \forall y \in \mathcal{S} : \langle x, y \rangle_{\mathcal{H}} = 0\}. \tag{4.13}$$

The orthogonal complement is always closed.

Theorem 59. ($\mathcal{S} \oplus \mathcal{S}^{\perp} \simeq \mathcal{H}$) *Let $\mathcal{H}$ be a Hilbert space and $\mathcal{S}$ be a subspace of $\mathcal{H}$, then*

$$\oplus : \mathcal{S} \times \mathcal{S}^{\perp} \to \mathcal{H} : (x, y) \mapsto x + y, \tag{4.14}$$

defines an isomorphism.

Proof. The proof is not particularly interesting for this thesis. It basically needs to be shown, that the function $\oplus$ is bijective and continuous. The interested reader can find the proof in Wagner (2004). □

Utilizing the definitions and results introduced so far, we can prove a theorem that will turn out to be important in the following.

Theorem 60. (Riesz representation theorem) *Let $\mathcal{H}$ be a Hilbert space, not necessarily a Hilbert space of functions. The mapping ψ defined as*

$$\psi : \mathcal{H} \to \mathcal{H}' : x \mapsto \langle x, \cdot \rangle_{\mathcal{H}} \tag{4.15}$$

is an isomorphism, i.e. ψ is bijective and continuous.

Proof. The ideas of the following proof were taken from Wagner (2004). In order to show that ψ defines an isomorphism between $\mathcal{H}$ and $\mathcal{H}^*$, we only need to prove that ψ exists and that it is bijective, since the linearity follows from the bilinearity of the inner product.

Well-posedness: From

$$\begin{aligned} |\psi(x)(y)| &= |\langle x, y \rangle_{\mathcal{H}}| && \text{(by definition)} \\ &\leq \|x\|_{\mathcal{H}} \|y\|_{\mathcal{H}} && \text{(Cauchy-Schwarz inequality),} \end{aligned} \tag{4.16}$$

for $x, y \in \mathcal{H}$, follows

$$\begin{aligned}
\|\psi(x)\| &:= \sup_{\|y\|_{\mathcal{H}} \leq 1} |\psi(x)(y)| && \text{(operator norm)} \\
&= \sup_{\|y\|_{\mathcal{H}} \leq 1} |\langle x, y\rangle_{\mathcal{H}}| && \text{(definition of } \psi) \\
&\leq \sup_{\|y\|_{\mathcal{H}} \leq 1} |\|x\|_{\mathcal{H}} \|y\|_{\mathcal{H}}| && \text{(Cauchy-Schwarz inequality)} \\
&= \|x\|_{\mathcal{H}} < \infty
\end{aligned} \tag{4.17}$$

and thereby the existence of $\psi(x)$ for all $x \in \mathcal{H}$.

Injectivity: In order to show that ψ is injective, we show that it is an isometric functional, i.e. for all $x \in \mathcal{H}$ the property $\|\psi(x)\| = \|x\|_{\mathcal{H}}$ holds. From Equation 4.17 we already know an upper bound for $\|\psi(x)\|$, namely $\|x\|_{\mathcal{H}}$. If we manage to show that the lower bound of $\|\psi(x)\|$ is also $\|x\|_{\mathcal{H}}$ for all $x \in \mathcal{H}$, we will know that ψ is isometric. The case $x = 0$ is simple, from Equation 4.17 and the properties of the norm, in particular from $\|\cdot\| \geq 0$, follows that $\|\psi[0]\| = 0$. If $x \in \mathcal{H}\backslash\{0\}$,

$$\begin{aligned}
\|\psi(x)\| &= \sup_{\|y\|_{\mathcal{H}} \leq 1} |\langle x, y\rangle_{\mathcal{H}}| && \text{(definition of } \psi) \\
&\geq \left|\langle x, \frac{x}{\|x\|_{\mathcal{H}}}\rangle_{\mathcal{H}}\right| && \text{(supremum property)} \\
&= \frac{\langle x, x\rangle_{\mathcal{H}}^2}{\|x\|_{\mathcal{H}}} = \|x\|_{\mathcal{H}} && \text{(absolute homogenity)} \\
&= \frac{\|x\|_{\mathcal{H}}^2}{\|x\|_{\mathcal{H}}} = \|x\|_{\mathcal{H}} && \text{(norm induced by inner product)}
\end{aligned} \tag{4.18}$$

will hold. Therefore, ψ is an isomeric functional, since Equation 4.17 and Equation 4.18 imply $\|x\|_{\mathcal{H}} \leq \|\psi(x)\| \leq \|x\|_{\mathcal{H}}$, which is equivalent to $\|\psi(x)\| = \|x\|_{\mathcal{H}}$.

Surjectivity: To prove the surjectivity of ψ we need to show that $\psi(\mathcal{H}) = \mathcal{H}'$ or equivalently that for every $f \in \mathcal{H}'$ there exists a $x \in \mathcal{H}$, with $\psi(x) = f$. The case $f = 0$ is trivial, it is easy to check that $\psi(0) = 0$. If $f \in \mathcal{H}'\backslash\{0\}$,

$$ker(f) = \{x \in \mathcal{H} : f(x) = 0\} \subseteq \mathcal{H} \tag{4.19}$$

is a complete subspace of $\mathcal{H}$. By Theorem 59 an element $x \in (ker(f))^{\perp}\backslash\{0\}$ exists. For all $x \in (ker(f))^{\perp}$ the inequality $f(x) \neq 0$ holds. Next, we are going to exploit that the functions $f \in \mathcal{H}'$ are linear to construct elements of $ker(f)$. To do that, consider the following derivation

$$\begin{aligned}
0 &= f(y) - f(y) && \text{(Trick 17)} \\
&= f(y) - \underbrace{\frac{f(y)}{f(x)}}_{\in\mathbb{R}} f(x) && \text{(Trick 17)} \\
&= f(y - \frac{f(y)}{f(x)}x) && (\mathbb{R} \text{ linearity of f}),
\end{aligned} \tag{4.20}$$

where $y \in \mathcal{H}$. From Equation 4.20 follows $y - \frac{f(y)}{f(x)}x \in ker(f)$ for all $y \in \mathcal{H}$. Now, since we have managed to construct an element of $ker(f)$ which depends on $f(y)$, we utilize the fact that a x

is an element of $(ker(f))^{\perp}$ in order to obtain a representation of $f(y)$. Thus, the inner product

$$\begin{aligned} 0 &= \Big\langle \underbrace{x}_{\in(ker(f))^{\perp}}, \underbrace{y - \frac{f(y)}{f(x)}x}_{\in ker(f)} \Big\rangle_{\mathcal{H}} && \text{(by definition 58)} \\ &= \langle x, y\rangle - \Big\langle x, \frac{f(y)}{f(x)}x \Big\rangle_{\mathcal{H}} && \text{(linearity of inner product)} \\ &= \langle x, y\rangle - \frac{f(y)}{f(x)}\langle x, x\rangle_{\mathcal{H}} && \text{(linearity of inner product),} \end{aligned} \tag{4.21}$$

which is equivalent to

$$\begin{aligned} f(y) &= \frac{\langle x, y\rangle_{\mathcal{H}}}{\|x\|^2} f(x) && \text{(norm induced by inner product)} \\ &= \langle \frac{xf(x)}{\|x\|^2}, y\rangle_{\mathcal{H}} && \text{(linearity of inner product),} \end{aligned} \tag{4.22}$$

has to be considered. Equation 4.22 implies that

$$\psi(\frac{xf(x)}{\|x\|^2}) = f. \tag{4.23}$$

As a consequence, the above procedure allows for finding an element $\psi^{-1}(f) \in \mathcal{H}$ for every $f \in \mathcal{H}'$, meaning that ψ is surjective. □

Remark 61. (Dual of a Hilbert space) *One direct implication of the Riesz representation theorem (Theorem 60) is that the dual of a Hilbert space $\mathcal{H}$ is a Hilbert space, with point-wise addition, point-wise scalar multiplication and the inner product*

$$\langle \cdot, \cdot \rangle_{\mathcal{H}'} : \mathcal{H}' \times \mathcal{H}' \to \mathbb{R} : (f, g) \mapsto \left\langle \psi^{-1}(f), \psi^{-1}(g) \right\rangle_{\mathcal{H}}, \tag{4.24}$$

where ψ is the isomorphism defined by Riesz representation theorem. The function, given by Definition 4.24, inherits all properties from the inner product of $\mathcal{H}$ and therefore is an inner product itself. Additionally, the completeness of $\mathcal{H}$ is inherited. We are going to use this observation when we consider $\mathscr{H}_{\phi}$ later in this section.

4.2.3 Functional Analysis perspective

If not stated otherwise, let $\mathcal{X}$ be an arbitrary set and $\mathcal{H}$ a Hilbert space of real-valued functions on $\mathcal{X}$ in the following. The notion of Hilbert spaces allows for working with functions in the same way as with vectors. The similarity between vectors and functions can be further increased by considering functions that can be defined point-wisely. A vector $v \in \mathbb{R}^n$ for example can be interpreted as a point-wise function from a subset of $\mathbb{N}$ to it's field $\mathbb{R}$, namely

$$v : \{1, \ldots, n\} \subseteq \mathbb{N} \to \mathbb{R} : i \mapsto v_i. \tag{4.25}$$

In order to study spaces of point-wise functions we define evaluation functionals. If the evaluation functionals are bounded we will be in a similar situation as in vector spaces.

Definition 62. (Evaluation functional) *An evaluation functional over the Hilbert space of functions $\mathcal{H}$ is a linear functional that evaluates a function $f \in \mathcal{H}$ at a point $x \in \mathcal{X}$,*

$$L_x : \mathcal{H} \to \mathbb{R} : f \mapsto f(x). \tag{4.26}$$

Evaluation functionals are elements of the dual space of $\mathcal{H}$, since they are linear functionals from $\mathcal{H}$ to $\mathbb{R}$.

Definition 63. (Reproducing kernel Hilbert space - RKHS) *Let L_x denote the evaluation functional for the point $x \in \mathcal{X}$. According to Akhiezer and Glazman (1993), we call a Hilbert space of functions $\mathcal{H}$ a reproducing kernel Hilbert space (RKHS) if all evaluation functionals L_x are continuous or, equivalently, if for all $x \in \mathcal{X}$, L_x is a bounded operator, i.e.*

$$\forall x \in \mathcal{X} : L_x \in \mathcal{H}' : \exists M > 0 : \forall f \in \mathcal{H} : |L_x[f]| \leq M \|f\|_{\mathcal{H}}. \tag{4.27}$$

In the following, we will learn a more intuitive, however equivalent, way to characterize RKHS.

Definition 64. (Reproducing kernel) *A reproducing kernel (r.k.) is a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, which satisfies the properties:*

1. *$k(\cdot, x) \in \mathcal{H}$ for any $x \in \mathcal{X}$ and*
2. *the reproducing property, namely $\forall f \in \mathcal{H} : \forall x \in \mathcal{X} : f(x) = \langle f, k(\cdot, x) \rangle_{\mathcal{H}}$.*

Every Hilbert space with a r.k. is a RKHS.

Theorem 65. (RKHS and r.k.) *For a Hilbert space of real-valued functions $\mathcal{H}$ the following statements are equivalent:*

1. *$\mathcal{H}$ has a reproducing kernel.*
2. *$\mathcal{H}$ is a reproducing kernel Hilbert space.*

Proof. The following proof is similar to the one by Tan (2014).

(1.) $\Rightarrow$ (2.): Let k denote the reproducing kernel of $\mathcal{H}$. Then, for an arbitrary $x \in \mathcal{X}$, consider the evaluation functional L_x. We want to show that L_x is bounded, therefore we consider $|L_x[f]|$ for an arbitrary function $f \in \mathcal{H}$. The statement follows from the derivation

$$\begin{aligned} |L_x[f]| &= |f(x)| && \text{(by definition)} \\ &= |\langle f, k(\cdot, x) \rangle_{\mathcal{H}}| && \text{(reproducing property, Def. 64)} \\ &\leq \|f\|_{\mathcal{H}} \underbrace{\|k(\cdot, x)\|_{\mathcal{H}}}_{=:M>0} && \text{(Cauchy-Schwarz inequality).} \end{aligned} \tag{4.28}$$

$\mathcal{H}$ is a RKHS, i.e. all evaluation functionals are bounded operators, since $f \in \mathcal{H}$ and $x \in \mathcal{X}$ were chosen arbitrarily and the fact that M does not depend on f.

(2.) $\Rightarrow$ (1.): Let $\mathcal{H}$ be a RKHS, then we know that for every point $x \in \mathcal{X}$ the evaluation functional L_x is bounded and an element of $\mathcal{H}'$. Therefore the isomorphism $\mathcal{H} \simeq \mathcal{H}'$, defined by the Riesz representation theorem (Theorem 60), allows us to find an element $k_x \in \mathcal{H}$ with

$$L_x = \langle k_x, \cdot \rangle_{\mathcal{H}} \tag{4.29}$$

for every $x \in \mathcal{X}$. Elements of $\mathcal{H}$ are functions from $\mathcal{X}$ to $\mathbb{R}$ by definition and therefore $k_x(y)$ can be written as

$$k_x(y) = L_y[k_x] = \langle k_y, k_x \rangle_{\mathcal{H}}. \tag{4.30}$$

Subsequently, using the symmetry of the inner product for a mathematically convenient notation, we can define a function

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} : (x, y) \mapsto \langle k_x, k_y \rangle_{\mathcal{H}}, \tag{4.31}$$

that satisfies

1. $k(\cdot, x) \in \mathcal{H}$ for any $x \in \mathcal{X}$ and
2. the reproducing property, namely $\forall f \in \mathcal{H} : \forall x \in \mathcal{X} : f(x) = \langle f, k(\cdot, x) \rangle_{\mathcal{H}}$.

Consequently k is a reproducing kernel. □

The uniqueness of the r.k. corresponding to a RKHS can be shown without much additional effort.

Theorem 66. (Unique correspondence) *If for a Hilbert space $\mathcal{H}$ a r.k. exists, it will be unique.*

Proof. Let k and $\hat{k}$ be reproducing kernels of a Hilbert space $\mathcal{H}$, by exploiting the symmetry and reproducing property of k and $\hat{k}$, the identity $k = \hat{k}$, can be shown. The derivation

$$\begin{aligned} k(x,y) &= \left\langle k(\cdot,y), \hat{k}(\cdot,x) \right\rangle_{\mathcal{H}} && \text{(Definition 64)} \\ &= \left\langle \hat{k}(\cdot,x), k(\cdot,y) \right\rangle_{\mathcal{H}} && \text{(symmetry of inner product)} \\ &= \hat{k}(y,x) && \text{(Definition 64)} \\ &= \left\langle \hat{k}(\cdot,y), \hat{k}(\cdot,x) \right\rangle_{\mathcal{H}} && \text{(Definition 64)} \\ &= \hat{k}(x,y) && \text{(symmetry i.p. \& Def. 64),} \end{aligned} \tag{4.32}$$

which is obtained by repeatedly applying of the reproducing properties of k and $\hat{k}$ and the symmetry of the inner product, is true for all $x, y \in \mathcal{X}$. Thus, Derivation 4.32 proves the statement $k = \hat{k}$. □

4.2.4 Positive Definite Kernels

After seeing the appealing properties of reproducing kernel Hilbert spaces, we would like to access them in a convenient way. Therefore, we study a class of bivariate functions, namely positive definite kernels, and show that there is a one-to-one correspondence between positive definite kernels and RKHSs.

Definition 67. (Positive definite kernel) *A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a positive definite kernel (p.d.) on $\mathcal{X}$ if*

$$\forall n \in \mathbb{N}, x_1, \ldots, x_n \in \mathcal{X}, c_1, \ldots, c_n \in \mathbb{R} : \sum_{i,j=1}^{n} c_i c_j k(x_i, x_j) \geq 0 \tag{4.33}$$

holds. Positive definite kernels are often referred to as Mercer kernels, because Mercer also considered positive definite kernels in Mercer (1909).

The following theorem connects RKHSs and p.d. kernels.

Theorem 68. (Moore-Aronszajn Theorem) *Aronszajn (1950) has shown that*

1. *for every RKHS, a corresponding p.d. kernel exists and*
2. *that for every p.d. kernel k on $\mathcal{X} \times \mathcal{X}$, a (unique) RKHS exists.*

Proof. **(1.):** For a RKHS $\mathcal{H}$ we have already got to know a good candidate function that could be a p.d. kernel, namely the reproducing kernel function

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} : \langle k_x, k_y \rangle_{\mathcal{H}}, \tag{4.34}$$

where k_x is the element in $\mathcal{H}$ with $L_x = \langle k_x, \cdot \rangle_{\mathcal{H}}$. k is *symmetric*, by definition, because every inner product is symmetric, when the corresponding field is $\mathbb{R}$. Let n be a natural number, $x_1, \ldots, x_n \in \mathcal{X}$ and $c_1, \ldots, c_n \in \mathbb{R}$, then the *positive definiteness* follows from the derivation

$$\begin{aligned} \sum_{i,j=1}^{n} c_i c_j k(x_i, x_j) &= \sum_{i,j=1}^{n} c_i c_j \left\langle k_{x_i}, k_{x_j} \right\rangle_{\mathcal{H}} && \text{(Equation 4.34)} \\ &= \left\langle \sum_{i=1}^{n} c_i k_{x_i}, \sum_{j=1}^{n} c_j k_{x_j} \right\rangle_{\mathcal{H}} && \text{(bilinearity of inner product)} \\ &\geq 0 && \text{(p.d. of inner product).} \end{aligned} \tag{4.35}$$

Therefore, the reproducing kernel k, defined by Equation 4.34 is a symmetric, positive definite kernel.

(2.): Now we want to proof that - given a p.d. kernel k - we can construct a RKHS. The construction used will be similar to the one by Hofmann et al. (2006). Note that for every $x \in \mathcal{X}$ the function $k(\cdot, x)$ has domain $\mathcal{X}$ and range $\mathbb{R}$. So far, we have always considered RKHSs of functions from $\mathcal{X}$ to $\mathcal{R}$. It is obvious that the set

$$\mathcal{H}_{pd} := \{\sum_{i=1}^{n} \alpha_i k(\cdot, x_i) : n \in \mathbb{N}, \alpha_1, \ldots, \alpha_n \in \mathbb{R} \textit{ and } x_1, \ldots, x_n \in \mathcal{X}\} \tag{4.36}$$

only contains functions from $\mathcal{X}$ to $\mathbb{R}$. It is straightforward to prove that $\mathcal{H}_{pd}$ is a $\mathbb{R}$-vector space, by checking the vector space axioms. We skip this part and claim that the vector space axioms follow from the linearity of the sum. In order to make $\mathcal{H}_{pd}$ a Hilbert space, we need an inner product. It can be shown that the function

$$\langle \cdot, \cdot \rangle_{\mathcal{H}_{pd}} : \mathcal{H}_{pd} \times \mathcal{H}_{pd} \to \mathbb{R} : (f, g) \mapsto \sum_{i=1}^{m} \sum_{j=1}^{n} \alpha_i \beta_j k(x_i, x_j), \tag{4.37}$$

for f and g in $\mathcal{H}_{pd}$, with the representations

$$f = \sum_{i=1}^{m} \alpha_i k(\cdot, x_i), \text{ where } m \in \mathbb{N}, \alpha_1, \ldots, \alpha_n \in \mathbb{R} \textit{ and } x_1, \ldots, x_m \in \mathcal{X} \tag{4.38}$$

and

$$g = \sum_{j=1}^{n} \beta_j k(\cdot, x_j), \text{ where } n \in \mathbb{N}, \beta_1, \ldots, \beta_n \in \mathbb{R} \textit{ and } x_1, \ldots, x_n \in \mathcal{X}, \tag{4.39}$$

defines an inner product on $\mathcal{H}_{pd}$. Most inner product axioms can be checked quickly:

1. $\langle \cdot, \cdot \rangle_{\mathcal{H}_{pd}}$ is symmetric, because the multiplication and addition in $\mathbb{R}$ are commutative and k is symmetric.

2. The bilinearity of $\langle \cdot, \cdot \rangle_{\mathcal{H}_{pd}}$ also follows from the distributivity of multiplication and addition in $\mathbb{R}$.

3. The positive definiteness, i.e. $\langle f, f \rangle_{\mathcal{H}_{pd}} \geq 0$ for all $f \in \mathcal{H}_{pd}$ follows directly from the positive definiteness of k.

It remains to be shown that $\langle f, f \rangle_{\mathcal{H}_{pd}} = 0$ implies $f = 0$, which would be straightforward if k

would have the reproducing property. If k was a reproducing kernel, the derivation

$$\begin{aligned}
0 &\leq f(x)^2 && \text{(Trick 17)}\\
&= \langle f, k(\cdot,x)\rangle^2_{\mathcal{H}_{pd}} && \text{(reproducing property, Def. 64)}\\
&= (\|f\|_{\mathcal{H}_{pd}}\|k(\cdot,x)\|_{\mathcal{H}_{pd}})^2 && \text{(Cauchy-Schwarz inequality)}\\
&= \langle f, f\rangle_{\mathcal{H}_{pd}} \langle k(\cdot,x), k(\cdot,x)\rangle_{\mathcal{H}_{pd}} && \text{(norm induced by inner product)}\\
&= \langle f, f\rangle_{\mathcal{H}_{pd}} \underbrace{k(x,x)}_{>0 \text{ for some } x} && \text{(reproducing property, Def. 64)}
\end{aligned} \tag{4.40}$$

would imply that $f(x) = 0$ only if $\langle f, f\rangle = 0$, for all $x \in \mathcal{X}$.

In order to complete the proof that $\mathcal{H}_{pd}$ is a Hilbert space, the completeness of $\mathcal{H}_{pd}$ with respect to the metric induced by $\langle\cdot,\cdot\rangle_{\mathcal{H}_{pd}}$ remains to be shown. According to Definition 53, a metric space $\mathcal{M}$ is complete if every Cauchy sequence in $\mathcal{M}$ converges in $\mathcal{M}$. Let $(f_i)_{i\in\mathbb{N}}$ be a Cauchy sequence in $\mathcal{H}_{pd}$, then, by Equation 4.40, we have that

$$(f_r(x) - f_s(x))^2 \leq \|f_r - f_s\|^2_{\mathcal{H}_{pd}} k(x,x), \forall r, s \in \mathbb{N} \text{ and } x \in \mathcal{X}. \tag{4.41}$$

Therefore, $(f_i)_{i\in\mathbb{N}}$ converges to a real-valued function on $\mathcal{X}$ and adding the limits of all Cauchy sequences and extending the definition of the inner product accordingly would complete $\mathcal{H}_{pd}$ [1].

Finally, we will show that k is really a reproducing kernel of $\mathcal{H}_{pd}$, then, by Theorem 65, we will know that $\mathcal{H}_{pd}$ is a reproducing kernel Hilbert space. First, we note that

$$\langle k(\cdot,x), k(\cdot,y)\rangle_{\mathcal{H}_{pd}} = k(x,y) \tag{4.42}$$

for x and y in $\mathcal{X}$. Next we consider $f(x)$ for an arbitrary $f \in \mathcal{H}$,

$$\begin{aligned}
f(x) &= \sum_{i=1}^{m} \alpha_i k(x, x_i) && \text{(Equation 4.38)}\\
&= \sum_{i=1}^{m} \alpha_i \langle k(\cdot,x), k(\cdot,x_i)\rangle_{\mathcal{H}_{pd}} && \text{(Equation 4.42)}\\
&= \left\langle k(\cdot,x), \sum_{i=1}^{m} \alpha_i k(\cdot,x_i)\right\rangle_{\mathcal{H}_{pd}} && \text{(bilinearity of inner product)}\\
&= \langle k(\cdot,x), f\rangle_{\mathcal{H}_{pd}} && \text{(Definition 4.37).}
\end{aligned} \tag{4.43}$$

According to Equation 4.34, k satisfies the reproducing property. Additionally, $k(\cdot,x)$ is an element of $\mathcal{H}_{pd}$ for all $x \in \mathcal{X}$. Therefore the p.d. kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a reproducing kernel of $\mathcal{H}_{pd}$. □

Theorem 68 allows for the convenient access and construction of reproducing kernel Hilbert spaces.

Remark 69. (Construction of RKHS) *Every operation on and between p.d. kernels that preserves their positive definiteness results in another p.d. kernel, which corresponds to a different RKHS. For the p.d. kernels*

$$k_1 : \mathcal{A} \times \mathcal{A} \mapsto \mathbb{R} \tag{4.44}$$

and

$$k_2 : \mathcal{B} \times \mathcal{B} \mapsto \mathbb{R}, \tag{4.45}$$

[1] For an extensive version of this step please have a look at Hofmann et al. (2006).

defined on the sets $\mathcal{A}$ and $\mathcal{B}$, the point-wise addition and multiplication of the p.d. kernels,

$$k_+ : (\mathcal{A} \times \mathcal{B}) \times (\mathcal{A} \times \mathcal{B}) \mapsto \mathbb{R} : ((a_1, b_1), (a_2, b_2)) \mapsto k_1(a_1, a_1) + k_2(b_1, b_2) \tag{4.46}$$

and

$$k_* : (\mathcal{A} \times \mathcal{B}) \times (\mathcal{A} \times \mathcal{B}) \mapsto \mathbb{R} : ((a_1, b_1), (a_2, b_2)) \mapsto k_1(a_1, a_1) * k_2(b_1, b_2), \tag{4.47}$$

yield p.d. kernels defined on the Cartesian product of the sets $\mathcal{A} \times \mathcal{B}$. Therefore, using the point-wise addition or multiplication of p.d. kernels RKHSs for almost arbitrary input spaces can be obtained[2].

4.2.5 Feature Space Mappings

Feature space mapping A feature space mapping ϕ from $\mathcal{X}$ to a Hilbert space $\mathcal{H}_\phi$ is defined by a set of component functions $\{\phi_1, \dots \phi_d \in \mathbb{R}^{\mathcal{X}}\}$, sometimes called a dictionary, where $d \in \mathbb{N} \cup \infty$.

When we studied the kernel trick we have derived positive definite kernels from feature maps ϕ, namely by considering inner products in feature space.

Proposition 70. (Feature map $\Rightarrow$ p.d. kernel) *For every feature map $\phi : \mathcal{X} \to \mathcal{H}_\phi$ the bivariate function*

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} : (x, y) \mapsto \langle \phi(x), \phi(y) \rangle_{\mathcal{H}_\phi} \tag{4.48}$$

is a p.d. kernel.

Proof. The function given by Equation 4.48 is symmetric, by the definition of the inner product, and positive definite, since

$$\begin{aligned} \sum_{i,j=1}^{n} c_i c_j k(x_i, x_j) &= \sum_{i,j=1}^{n} c_i c_j \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}_\phi} && \text{(Definition 4.48)} \\ &= \left\langle \sum_{i=1}^{n} c_i \phi(x_i), \sum_{j=1}^{n} c_j \phi(x_j) \right\rangle_{\mathcal{H}_\phi} && \text{(bilinearity of inner product)} \\ &>= 0 && \text{(p.d. of inner product)} \end{aligned} \tag{4.49}$$

holds for all $n \in \mathbb{N}, x_1, \dots, x_n \in \mathcal{X}$ and $c_1, \dots, c_n \in \mathbb{R}$. □

As a consequence of Theorem 70, a unique RKHS exists by the Moore-Aronszajn Theorem 68. Earlier, in Equation 3.29, we denoted that space as $\mathscr{H}_k$. Additionally, it is easy to see that the mapping

$$\phi : \mathcal{X} \to \mathcal{H} : x \mapsto k(\cdot, x) \tag{4.50}$$

is a feature space mapping, for a reproducing kernel k with RKHS $\mathcal{H}$.

In order to close the loop and to ultimately prove that the kernel trick is legit, it only remains to be shown that the hypothesis space of linear functions in the feature space $\mathscr{H}_\phi$ and the RKHS $\mathscr{H}_k$ are equivalent.

[2] A the proof of the positive definiteness of k_+ and k_* and a list of other operations that preserve the positive definiteness of kernels are given by Bishop (2006).

Proposition 71. ($\mathscr{H}_\phi$ is a RKHS) *Given a feature space mapping $\phi : \mathcal{X} \to \mathcal{H}_\phi$ the set*

$$\begin{aligned}\mathscr{H}_\phi &= \{g : \mathcal{X} \to \mathbb{R} : g = f \circ \phi, f \in \mathcal{H}'_\phi\} \\ &= \{g : \mathcal{X} \to \mathbb{R} : g = \langle w, \phi(\cdot)\rangle_{\mathcal{H}_\phi},\ \textit{with } w = \psi^{-1}(f) \textit{ and } f \in \mathcal{H}'_\phi\} \\ &\textit{(Theorem 60)}\end{aligned} \tag{4.51}$$

is a RKHS.

Proof. By definition of the feature space mapping ϕ (Definition 18), $\mathcal{H}_\phi$ is a Hilbert space. According to Remark 61, the dual of a Hilbert space is a Hilbert space. Therefore $\mathcal{H}'_\phi$ is a Hilbert space. Elements of $\mathscr{H}_\phi$ are compositions of ϕ and functions $\mathcal{H}'_\phi$, i.e. $g \in \mathscr{H}_\phi$ can be written as $f \circ \phi$, with $f \in \mathcal{H}'_\phi$. Together with the addition,

$$+ : \mathscr{H}_\phi \times \mathscr{H}_\phi \to \mathscr{H}_\phi : (g, \hat{g}) \mapsto (f + \hat{f}) \circ \phi, \tag{4.52}$$

the scalar multiplication

$$\cdot : \mathbb{R} \times \mathscr{H}_\phi \to \mathscr{H}_\phi : (\lambda, g) \mapsto (\lambda f) \circ \phi \tag{4.53}$$

and the inner product

$$\langle \cdot, \cdot \rangle_{\mathscr{H}_\phi} : \mathscr{H}_\phi \times \mathscr{H}_\phi \to \mathbb{R} : (g, \hat{g}) \mapsto \left\langle f, \hat{f} \right\rangle_{\mathcal{H}'_\phi}, \tag{4.54}$$

where $g = f \circ \phi$ and $\hat{g} = \hat{f} \circ \phi$, the space $\mathscr{H}_\phi$ is a Hilbert space. Again instead of checking all vector space axioms, we claim that the defined addition, Equation 4.52, and scalar multiplication, Equation 4.53, inherit the linear structure from the addition and scalar multiplication in $\mathcal{H}'_\phi$. Analogously, we claim that the inner product properties are inherited from the inner product in $\mathcal{H}'_\phi$.

In order to show that $\mathscr{H}_\phi$ is a RKHS, we check whether the evaluation functionals are bounded, i.e. whether

$$\forall x \in \mathcal{X} : L_x \in \mathscr{H}'_\phi : \exists M > 0 : \forall g \in \mathscr{H}_\phi : |L_x[g]| \leq M \|g\|_{\mathscr{H}_\phi} \tag{4.55}$$

is satisfied. For an arbitrary element $x \in \mathcal{X}$ and an arbitrary $f \in \mathscr{H}_\phi$ consider

$$\begin{aligned}|L_x[g]| &= |g(x)| && \text{(Definition 62)} \\ &= |f(\phi(x))| && \text{(Definition 3.28)} \\ &= \left|\langle w, \phi(x)\rangle_{\mathcal{H}_\phi}\right| && \text{(Equation 4.51)} \\ &\leq \|w\|_{\mathcal{H}_\phi} \underbrace{\|\phi(x)\|_{\mathcal{H}_\phi}}_{=:M} && \text{(Cauchy-Schwarz inequality)} \\ &= M\|\psi^{-1}(f)\|_{\mathcal{H}_\phi} && \text{(Theorem 60)} \\ &= M\|f\|_{\mathcal{H}'_\phi} && \text{(Remark 61)} \\ &= M\|g\|_{\mathscr{H}_\phi} && \text{(Definition 4.54)},\end{aligned} \tag{4.56}$$

which implies that every evaluation functional is bounded. □

Proposition 72. ($\mathscr{H}_k \Leftrightarrow \mathscr{H}_\phi$) *Given a feature space mapping $\phi : \mathcal{X} \to \mathcal{H}_\phi$ the RKHS*

$$\mathscr{H}_\phi = \{g : \mathcal{X} \to \mathbb{R} : g = f \circ \phi, f \in \mathcal{H}'_\phi\} \tag{4.57}$$

and the RKHS $\mathscr{H}_k$

$$\mathscr{H}_k = \{g : \mathcal{X} \to \mathbb{R} : g = \sum_{i=1}^{n} \alpha_i k(x_i, \cdot),\ \textit{for } n \in \mathbb{N}, x_1, \ldots, x_n \in \mathcal{X}, \alpha_1, \ldots \alpha_n \in \mathbb{R}\}, \tag{4.58}$$

corresponding to the p.d. kernel $k := \langle \phi(\cdot), \phi(\cdot)\rangle_{\mathcal{H}_\phi}$, are equivalent.

Proof. The equivalence of two sets can be proved by showing that they contain each other. Let's start with the simple direction.

"⊂": We consider $g \in \mathscr{H}_k$, with the representation

$$g = \sum_{i=1}^{n} \alpha_i k(\cdot, x_i), \tag{4.59}$$

where $n \in \mathbb{N}, \alpha_i \in \mathbb{R}$ and $x_i \in \mathcal{X}$ for $i \in \{1, \dots, n\}$. By using the definition of k and the bilinearity and symmetry of the inner product,

$$\begin{aligned} g &= \sum_{i=1}^{n} \alpha_i k(\cdot, x_i) \\ &= \sum_{i=1}^{n} \alpha_i \langle \phi(\cdot), \phi(x_i) \rangle_{\mathcal{H}_\phi} \quad \text{(Equation 4.48)} \\ &= \langle \phi(\cdot), \sum_{i=1}^{n} \alpha_i \phi(x_i) \rangle_{\mathcal{H}_\phi} \quad \text{(bilinearity of inner product)} \\ &= \langle \underbrace{\sum_{i=1}^{n} \alpha_i \phi(x_i)}_{\in \mathcal{H}_\phi}, \phi(\cdot) \rangle_{\mathcal{H}_\phi} \quad \text{(symmetry of the inner product)}, \end{aligned} \tag{4.60}$$

we observe that $g \in \mathscr{H}_\phi$. g was chosen arbitrarily, which implies $\mathscr{H}_k \subset \mathscr{H}_\phi$.

"⊃": Now we consider $h \in \mathscr{H}_\phi$ of the form

$$h = \langle w, \phi(\cdot) \rangle, \tag{4.61}$$

in which $w \in \mathcal{H}_\phi$. Because $\mathcal{H}_\phi$ is a Hilbert space, w can be represented as linear combination

$$w = \sum_{j=1}^{m} \beta_j v_j, \tag{4.62}$$

in which $v_1, \dots, v_m$ are elements of $\mathcal{H}_\phi$. Given that every $v_j \in \mathcal{H}_\phi$ can be written as $v_j = \phi(x_j)$ for $j \in \{1, \dots, m\}$, the statement $\mathscr{H}_\phi \subset \mathscr{H}_k$ follows from Derivation 4.60. □

All in all we have shown that every RKHS has a unique reproducing kernel and that the equivalence chain "reproducing kernel ⇔ positive definite kernel ⇔ kernel defined by feature map" holds. In addition, we proved an alternative equivalent way from feature space mapping to RKHS, namely by considering linear hyperplanes in the feature space. Hence, all three perspectives illustrated in Figure 4.1 are equivalent.

Due to of the correspondence between kernels and inner products in feature spaces, kernel functions are often interpreted as similarity functions. The inner-product is closely related to the angle between the two vectors of interest

$$cos(\alpha) = \frac{\langle u, v \rangle}{\|u\| \|v\|}, \tag{4.63}$$

where u and v are elements of the inner-product space and α denotes the angle between them. Looking at corresponding kernel function

$$k(u, v) = cos(\alpha) \|u\| \|v\| \tag{4.64}$$

from that point of view reveals that the kernel function reaches its maximal value when the angle between the two vectors is equal to zero, in other words when the cosine of the angle is equal to one.

4.2.6 Mercer Theorem - a Fourth View

Let's briefly consider another way to find a feature map corresponding to a p.d. kernel, namely the "integral equations"-way. As a side effect we thereby show that the mapping $x \mapsto k(\cdot, x)$ defined in Section 4.2.5 is not unique. According to the Mercer Theorem for every positive definite kernel, a feature space/RKHS can be found. Recall the concept of linear maps.

Definition 73. (Linear map) *A linear map f is a mapping $f : \mathcal{V} \to \mathcal{W}$ between two $\mathbb{K}$-vector spaces $\mathcal{V}$ and $\mathcal{W}$ that fulfills the following properties: Let $x, y \in \mathcal{V}$ and $\lambda \in K$.*

1. $f(x + y) = f(x) + f(y)$
2. $f(\lambda x) = \lambda f(x)$
3. $f(0) = 0$

In the case that $\mathcal{V} = \mathcal{W}$ the map is called linear operator or endomorphism of V.

The following version of the Mercer theorem is taken form Gu (2008).

Theorem 74. (Mercer Theorem) *Suppose k is a continuous symmetric non-negative definite kernel and*

$$T_k : \mathbf{L}^2([a, b]) \to \mathbf{L}^2([a, b]) : \phi \mapsto \int_{[a,b]} k(\cdot, s)\phi(s)ds \tag{4.65}$$

is a linear operator on functions. Then there is an orthonormal basis $(e_1, \ldots, e_k)$ of $L^2[a, b]$ consisting of eigenfunctions $\{e_i\}_{i=1}^{\infty}$ of T_k such that the corresponding sequence of eigenvalues $\lambda_1, \ldots, \lambda_k$ is nonnegative. The eigenfunctions corresponding to non-zero eigenvalues are continuous on $[a, b]$ and k has the representation

$$k(s, t) = \sum_{i=1}^{\infty} \lambda_i e_i(s) e_i(t) \tag{4.66}$$

where the convergence is absolute and uniform.

Informally, the Mercer Theorem states that for every positive definite kernel a feature space can be found, namely by spectral decomposition of T_k. Despite the fact that in practice the basis representation of the RKHS used in the Mercer Theorem is not the one of choice, the Mercer theorem delivers the theoretical foundation or legitimation of the kernel trick. In a certain sense in the machine learning context the Mercer theorem can be seen as a moralistic one, guaranteeing that the kernel trick is a legit operation. Utilizing the result of the Mercer theorem a feature map

$$\phi : \mathcal{X} \to \mathcal{L}^2[a, b] : x \mapsto (\sqrt{\lambda_1} e_1(x), \ldots, \sqrt{\lambda_k} e_k(x))$$

can be found.

Mercer Theorem Finite Dimensional Analog

For an easier understanding of the underlying idea behind the Mercer theorem it can be helpful to consider it in a finite dimensional case. Let $\mathcal{X}$ be $\mathbb{R}^n$ with the basis $\langle x_1, \ldots, x_n \rangle$, then the kernel function k becomes a matrix $K_{ij} = k(x_i, x_j)$ and functions $\phi : \mathcal{X} \to \mathbb{R} \in \mathbb{R}^{\mathcal{X}}$ become rows $(\phi_1, \ldots, \phi_n) \in \mathbb{R}^{1xn}$. Consequently the application of a linear operator becomes a matrix vector multiplication

$$T_k[\phi] : \mathbb{R}^{\mathcal{X}} \to \mathbb{R}^{\mathcal{X}} : K\phi'.$$

In order to emphasize the similarity with the general case of the Mercer theorem it makes sense to consider

$$T_k[\phi]_i = \sum_{s=1}^{n} K_{is}\phi_s.$$

Since K is a symmetric positive definite matrix the spectral theorem can be applied to diagonalize it

$$K = Pdiag(\lambda_1, ...\lambda_n)P'.$$

Therefore

$$\begin{aligned} K_{ij} &= (Pdiag(\lambda_1, ...\lambda_n)P')_{ij} \\ &= \sum_{t=1}^{n} \lambda_t P_{ti} P_{tj}. \end{aligned}$$

To sum up by exploiting the fact that every symmetric positive definite matrix is a normal matrix ($K'K = KK'$) a orthonormal basis of $\mathcal{X}$, where K can be diagonalized, can be found by application of the spectral theorem for normal functions. After the diagonalization step it is straightforward to find a feature map

$$\psi : \mathcal{X} \to \mathbb{R}^{\mathcal{X}} : x \mapsto (\sqrt{\lambda_1} P_{1-} x, \ldots, \sqrt{\lambda_n} P_{n-} x),$$

for which the property

$$k(x, y) = \langle \psi(x), \psi(y) \rangle, x, y \in \mathcal{X}$$

holds.

4.3 RKHS and Regularized Risk Minimization

So far, we have considered the whole input space $\mathcal{X}$ as known, however, this is not the case in most machine learning problems. Typically, only a small fraction of the whole space, namely the training set, is known. At this point the question, whether the known subset of the whole space is sufficient to represent the optimal solution of the machine learning problem, which can be characterized by a loss function and a regularized risk functional, rises naturally and is answered by the representer theorem. The representer theorem basically states that - under certain conditions - the training data points are sufficient to represent the optimal solution of the machine learning problem. Therefore, the representer theorem can be utilized to determine whether a kernel version of certain machine learning problems exists or not.

Theorem 75. (Representer Theorem) *If the hypothesis space $\mathcal{H}$ is the RKHS defined by a p.d. kernel k, then each minimizer $f^* \in \mathcal{H}$ of the regularized risk functional*

$$R_{reg}[f; \mathbf{z}] := \frac{1}{m} \sum_{i=1}^{m} c(f(x_i), y_i) + \lambda \Omega(\|f\|_{\mathcal{H}}), \tag{4.67}$$

where $\mathbf{z} = (\mathbf{x}, \mathbf{y})$ is a training set of size m, $c : \mathbb{R} \times \mathcal{Y} \to [0, \infty)$ a loss function, $\lambda > 0$ a trade-off parameter and Ω a strictly monotonic increasing function, admits a representation of the form

$$f^* = \sum_{i=1}^{m} \alpha_i k(\cdot, x_i). \tag{4.68}$$

Proof. Let $\mathcal{S}$ denote the subspace given by the span of kernel evaluations at training points

$$\mathcal{S} := \{\sum_{i=1}^{m} \alpha_i k(\cdot, x_i) : \alpha_1, \ldots, \alpha_m \in \mathbb{R}\}. \tag{4.69}$$

In order to proof the representer theorem we consider the isomorphism between $\mathcal{S} \oplus \mathcal{S}^\perp$ and $\mathcal{H}$ given by Theorem 59. Consequently, every element $f \in \mathcal{H}$ can be represented as a linear combination of kernel evaluations at the training points and an orthogonal part

$$f = \underbrace{\sum_{i=1}^{m} \beta_i k(\cdot, x_i)}_{\in \mathcal{S}} + \underbrace{g}_{\in \mathcal{S}^\perp}, \tag{4.70}$$

where $\beta_1, \ldots, \beta_m \in \mathbb{R}$. It is easy to see that every function evaluation at a training point $x \in \mathbf{x}$ is independent of the orthogonal part, i.e.

$$\begin{aligned} f(x) &= \left\langle \sum_{i=1}^{m} \beta_i k(\cdot, x_i) + g, k(\cdot, x) \right\rangle_{\mathcal{H}} && \text{(reproducing property, Def. 64)} \\ &= \left\langle \sum_{i=1}^{m} \beta_i k(\cdot, x_i), k(\cdot, x) \right\rangle_{\mathcal{H}} + \underbrace{\langle g, k(\cdot, x) \rangle_{\mathcal{H}}}_{=0} && \text{(linearity of inner product)} \\ &= \left\langle \sum_{i=1}^{m} \beta_i k(\cdot, x_i), k(\cdot, x) \right\rangle_{\mathcal{H}} && (g \in \mathcal{S}^\perp \text{ and } k(\cdot, x) \in \mathcal{S}). \end{aligned} \tag{4.71}$$

Considering $\|f\|_{\mathcal{H}}$ yields $\Omega(\|\sum_{i=1}^{m} \beta_i k(\cdot, x_i)\|_{\mathcal{H}}) \leq \Omega(\|f\|_{\mathcal{H}})$, since

$$\begin{aligned} \|f\|_{\mathcal{H}} &= \|\sum_{i=1}^{m} \beta_i k(\cdot, x_i) + g\|_{\mathcal{H}} && \text{(Equation 4.70)} \\ &= \sqrt{\left\langle \sum_{i=1}^{m} \beta_i k(\cdot, x_i) + g, \sum_{i=1}^{m} \beta_i k(\cdot, x_i) + g \right\rangle_{\mathcal{H}}} && \text{(norm induced by inner product)} \\ &= \sqrt{\|\sum_{i=1}^{m} \beta_i k(\cdot, x_i)\|_{\mathcal{H}}^2 + 2 \underbrace{\left\langle \sum_{i=1}^{m} \beta_i k(\cdot, x_i), g \right\rangle_{\mathcal{H}}}_{=0} + \|g\|_{\mathcal{H}}^2} && \text{(linearity of inner product)} \\ &= \sqrt{\|\sum_{i=1}^{m} \beta_i k(\cdot, x_i)\|_{\mathcal{H}}^2 + \|g\|_{\mathcal{H}}^2} && \text{(orthogonal complement)} \\ &\geq \sqrt{\|\sum_{i=1}^{m} \beta_i k(\cdot, x_i)\|_{\mathcal{H}}^2} && (\|g\|_{\mathcal{H}}^2 \geq 0) \\ &= \|\sum_{i=1}^{m} \beta_i k(\cdot, x_i)\|_{\mathcal{H}} \end{aligned} \tag{4.72}$$

and Ω is a strictly monotonic increasing function. Hence every minimizer f^* of the regularized risk functional $R_{reg}[\cdot; \mathbf{z}]$ must live in $\mathcal{S}$. □

Chapter 5

Structured Output Learning

5.1 Introduction

Up to this point we mainly have considered real valued machine learning problems like classification and regression. We have studied methods that can handle those problems pretty well, namely the SVM for classification and a regularized version of regression. Both methods owe part of their success to the fact that non-linear hypothesis spaces can be utilized implicitly and efficiently thanks to the kernel trick.

Recently, driven by the requirements imposed by practical learning tasks as, for instance, learning tasks related to natural language processing or computational biology, a new stream of machine learning methods, the so-called structured output learning or structured output prediction methods, emerged. Many structured learning tasks involve labeling a set of *inter-related instances*. Popular example tasks are optical character recognition, part-of-speech tagging or semantic image segmentation. In optical character recognition and part-of-speech tagging a sequence of input objects $x = (x_1, \ldots, x_t)$, to be more accurate a sequence of images of characters and a sequence of words, is given and a label sequence of equal length $y = (y_1, \ldots, y_t)$, one character per image and one word category per word, is required. Clearly, in both cases there are dependencies within the output structure: in optical character recognition certain characters are more likely to occur next to other characters and in part-of-speech tagging the rules of grammar enforce certain correlations on the word categories. Similarly, in semantic image segmentation a semantic label has to be assigned to every pixel of the image and frequently the assumption is made, that close by pixels tend to have the same label. Tsochantaridis et al. (2005b) summarize cases, where the outputs describe a configuration over components, as structured learning problems with *macro-labels*. However, the macro-label case doesn't cover all structured output problems. For example, in semantic parsing for a given sequence of words a corresponding parse tree or in classification with taxonomies for an input a the corresponding path in a taxonomy tree is required.

Although there is no concise definition of structured output learning in the main literature to the best of my knowledge, the key observation that has to be made is that in every structured output learning task the outputs possess a certain kind of structure, i.e. various dependencies between the components that compose the outputs exist.

It is well known that naive solutions, involving the decomposition of the structured learning task into independent scalar-valued tasks, fail to produce viable solutions. In many cases this is caused by the cardinality of the output set. For example, if we have an alphabet Σ and macro-labels of the shape $y \in \Sigma^t$, the naive solution, i.e. to solve $|\Sigma|^t$ independent 1-vs-all classification tasks, will be infeasible if the number $|\Sigma|^t$ is sufficiently large. Even if the resulting problem is feasible the results might be poor in case there is a certain degree of correlation between the

outputs.

Despite the fact that graphical models [1] provide a framework to incorporate dependencies of various kinds, we will proceed with the "kernel methods" line of thinking and consider extensions of the binary SVM to the structured output case, by utilizing and extending the concepts that enabled us to work with arbitrary inputs. More precisely, we will consider so-called discriminant or compatibility functions that measure the compatibility between inputs and outputs as hypothesis functions, by approximating them linearly in a joint feature space into which the input-output pairs are embedded. Inference corresponds to maximizing the learned discriminant function with respect to the output variable. In order to learn discriminant functions we will have a look at the generalization of the binary SVM performed by Tsochantaridis et al. (2005b) and at simplifications.

5.2 Background

In structured output learning problems the objective is to find a function

$$f : \mathcal{X} \to \mathcal{Y}, \tag{5.1}$$

that associates an element of an arbitrary input set $\mathcal{X}$ with an element of an structured output set $\mathcal{Y}$. In the supervised learning scenario that function should minimize a risk functional

$$R[f] = \int_{\mathcal{X}\times\mathcal{Y}} c(f(x), y) d\mathbf{P}_{(\mathcal{X},\mathcal{Y})}(x, y) \tag{5.2}$$

or in most practical cases a regularized empirical risk functional

$$R_{reg}[f; \mathbf{z}] = \frac{1}{m} \sum_{i=1}^{m} c(f(x_i), y_i) + \lambda\Omega(\|f\|_{\mathcal{H}}), \tag{5.3}$$

where c is a loss function, $\mathbf{z}$ is a training set of size m and $\Omega(\|f\|_{\mathcal{H}})$ a regularizing term.

In the following we are examining the two different ways of designing suitable hypothesis functions that dominate the literature.

5.2.1 The Intuitive Approach

Continuing the line of thinking that we have used so far, namely to represent the arbitrary inputs by feature vectors which are elements of a Hilbert space and to consider linear operators within that, it would be intuitive to define an analogon to the feature space in order to represent the elements of the output set, let's say a label space, and consequently, to consider linear operators between the feature space and the label space. Let ϕ be a feature space mapping, ψ be a label space mapping and $\mathcal{H}_\phi$ and $\mathcal{H}_\psi$ the Hilbert spaces induced by ϕ and ψ, respectively; then such a linear operator would be

$$F : \mathcal{H}_\phi \to \mathcal{H}_\psi : \phi(x) \mapsto F(\phi(x)). \tag{5.4}$$

In the finite dimensional case F can be expressed by a matrix-vector multiplication

$$F(\phi(x)) := W\phi(x). \tag{5.5}$$

In case this has not become clear, please have a look at the following example.

[1] The interested reader is referred to Taskar et al. (2003), who combine the principles of the SVM and graphical models in an impressive way in order to get the best of both worlds.

Example 76. (Linear mapping) *If $\mathcal{H}_\phi := \mathbb{R}^n$ and $\mathcal{H}_\psi := \mathbb{R}$, a the row vector $w' \in \mathbb{R}^{1\times n}$ defines the linear map $\mathbb{R}^n \to \mathbb{R} : x \mapsto w'x$. Analogously, if $\mathcal{H}_\phi := \mathbb{R}^n$ and $\mathcal{H}_\psi := \mathbb{R}^m$, a linear map from $\mathbb{R}^n$ to $\mathbb{R}^m$ is defined by a matrix $W \in \mathbb{R}^{m\times n} : \mathbb{R}^n \to \mathbb{R}^m : x \mapsto Wx$.*

An important observation that has to be made here, is that even if the optimal label vector in $\mathcal{H}_\psi$ is found for a certain input $x \in \mathcal{X}$ using F, the corresponding element of the output set will still be unknown. Therefore in order to obtain an actual prediction for an input $x \in \mathcal{X}$ the pre-image problem

$$y^*(x) = \arg\min_{y\in\mathcal{Y}} \|F(\phi(x)) - \psi(y)\|_{\mathcal{H}_\psi} \tag{5.6}$$

needs to be solved. Hence, for every prediction the whole output set needs to be considered in order to find the most similar output to the predicted label vector $F(\phi(x))$. Accordingly, the functions of interest take the form

$$f : \mathcal{X} \to \mathcal{Y} : x \mapsto \arg\min_{y\in\mathcal{Y}} \|F(\phi(x)) - \psi(y)\|_{\mathcal{H}_\psi}. \tag{5.7}$$

The necessity of the solution of this pre-image problem makes structured output learning significantly harder than the real valued learning problems. One of the earliest implementations of such a scheme is the so-called *Kernel Dependency Estimation* introduced by Weston et al. (2002), where a linear mapping between a feature space induced by a similarity kernel and a label space induced by a loss function kernel is learned. Similarly, in the *Maximum Margin Regression* proposed by Szedmak et al. (2005) and in the extension of the *Minimal Norm Interpolation* made by Micchelli and Pontil (2005) separate representation spaces for inputs and outputs are considered. Despite this being an intuitive and straightforward approach to model the problem, the majority of the structured learning work focuses on a slightly more general approach.

5.2.2 The General Approach

Instead of considering separate representation spaces for inputs and outputs in the generalizations of the SVM by Tsochantaridis et al. (2005b) and Weston et al. (2007) and in the *Joint Kernel Support Estimation* introduced by Lampert and Blaschko (2009) a joint feature space considered. In order to understand why the joint feature space makes sense, we look at Equation 5.6 in more detail. Analogously to Weston et al. (2007) we consider normalized outputs only, i.e. $\|\psi(y)\|_{\mathcal{H}_\psi} = 1$ for all $y \in \mathcal{Y}$, in order to guarantee the well-posedness of the pre-image problem. A simple derivation

$$\begin{aligned}
y^*(x) &= \arg\min_{y\in\mathcal{Y}} \underbrace{\|F(\phi(x)) - \psi(y)\|_{\mathcal{H}_\psi}}_{\geq 0} \\
&= \arg\min_{y\in\mathcal{Y}} \|F(\phi(x)) - \psi(y)\|^2_{\mathcal{H}_\psi} && \text{(strict monotony of } \cdot^2\text{)} \\
&= \arg\min_{y\in\mathcal{Y}} \langle F(\phi(x)) - \psi(y), F(\phi(x)) - \psi(y)\rangle_{\mathcal{H}_\psi} && \text{(norm induced by inner product)} \\
&= \arg\min_{y\in\mathcal{Y}} \underbrace{\|F(\phi(x))\|^2_{\mathcal{H}_\psi}}_{const.} - 2\langle F(\phi(x)), \psi(y)\rangle_{\mathcal{H}_\psi} + \underbrace{\|\psi(y)\|^2_{\mathcal{H}_\psi}}_{=1} && \text{(bilinearity of the inner product)} \\
&= \arg\max_{y\in\mathcal{Y}} \langle F(\phi(x)), \psi(y)\rangle_{\mathcal{H}_\psi} && \text{(normalized outputs),}
\end{aligned} \tag{5.8}$$

shows not only that the solution to the pre-image problem is the $y \in \mathcal{Y}$ aligned with $F(\phi(x))$, but also illuminates the problem from a different angle. The expression $\langle F(\phi(x)), \psi(y)\rangle_{\mathcal{H}_\psi}$ can

be interpreted as a bilinear similarity or compatibility function

$$C : \mathcal{H}_\phi \times \mathcal{H}_\psi \to \mathbb{R} : (\phi(x), \psi(y)) \mapsto \langle F(\phi(x)), \psi(y) \rangle_{\mathcal{H}_\psi}, \tag{5.9}$$

since the function F is linear and the inner product is bilinear. As we know from Appendix 5.A a linear compatibility function

$$\tilde{C} : \mathcal{H}_\phi \otimes \mathcal{H}_\psi \to \mathbb{R}, \tag{5.10}$$

that satisfies the property $\hat{C}(\phi(x) \otimes \psi(y)) = C(\phi(x), \psi(y))$ for all $\phi(x) \in \mathcal{H}_\phi$ and $\psi(y) \in \mathcal{H}_\psi$, can be found by considering the tensor product of $\mathcal{H}_\phi$ and $\mathcal{H}_\phi$. Therefore, the back projection involved in the pre-image problem can be interpreted as a maximization of a compatibility function, which is approximated linearly in a joint representation space $\mathcal{H}_\Psi$, where $\mathcal{H}_\Psi$ is defined as the tensor product of $\mathcal{H}_\phi$ and $\mathcal{H}_\phi$ in the above example. As suggested by it's name, the compatibility function measures the compatibility of input-output pairs, i.e. evaluated at an input and the corresponding output the compatibility function should be maximal. Note that the tensor product of $\mathcal{H}_\phi$ and $\mathcal{H}_\psi$ is a rather simple example for a joint feature space. In general compatibility functions that are linear in an arbitrary joint feature space

$$\mathscr{C} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R} : (x, y) \mapsto \langle w, \Psi(x, y) \rangle_{\mathcal{H}_\Psi}, \tag{5.11}$$

where Ψ is a joint feature space mapping from $\mathcal{X} \times \mathcal{Y}$ to a Hilbert space $\mathcal{H}_\Psi$, are of interest. As a consequence, hypothesis functions that can be expressed in this fashion take the form

$$f : \mathcal{X} \to \mathcal{Y} : x \mapsto \arg\max_{y \in \mathcal{Y}} \mathscr{C}(x, y), \tag{5.12}$$

and are a true generalization of the ones in Equation 5.7. Additionally, the linear compatibility function on a joint feature space captures the prediction functions that originate from graphical model based approaches, see Nowozin and Lampert (2011). When a graphical model is used for prediction usually a so-called energy function is minimized. The corresponding compatibility function is obtained by taking the negative of the energy function.

5.2.3 Learning with Joint Feature Maps

Learning a function f of the shape given by Equation 5.12 corresponds to finding a suitable compatibility function $\mathscr{C}$, namely the minimizer of

$$\begin{aligned} R_{reg}[f; \mathbf{z}] &= \frac{1}{m} \sum_{i=1}^{m} c(f(x_i), y_i) + \lambda \Omega(\|f\|_{\mathcal{H}}) \\ &= \frac{1}{m} \sum_{i=1}^{m} c(\arg\max_{y \in \mathcal{Y}} \mathscr{C}(x_i, y), y_i) + \lambda \Omega(\|\mathscr{C}\|_{\mathscr{H}_\Psi}). \end{aligned} \tag{5.13}$$

According to Theorem 71, compatibility functions given by Equation 5.11 live in a reproducing kernel Hilbert space $\mathscr{H}_\Psi$ since they are linear forms on a Hilbert space $\mathcal{H}_\Psi$ and $\mathcal{X} \times \mathcal{Y}$ is a set. Therefore, thanks to the reproducing property every evaluation of a compatibility function $\mathscr{C}(x, y)$ can be represented by means of a reproducing kernel

$$J : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \to \mathbb{R} : ((x, y), (\hat{x}, \hat{y})) \mapsto \langle \Psi(x, y), \Psi(\hat{x}, \hat{y}) \rangle_{\mathcal{H}_\Psi}, \tag{5.14}$$

namely by considering the inner product

$$\mathscr{C}(x, y) = \langle \mathscr{C}, J(\cdot, (x, y)) \rangle_{\mathscr{H}_\Psi}. \tag{5.15}$$

The reproducing kernels corresponding to joint feature spaces like the one given by Equation 5.14 are often referred to as joint kernel functions.

Combining the above observations and following the same argumentation as in the representer theorem (Theorem 75), it is straightforward to derive[2] that the compatibility function $\mathscr{C}^*$ corresponding to the the minimizer f^* of Equation 5.13 admits a representation of the form

$$\mathscr{C}^* = \sum_{i=1}^{m} \int_{\mathcal{Y}} \beta_i(y) J(\cdot, (x_i, y)) d\mu(y), \tag{5.16}$$

where μ is a measure on $\mathcal{Y}$. Typically, in the discrete case the counting measure and in the continuous case the Lebesgue measure is used yielding

$$\mathscr{C}^* = \sum_{i=1}^{m} \sum_{y \in \mathcal{Y}} \beta_{iy} J(\cdot, (x_i, y)) \tag{5.17}$$

and

$$\mathscr{C}^* = \sum_{i=1}^{m} \int_{\mathcal{Y}} \beta_i(y) J(\cdot, (x_i, y)) d\lambda(y) \tag{5.18}$$

respectively. To make a long story short, using the notion of joint feature spaces allows for reusing most of the kernel theory introduced in Chapter 4 in a natural way. As a consequence, joint feature spaces with an easily computable scalar product are particularly interesting.

5.2.4 Designing Joint Kernels

There are various ways of designing the joint kernel functions used for structured learning algorithms. Similar to the kernels used for the real-valued problems, joint kernels can be either defined as the inner product induced by the joint feature map $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathcal{H}_\Psi$ or directly by specifying $J((x, y), (\hat{x}, \hat{y}))$. It is important to note that prior knowledge about the task – such as input-input, input-output and output-output correlations – can be included by designing the joint kernel appropriately[3].

One straightforward way of designing joint kernels is to make use of Proposition 21 of Tsochantaridis et al. (2005b), which states that the inner product in a joint feature space $\mathcal{H}_\Psi$ of the form $\mathcal{H}_\phi \otimes \mathcal{H}_\psi$ can be written as

$$\langle \Psi(x, y), \Psi(\hat{x}, \hat{y}) \rangle_{\mathcal{H}_\Psi} = \langle \phi(x), \phi(\hat{x}) \rangle_{\mathcal{H}_\phi} \langle \psi(y), \psi(\hat{y}) \rangle_{\mathcal{H}_\psi}. \tag{5.19}$$

Therefore joint kernel evaluations can be defined as the product of input kernel and output kernel evaluations.

$$J((x, y), (\hat{x}, \hat{y})) := K_{input}(x, \hat{x}) K_{output}(y, \hat{y}). \tag{5.20}$$

We already know several input kernels, therefore the final remaining issue is the definition of suitable output kernels.

Weston et al. (2002) present an elegant way of transforming certain types of loss functions into output kernels, namely by choosing the output kernel

$$K_{output}(y, \hat{y}) := \langle \psi(y), \psi(\hat{y}) \rangle_{\mathcal{H}_\psi} \tag{5.21}$$

[2]The interested reader is referred to Lafferty et al. (2004), who perform a similar derivation in order to proof a variation of the Representer Theorem for *Conditional Random Fields*.

[3]Weston et al. (2007) show how to incorporate those correlations, when designing the joint kernel.

in a way that the equality

$$\begin{aligned} c(y,\hat{y})^2 &\overset{!}{=} \|\psi(y)-\psi(\hat{y})\|^2_{\mathcal{H}_\psi} \\ &= \langle \psi(y)-\psi(\hat{y}), \psi(y)-\psi(\hat{y})\rangle && \text{(norm induced by inner product)} \\ &= K_{output}(y,y) - 2K_{output}(y,\hat{y}) + K_{output}(\hat{y},\hat{y}) && \text{(bilinearity of inner product)} \end{aligned} \tag{5.22}$$

holds. Table 5.1 summarizes some examples. The relationship given by Equation 5.22 can be easily verified.

Type	**Loss function**	**Kernel function**
Classification	$1-\mathbb{I}_{y=\hat{y}}(y,\hat{y})$	$\frac{1}{2}\mathbb{I}_{y=\hat{y}}(y,\hat{y})$
Regression	$\|y-\hat{y}\|^2$	$\langle y,\hat{y}\rangle$
"Arbitrary"	$c(y_i,y_j) := D_{ij}$	$\frac{1}{2}\left(\|D_{ij}\|^2 - \sum_{p=1}^m c_p\|D_{ip}\|^2 - \sum_{q=1}^m c_q\|D_{qj}\|^2 + -\sum_{p,q=1}^m c_pc_q\|D_{pq}\|^2\right)$

Table 5.1: Some output kernels derived from loss functions. Note that in the "arbitrary" case the coefficients must satisfy $\sum_i c_i = 1$.

5.3 Structured Support Vector Machine

When minimizing Equation 5.13 with an inappropriate error function, for example the zero-one loss

$$c_{01}(\hat{y},y) := \mathbb{I}_{y\neq\hat{y}}(\hat{y},y), \tag{5.23}$$

we face the same problem as in the real valued case, namely that in the linearly separable case possibly infinitely many indistinguishable solutions to the minimization problem exist. Therefore, generalizations of the maximum-margin approach used in the binary Support Vector Machine, see Section 2.4, are of great interest in order to derive a well-posed optimization framework for the structured output case. In the following we will comprehend the generalization of the maximum margin principle to the structured output case, where the output space $\mathcal{Y}$ is assumed to be discrete, performed by Tsochantaridis et al. (2005b).

Let $\mathcal{X}$ be an arbitrary set, $\mathcal{Y}$ a discrete set, $\Psi : \mathcal{X}\times\mathcal{Y} \to \mathcal{H}_\Psi$ a joint feature space mapping and $\mathbf{z} = (\mathbf{x},\mathbf{y})$ a training set of size m. We consider compatibility functions parametrized by $w \in \mathcal{H}_\Psi$

$$\mathscr{C}_w(x,y) := \langle w, \Psi(x,y)\rangle_{\mathcal{H}_\Psi}. \tag{5.24}$$

5.3.1 Linearly Separable Case

For compatibility functions that separate the training data the nonlinear inequalities

$$\forall i \in \{1,\dots,m\} : \max_{y\in\mathcal{Y}\setminus\{y_i\}} \mathscr{C}_w(x_i,y) \le \mathscr{C}_w(x_i,y_i), \tag{5.25}$$

expressing that the compatibility score of x_i and the correct label y_i is the highest, are satisfied. By using the definition of the maximum they are equivalent to the linear inequalities

$$\forall i \in \{1,\dots,m\} : \forall y \in \mathcal{Y}\setminus\{y_i\} : \mathscr{C}_w(x_i,y) \le \mathscr{C}_w(x_i,y_i), \tag{5.26}$$

which can be rewritten to

$$\forall i \in \{1,\dots,m\} : \forall y \in \mathcal{Y}\setminus\{y_i\} : \langle w, \Psi(x_i,y_i) - \Psi(x_i,y)\rangle_{\mathcal{H}_\Psi} \ge 0, \tag{5.27}$$

by Definition 5.24 and the bilinearity of the inner product. In order to obtain a unique w, satisfying Property 5.27, the one with the largest margin is selected. Tsochantaridis et al. (2005b) generalizes definition of the margin introduced by Vapnik (1998) (Definition 38) to the structured output case by considering the minimal difference between the correct score and the closest runner up

$$\begin{aligned}\gamma(w) &:= \min_{\substack{i\in\{1,\dots,m\},\\ y\in\mathcal{Y}\setminus\{y_i\}}} \mathscr{C}_w(x_i,y_i) - \mathscr{C}_w(x_i,y) \\ &= \min_{\substack{i\in\{1,\dots,m\},\\ y\in\mathcal{Y}\setminus\{y_i\}}} \langle w, \Psi(x_i,y_i) - \Psi(x_i,y)\rangle\,.\end{aligned} \tag{5.28}$$

Remark 77. (Generalization of the margin) *It is straightforward to verify, that Expression 5.28 is a generalization of the expression for the margin in the binary case in Definition 38, by considering the scoring function*

$$\mathscr{C}_w(x,y) := y\,\langle w, \phi(x)\rangle\,. \tag{5.29}$$

By substituting this scoring function into Expression 5.28 we get

$$\begin{aligned}\gamma(w) &= \min_{\substack{i\in\{1,\dots,m\},\\ y\in\mathcal{Y}\setminus\{y_i\}}} \mathscr{C}(x_i,y_i) - \mathscr{C}(x_i,y) \\ &= \min_{\substack{i\in\{1,\dots,m\},\\ y\in\mathcal{Y}\setminus\{y_i\}}} y_i\,\langle w,\phi(x_i)\rangle - y\,\langle w,\phi(x_i)\rangle \quad \textit{(Definition 5.29)} \\ &= \min_{\substack{i\in\{1,\dots,m\},\\ y\in\mathcal{Y}\setminus\{y_i\}}} (y_i - y)\,\langle w,\phi(x_i)\rangle \qquad \textit{(bilinearity of the inner product).}\end{aligned} \tag{5.30}$$

The observation that in the linear separable case $(y_i - y)\,\langle w,\phi(x_i)\rangle$ is equal to $2|\langle w,\phi(x_i)\rangle|$, which is the distance between $\phi(x_i)$ and the hyperplane multiplied by a constant, concludes the verification.

Using Definition 5.28, maximizing the margin in the linear separable case results in the optimization problem

$$\boxed{\begin{array}{ll}\max & \gamma(w) \\ \text{w.r.t.} & w \in \{w \in \mathcal{H}_\Psi : \|w\|_{\mathcal{H}_\Psi} = 1\}, \\ \text{s.t.} & \forall y \in \mathcal{Y}\setminus\{y_i\} : \langle w, \Psi(x_i,y_i) - \Psi(x_i,y)\rangle_{\mathcal{H}_\Psi} \geq \gamma(w), \\ & \text{for } i \in \{1,\dots,m\}.\end{array}} \tag{5.31}$$

Note that without the normalization of w the maximization problem would not be well defined.

The Optimization Problem 5.31 can be rewritten into a convex quadratic programming problem by considering the constraints

$$\begin{aligned}&\langle w, \Psi(x_i,y_i) - \Psi(x_i,y)\rangle_{\mathcal{H}_\Psi} \geq \gamma(w) \Leftrightarrow \\ &\left\langle \frac{w}{\gamma(w)}, \Psi(x_i,y_i) - \Psi(x_i,y)\right\rangle_{\mathcal{H}_\Psi} \geq 1 \Leftrightarrow \\ &\langle v, \Psi(x_i,y_i) - \Psi(x_i,y)\rangle_{\mathcal{H}_\Psi} \qquad\qquad (v := \frac{w}{\gamma(w)})\end{aligned} \tag{5.32}$$

and noting that $\gamma(w)$ is maximized when $\|v\|^2_{\mathcal{H}_\Psi}$ is minimized, because of the relationship

$$\begin{aligned}\|v\|^2_{\mathcal{H}_\Psi} &= \|\frac{w}{\gamma(w)}\|^2_{\mathcal{H}_\Psi} \\ &= \frac{1}{\gamma(w)^2}\|w\|^2_{\mathcal{H}_\Psi} \quad \text{(absolute homogeneity of the norm)} \\ &= \frac{1}{\gamma(w)^2} \qquad (\|w\|_{\mathcal{H}_\Psi} = 1).\end{aligned} \tag{5.33}$$

The resulting quadratic programming problem is

$$\boxed{\begin{array}{ll} \min & \|v\|^2_{\mathcal{H}_\Psi} \\ \text{w.r.t.} & v \in \mathcal{H}_\Psi, \\ \text{s.t.} & \forall y \in \mathcal{Y}\backslash\{y_i\} : \langle v, \Psi(x_i, y_i) - \Psi(x_i, y)\rangle_{\mathcal{H}_\Psi} \geq 1, \\ & \text{for } i \in \{1, \ldots, m\}. \end{array}} \tag{5.34}$$

Remark 78. (Continuous outputs) *Weston et al. (2007) generalize the structured SVM to continuous output spaces, by adjusting the constraints from Optimization Problem 5.34 to*

$$\forall i \in \{1, \ldots, m\} : \forall y \in \mathcal{Y} : \|y_i - y\| \geq \epsilon : \langle v, \Psi(x_i, y_i) - \Psi(x_i, y)\rangle_{\mathcal{H}_\Psi} \geq \frac{\epsilon^2}{2}. \tag{5.35}$$

5.3.2 Non-linearly Separable Case

When the training set $\mathbf{z}$ is not linearly separable slack variables can be introduced to account for the errors analogously as it has been performed in binary SVM. There are several different ways of using slack variables, for example one could either use one slack variable per linear constraint or one slack variable per non-linear constraint. Tsochantaridis et al. (2005b) focus on the latter approach. Adding a penalty term for the slacks to the objective and adjusting the constraints, leads to the optimization problem

$$\boxed{\begin{array}{ll} \min & \|v\|^2_{\mathcal{H}_\Psi} + \frac{C}{m}\sum_{i=1}^m \xi_i \\ \text{w.r.t.} & v \in \mathcal{H}_\Psi, \xi \in \mathbb{R}^m, \\ \text{s.t.} & \forall y \in \mathcal{Y}\backslash\{y_i\} : \langle v, \Psi(x_i, y_i) - \Psi(x_i, y)\rangle_{\mathcal{H}_\Psi} \geq 1 - \xi_i, \\ & \xi_i \geq 0, \text{ for } i \in \{1, \ldots, m\}, \end{array}} \tag{5.36}$$

where C controls the trade-off between margin maximization and training error minimization.

5.3.3 Arbitrary Loss Function

Despite the fact that in the linear separable case many loss functions, i.e. loss functions that satisfy $c(y, y) = 0$, don't lead to a well posed optimization problem, there is a second difficulty that arises when minimizing Equation 5.13, namely that $c(\arg\max_{y\in\mathcal{Y}} \mathscr{C}_w(x_i, y), y_i)$ is not continuous with respect to w. In the current setting $f(x) := \arg\max_{y\in\mathcal{Y}} \mathscr{C}_w(x, y)$ maps into the discrete space $\mathcal{Y}$, as a result $c(\arg\max_{y\in\mathcal{Y}} \mathscr{C}_w(x_i, y), y_i)$ is a piecewise constant function with respect to w.

Tsochantaridis et al. (2005b) present two intuitive approaches to augment the maximum margin learning framework, described by Optimization Problem 5.36, with arbitrary loss functions. One involves rescaling the slack variables and the other rescaling the margin. The sum over the optimal values for the slack variables of the corresponding optimization problems, provide upper-bounds for the empirical risk of the loss function, see Proposition 1 and Proposition 2 of Tsochantaridis et al. (2005b). We briefly outline the slack rescaling approach.

Slack rescaling

The underlying idea of the slack rescaling approach is that the penalty for the violation of a margin constraint $f(x_i) \neq y_i$ should scale with the corresponding loss $c(f(x_i), y_i)$. The amplification of the penalty ξ_i by the factor $c(f(x_i), y_i)$ is accomplished by scaling ξ_i in margin constraints with the inverse loss, which yields

$$\boxed{\begin{array}{ll} \min & \|v\|^2_{\mathcal{H}_\Psi} + \frac{C}{m}\sum_{i=1}^m \xi_i \\ \text{w.r.t.} & v \in \mathcal{H}_\Psi, \xi \in \mathbb{R}^m, \\ \text{s.t.} & \forall y \in \mathcal{Y}\backslash\{y_i\} : \langle v, \Psi(x_i, y_i) - \Psi(x_i, y)\rangle_{\mathcal{H}_\Psi} \geq 1 - \frac{\xi_i}{c(y, y_i)}, \\ & \xi_i \geq 0, \text{ for } i \in \{1, \ldots, m\}. \end{array}} \tag{5.37}$$

By considering the dual problem of Optimization Problem 5.37 and thereby noting that the optimal slacks are given by

$$\xi_i^* = \max\{0, \max_{y \neq y_i}\{c(y, y_i)(1 - \langle v, \Psi(x_i, y_i) - \Psi(x_i, y)\rangle_{\mathcal{H}_\Psi})\}\} \tag{5.38}$$

it is straightforward to show that the inequality

$$\frac{1}{m}\sum_{i=1}^m c(f(x_i), y_i) \leq \frac{1}{m}\sum_{i=1}^m \xi_i^* \tag{5.39}$$

holds, namely by showing that every summand on the left is bounded by the one on the right. The case, where $f(x_i) = y_i$, meaning that $c(f(x_i), y_i) = 0$, is trivial, since $\xi_i \geq 0$ per definition. When $f(x_i) \neq y_i$ holds on the other hand, then $\langle v, \Psi(x_i, y_i) - \Psi(x_i, f(x_i))\rangle \leq 0$ and since $f(x_i)$ is defined as $\arg\max_{y \in \mathcal{Y}} \langle v, \Psi(x_i, y)\rangle_{\mathcal{H}_\Psi}$ the equality

$$\xi_i^* = c(f(x_i), y_i)(1 - \underbrace{\langle v, \Psi(x_i, y_i) - \Psi(x_i, f(x_i))\rangle_{\mathcal{H}_\Psi}}_{\leq 0}) \tag{5.40}$$

holds. Thus $\frac{\xi_i^*}{c(f(x_i), y_i)} \geq 1$, which is equivalent to $\xi_i^* \geq c(f(x_i), y_i)$. Therefore, using the slack rescaling approach in order to incorporate arbitrary loss functions in the maximum margin framework, leads to the thought of exchanging the badly-behaving loss function with a well-behaving one.

Margin rescaling

Alternatively, the margin can be rescaled instead of the slack variables, resulting in

$$\boxed{\begin{array}{ll} \min & \|v\|^2_{\mathcal{H}_\Psi} + \frac{C}{m}\sum_{i=1}^m \xi_i \\ \text{w.r.t.} & v \in \mathcal{H}_\Psi, \xi \in \mathbb{R}^m, \\ \text{s.t.} & \forall y \in \mathcal{Y}\backslash\{y_i\} : \langle v, \Psi(x_i, y_i) - \Psi(x_i, y)\rangle_{\mathcal{H}_\Psi} \geq c(y, y_i) - \xi_i, \\ & \xi_i \geq 0, \text{ for } i \in \{1, \ldots, m\}. \end{array}} \tag{5.41}$$

The average over the optimal slacks of Optimization Problem 5.41 provides an upper bound to the empirical risk as well, see Tsochantaridis et al. (2005b) Proposition 2. Note that instead of the constraints

$$\forall y \in \mathcal{Y}\backslash\{y_i\} : \langle v, \Psi(x_i, y_i) - \Psi(x_i, y)\rangle_{\mathcal{H}_\Psi} \geq c(y, y_i) - \xi_i \tag{5.42}$$

the constraints

$$\forall y \in \mathcal{Y} : \langle v, \Psi(x_i, y_i) - \Psi(x_i, y)\rangle_{\mathcal{H}_\Psi} \geq c(y, y_i) - \xi_i \tag{5.43}$$

can be considered, since $\langle v, \Psi(x_i, y_i) - \Psi(x_i, y_i)\rangle_{\mathcal{H}_\Psi}$ is equal to zero and thereby greater or equal $-\xi_i$.

5.3.4 Simplifications

Optimization Problem 5.34 is hard to solve, not least because of a training set of size m and an output space of size $|\mathcal{Y}|$, the number of constraints is $m(|\mathcal{Y}|-1)$. In order to solve the problem efficiently, typically a variety of assumptions are made. Among others, the strongest assumption that is made for example by Tsochantaridis et al. (2005b) to solve the problem efficiently, is the existence of an algorithm that solves the pre-image problem in polynomial time. Corollary 1 of Proposition 1 of Gärtner and Vembu (2009) shows that for a large class of output set - hypothesis space pairs $(\mathcal{Y}, \mathscr{H})$ this assumption is violated.

This result suggests to put effort into simplifying the problem. For example Gärtner and Vembu (2009) considered hypotheses that are linear in a tensor product space together with output sets for which $|\mathcal{Y}|, \sum_{y\in\mathcal{Y}} \psi(y)$ and $\sum_{y\in\mathcal{Y}} \psi(y)\psi(y)'$ can be computed efficiently. Alternatively, similar optimization problems, that carry the spirit of maximum margin learning but are more efficient to solve, can be derived by relaxing the constraints of the structured SVM optimization problems.

Example 79. (Relaxation of the Constraints)
Let $\mathcal{Y} = \{-1,+1\}^n$, $\Psi(x_i, y)$ be defined as $\phi(x_i) \otimes y$ and $c(y, \hat{y})$ be the Hamming distance

$$c(y,\hat{y}) = \sum_{i=1}^{n} \mathbb{I}_{(y\neq\hat{y})}(y,\hat{y}). \tag{5.44}$$

The Structured SVM constraints of Optimization Problem 5.41 can be written as

$$\langle v, \Psi(x_i,y_i) - \Psi(x_i,y)\rangle_{\mathcal{H}_\Psi} \geq c(y_i,y) - \xi_i, \forall i \in \{1,\dots,m\}, y \in \mathcal{Y}. \tag{5.45}$$

If the constraints in Optimization Problem 5.45 are satisfied, the constraints

$$\langle v, \Psi(x_i,y_i) - \Psi(x_i,y)\rangle_{\mathcal{H}_\Psi} \geq \tfrac{c(y_i,y)}{C_i} - \xi_i, \forall i \in \{1,\dots,m\}, y \in \mathcal{Y}, \tag{5.46}$$

where C_i is a positive constant will be satisfied as well. In the following we will consider the average over all constraints, therefore choosing

$$C_i := \sum_{y\in\mathcal{Y}} c(y_i,y) = \frac{n}{2} \tag{5.47}$$

will lead to a simpler expression. When averaging over the constraints with respect to the domain $\mathcal{Y}$ the first term becomes

$$\textstyle\sum_{y\in\mathcal{Y}} \langle v, \Psi(x_i,y_i)\rangle_{\mathcal{H}_\Psi} / |\mathcal{Y}| \quad = \langle v, \Psi(x_i,y_i)\rangle_{\mathcal{H}_\Psi}, \tag{5.48}$$

the second term vanishes

$$\begin{aligned}
\sum_{y\in\mathcal{Y}} \langle v, \Psi(x_i,y)\rangle_{\mathcal{H}_\Psi} / |\mathcal{Y}| &= \sum_{y\in\mathcal{Y}} \langle v, y\otimes\phi(x_i)\rangle_{\mathcal{H}_\Psi} / |\mathcal{Y}| && \textit{(assumption)}\\
&= \langle v, (1/|\mathcal{Y}|) \sum_{y\in\mathcal{Y}} y\otimes\phi(x_i)\rangle_{\mathcal{H}_\Psi} && \textit{(bilinearity of inner product)}\\
&= \langle v, (\sum_{y\in\mathcal{Y}} y/|\mathcal{Y}|) \otimes \phi(x_i)\rangle_{\mathcal{H}_\Psi} && \textit{(bilinearity of tensor product)}\\
&= \langle v, \mathbf{0}\otimes\phi(x_i)\rangle_{\mathcal{H}_\Psi} && \textit{(symmetry of } \mathcal{Y}\textit{)}\\
&= 0
\end{aligned} \tag{5.49}$$

because $\sum_{y\in\mathcal{Y}} y = 0$, for the Hamming distance we have

$$\textstyle\sum_{y\in\mathcal{Y}} \frac{c(y_i,y)}{n/2} / |\mathcal{Y}| \quad = 1 \tag{5.50}$$

and for the slack we get

$$\sum_{y\in\mathcal{Y}} \xi_i/|\mathcal{Y}| \quad = \xi_i. \tag{5.51}$$

Putting together these terms will leads to the same constraints as in the optimization problem of the Maximum Margin Regression*, introduced by Szedmak et al. (2005), where the constraints are given by*

$$\langle y_i, v\phi(x_i)\rangle_{\mathcal{H}_\Psi} \geq 1 - \xi_i, \forall i \in \{1, \ldots, m\}, \tag{5.52}$$

because of the equality

$$\langle v, y_i \otimes \phi(x_i)\rangle_{\mathcal{H}_\Psi} = \langle y_i, v\phi(x_i)\rangle_{\mathcal{H}_\Psi}. \tag{5.53}$$

It is worth pointing out, that Szedmak et al. (2005) derived the same optimization problem by generalizing the binary SVM in a geometrical sense.

5.4 Maximum Margin Regression

The maximum margin regression (MMR), introduced by Szedmak et al. (2005), can be seen as a geometrical generalization of the two-class support vector machine to structured output spaces. Astikainen et al. (2008) use the MMR method in order to predict enzyme functions and Xiong et al. (2015) use it for image annotation.

5.4.1 Problem Formulation

In the binary SVM the objective is to find the hyperplane with the maximum margin. This hyperplane is parametrized by a linear form in the feature space. In order to consider structured outputs the outputs can be mapped into a label space. Both the feature space $\mathcal{H}_\phi$ and the label space $\mathcal{H}_\psi$ are Hilbert spaces. As a consequence instead of a linear form

$$h_w : \mathcal{H}_\phi \to \mathbb{R} : \phi(x) \mapsto w'\phi(x) \tag{5.54}$$

a linear map h_W from the feature space into the label space

$$h_W : \mathcal{H}_\phi \to \mathcal{H}_\psi : \phi(x) \mapsto W\phi(x) \tag{5.55}$$

is required. After introducing the generalization of our linear function h_w in the form of a linear mapping h_W the objective and the remaining constraints of the binary SVM optimization problem can be rewritten correspondingly. In the objective an operator norm has to be used in place of the vector norm and the constraints can be generalized using the properties of the output Hilbert space $\mathcal{H}_\psi$. Figure 5.1 summarizes these changes. Omitting the bias term, the resulting optimization problem is

$$\boxed{\begin{array}{ll} \min & \|W\|^2_{Frobenius} + C\sum_{i=1}^m \xi_i \\ \text{w.r.t.} & W : \mathcal{H}_\phi \to \mathcal{H}_\psi, \xi \in \mathbb{R}^m, \\ \text{s.t.} & \left\langle \psi(y_i), W\phi(x_i)\right\rangle_{\mathcal{H}_\psi} \geq 1 - \xi_i, \\ & \xi_i \geq 0, \text{ for } i \in \{1, \ldots, m\}. \end{array}} \tag{5.56}$$

In order to predict, the pre-image problem

$$y^* = \arg\max_{y\in\mathcal{Y}} \langle \psi(y), W\phi(x)\rangle_{\mathcal{H}_\psi} \tag{5.57}$$

has to be solved. Because of the relationship between the inner product of two vectors u and v and the angle between them

$$cos(\sphericalangle(u, v)) = \frac{\langle u, v\rangle}{\|u\|\|v\|}, \tag{5.58}$$

solving the pre-image problem corresponds to finding the y of which the label vector $\psi(y)$ is maximally aligned with the prediction $W\phi(x)$, i.e. $\sphericalangle(\psi(y), W\phi(x))$ is equal to zero.

	Binary class learning	Vector label learning
	Support Vector Machine (SVM)	Maximum Margin Regression (MMR)
min	$\frac{1}{2}\underbrace{\boxed{W^TW}}_{\|W\|_2^2}+C\mathbf{1}^T\xi$	$\frac{1}{2}\underbrace{\boxed{\mathbf{tr}(W^TW)}}_{\|W\|^2_{Frobenius}}+C\mathbf{1}^T\xi$
w.r.t.	$\boxed{w:\mathcal{H}_\phi\to\mathbb{R}}$, normal vec. $\boxed{b\in\mathbb{R}}$, bias $\xi\in\mathbb{R}^m$, error vector	$\boxed{W:\mathcal{H}_\phi\to\mathcal{H}_\psi}$, linear operator $\boxed{b\in\mathcal{H}_\psi}$, translation(bias) $\xi\in\mathbb{R}^m$, error vector
s.t.	$\boxed{y_i(w^T\phi(x_i)+b)}\geq 1-\xi_i$ $\xi\geq\mathbf{0},\ i=\{1,\dots,m\}$	$\boxed{\langle\psi(y_i), W\phi(x_i)+b\rangle_{\mathcal{H}_\psi}}\geq 1-\xi_i$ $\xi\geq\mathbf{0},\ i=\{1,\dots,m\}$

Figure 5.1: The changes in the optimization problem from SVM to MMR.

5.4.2 Kernel Version

Proposition 80. (MMR kernel version) *The dual problem of Optimization Problem 5.56 is given by*

$$\boxed{\begin{array}{ll}\min & \frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n\alpha_i\alpha_j k_{\mathcal{X}}(x_i,x_j)k_{\mathcal{Y}}(y_i,y_j)-\mathbf{1}'\alpha\\ w.r.t. & \alpha\\ s.t. & 0\leq\alpha_i\leq C, i\in\{1,\dots,n\},\end{array}} \tag{5.59}$$

where $k_{\mathcal{X}}(x_i,x_j):=\langle\phi(x_i),\phi(x_j)\rangle$ is an input kernel and $k_{\mathcal{Y}}(y_i,y_j):=\langle\psi(y_i),\psi(y_j)\rangle$ is an output kernel.

Proof. In order to obtain the kernel version the Lagrange dual function needs to be considered. The Lagrange dual function g is defined as the infimum of the Lagrangian with respect to the parameters of the objective function and gives us a lower bound for the objective function f. Let's refer to the objective function as f

$$f(W,\xi):=\frac{1}{2}\|W\|^2+C\sum_{i=1}^n\xi_i. \tag{5.60}$$

The corresponding Lagrangian function can be obtained by converting the greater-than-equal constraints into lower-than-equal constraints, by multiplying with minus one and using the definition of the Lagrangian function (Definition 2.69), resulting in

$$L(W,\xi,\alpha,\beta):=f(W,\xi)-\sum_{i=1}^n\alpha_i\langle\psi(y_i),W\phi(x_i)\rangle+\sum_{i=1}^n\alpha_i-\sum_{i=1}^n\alpha_i\xi_i-\sum_{i=1}^n\beta_i\xi_i. \tag{5.61}$$

The Lagrange dual function

$$g(\alpha,\beta):=\inf_{W,\xi}L(W,\xi,\alpha,\beta) \tag{5.62}$$

provides a lower bound on the objective function $g(\alpha, \beta) \leq f(W, \xi)$. The infimum can be found by setting the derivatives with respect to W

$$\frac{\partial L}{\partial W} \stackrel{!}{=} 0 \tag{5.63}$$

and ξ

$$\frac{\partial L}{\partial \xi} \stackrel{!}{=} 0 \tag{5.64}$$

to zero.

When using the Frobenius norm $\|W\|^2 = tr(W'W)$, the derivative with respect to W becomes

$$\frac{\partial L}{\partial W} = W - \sum_{i=1}^{n} \alpha_i \psi(y_i)\phi(x_i)', \tag{5.65}$$

because of the identities

$$\frac{\partial tr(W'W)}{\partial W} = 2W \tag{5.66}$$

and

$$\frac{\partial \langle \psi(y_i), W\phi(x_i) \rangle}{\partial W} = \frac{\partial \psi(y_i)' W \phi(x_i)}{\partial W} = \psi(y_i)\phi(x_i)'. \tag{5.67}$$

Therefore, W can be expressed in terms of α

$$W = \sum_{i=1}^{n} \alpha_i \psi(y_i)\phi(x_i)'. \tag{5.68}$$

The derivative with respect to ξ is

$$\frac{\partial L}{\partial \xi} = C\mathbf{1} - \alpha - \beta. \tag{5.69}$$

By rearranging the terms the following expression is obtained for β

$$\beta = C\mathbf{1} - \alpha. \tag{5.70}$$

The Lagrange dual function now can be written explicitly by substituting Equation 5.68 and Equation 5.70 into Equation 5.61. Without further simplifications the following expression is obtained

$$\begin{aligned} g(\alpha, \beta) = & \\ & \frac{1}{2} tr\left(\left(\sum_{i=1}^{n} \alpha_i \psi(y_i)\phi(x_i)'\right)' \left(\sum_{i=1}^{n} \alpha_i \psi(y_i)\phi(x_i)'\right)\right) \\ & + C\mathbf{1}'\xi - \sum_{i}^{n} \alpha_i \left\langle \psi(y_i), \left(\sum_{i=1}^{n} \alpha_i \psi(y_i)\phi(x_i)'\right) \phi(x_i) \right\rangle \\ & + \mathbf{1}'\alpha - \alpha'\xi - C\mathbf{1}'\xi + \alpha'\xi, \end{aligned} \tag{5.71}$$

where the terms containing ξ cancel. The dual function can be further simplified by unfolding the matrix multiplications inside the trace and inside the inner product. Let's consider the part with the trace first. Assuming that the dimensionality of the feature space $\mathcal{H}_\phi$ is d and the dimensionality of the output space $\mathcal{H}_\psi$ is k, the shape of W is $k \times d$. Therefore, the trace of $W'W$ is

$$tr(W'W) = \sum_{v=1}^{d} (W_{-v})' W_{-v}, \tag{5.72}$$

where W_{-v} denotes the v-th column of W, which is equal to

$$W_{-v} = \sum_{i=1}^{n} \alpha_i \begin{pmatrix} \psi(y_i)_1 \phi(x_i)_v \\ \vdots \\ \psi(y_i)_k \phi(x_i)_v \end{pmatrix}. \tag{5.73}$$

Substituting Equation 5.73 into Equation 5.72 and performing the vector multiplication we get

$$tr(W'W) = \sum_{v}^{d} \sum_{u}^{k} \left(\sum_{i=1}^{n} \alpha_i \psi(y_i)_u \phi(x_i)_v \cdot \sum_{j}^{n} \alpha_j \psi(y_j)_u \phi(x_j)_v \right). \tag{5.74}$$

By making use of the commutativity of the sum $a + b = b + a$ and the distributive law $a \cdot b + a \cdot c = a \cdot (b + c)$ the following expression can be obtained

$$\begin{aligned} tr(W'W) &= \sum_{i=1}^{n} \sum_{j}^{n} \alpha_i \alpha_j \left(\sum_{u}^{k} \psi(y_i)_u \psi(y_j)_u \cdot \sum_{v}^{d} \phi(x_i)_v \phi(x_j)_v \right) \\ &= \sum_{i=1}^{n} \sum_{j}^{n} \alpha_i \alpha_j \langle \phi(x_i), \phi(x_j) \rangle \langle \psi(y_i), \psi(y_j) \rangle, \end{aligned} \tag{5.75}$$

where the the inputs and the outputs only enter via inner-products. The simplification of the remaining part of the formula is similar

$$\begin{aligned} &\sum_{i=1}^{n} \alpha_i \left\langle \psi(y_i), \left(\sum_{i=1}^{n} \alpha_i \psi(y_i) \phi(x_i)' \right) \phi(x_i) \right\rangle && \text{(inner-product and matrix-multiplication)} \\ &= \sum_{i=1}^{n} \alpha_i \left(\sum_{u}^{k} \psi(y_i)_u \cdot \sum_{v}^{d} \phi(x_i)_v \cdot \sum_{j}^{n} \alpha_j \psi(y_j)_u \phi(x_j)_v \right) && \text{(commutativity and distributivity)} \\ &= \sum_{i=1}^{n} \sum_{j}^{n} \alpha_i \alpha_j \left(\sum_{u}^{k} \psi(y_i)_u \psi(y_j)_u \cdot \sum_{v}^{d} \phi(x_i)_v \phi(x_j)_v \right) && \text{(definition of inner-product)} \\ &= \sum_{i=1}^{n} \sum_{j}^{n} \alpha_i \alpha_j \langle \phi(x_i), \phi(x_j) \rangle \langle \psi(y_i), \psi(y_j) \rangle. \end{aligned} \tag{5.76}$$

Putting Equation 5.75 and Equation 5.76 back into Equation 5.71 results in

$$g(\alpha) = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \langle \phi(x_i), \phi(x_j) \rangle \langle \psi(y_i), \psi(y_j) \rangle + \mathbf{1}'\alpha, \tag{5.77}$$

where the parameter β does not occur any longer. Since the dual is a lower bound on the objective function f, it has to be maximized in order to find the best α. Maximizing g is equivalent to minimizing $(-1)g(\alpha)$, resulting in the Optimization Problem 5.59, where the box-constraints for α_i come from the Karush-Kuhn-Tucker conditions (Boyd and Vandenberghe, 2004) and from Equation 5.70, since $\alpha_i \geq 0$, $\beta_i \geq 0$ and $\alpha_i = C - \beta_i$ imply that $\alpha_i \leq C$ for $i \in \{1, \ldots, n\}$. □

Proposition 81. (MMR predictor) *The pre-image problem of the MMR takes the form*

$$y^*(x) = \arg\max_{y \in \mathcal{Y}} \sum_{i=1}^{n} \alpha_i k_{\mathcal{X}}(x_i, x) k_{\mathcal{Y}}(y_i, y) \tag{5.78}$$

Proof. Substituting the parametrization of the learned mappings, given by Equation 5.68, into the Pre-image Problem 5.57 and using the bilinearity of the inner product yields

$$\begin{aligned} y^*(x) &= \arg\max_{y \in \mathcal{Y}} \langle \psi(y), W\phi(x) \rangle && \text{(Pre-image Problem 5.57)} \\ &= \arg\max_{y \in \mathcal{Y}} \left\langle \psi(y), \sum_{i=1}^{n} \alpha_i \psi(y_i) \underbrace{\phi(x_i)'\phi(x)}_{=\langle \phi(x_i), \phi(x) \rangle} \right\rangle && \text{(Equation 5.68)} \\ &= \arg\max_{y \in \mathcal{Y}} \sum_{i=1}^{n} \alpha_i \langle \phi(x_i), \phi(x) \rangle \langle \psi(y), \psi(y_i) \rangle && \text{(bilinearity i.p.).} \end{aligned} \tag{5.79}$$

The application of the kernel trick on the last line of Derivation 5.79 leads to Equation 5.78, which closes the proof. □

Appendix

5.A The Tensor Product

Without going into too much math, we will introduce the tensor product as a powerful tool, that enables us to work with multilinear functions as if they were linear functions.

Definition 82. (Multilinear function) *Let n be in $\mathbb{N}$ and $\mathcal{V}_1, \dots, \mathcal{V}_n, \mathcal{W}$ be $\mathbb{K}$-vector spaces. A function $f : \mathcal{V}_1 \times \cdots \times \mathcal{V}_n \to \mathcal{W}$ is called multilinear, if the functions*

$$f_i : \mathcal{V}_i \to \mathcal{W} : v_i \mapsto f(v_1, \dots, v_i, \dots, v_n) \tag{5.80}$$

are linear for all $i \in \{1, \dots, n\}$ and all fixed $(n-1)$-tuples

$$(v_1, \dots, v_{i-1}, v_{i+1}, \dots, v_n) \in \mathcal{V}_1 \times \cdots \times \mathcal{V}_{i-1} \times \mathcal{V}_{i+1} \times \cdots \times \mathcal{V}_n. \tag{5.81}$$

If n is equal to two we will call a multilinear function bilinear*.*

For the sake of simplicity we consider bilinear functions for now.

Definition 83. (Tensor product) *Let $\mathcal{V}_1$ and $\mathcal{V}_2$ be $\mathbb{K}$-vector spaces. The tensor product $\mathcal{V}_1 \otimes \mathcal{V}_2$ of $\mathcal{V}_1$ and $\mathcal{V}_2$ is a $\mathbb{K}$-vector space, for which a bilinear mapping $\otimes : \mathcal{V}_1 \times \mathcal{V}_2 \to \mathcal{V}_1 \otimes \mathcal{V}_2$ exists, that fulfills the* universal property*:*

For every bilinear mapping b from $\mathcal{V}_1 \times \mathcal{V}_2$ to a $\mathbb{K}$-vector space $\mathcal{W}$

$$b : \mathcal{V}_1 \times \mathcal{V}_2 \to \mathcal{W}, \tag{5.82}$$

an uniquely determined linear mapping $\tilde{b}$ from $\mathcal{V}_1 \otimes \mathcal{V}_2$ to $\mathcal{W}$ exists

$$\tilde{b} : \mathcal{V}_1 \otimes \mathcal{V}_2 \to \mathcal{W}, \tag{5.83}$$

such that the equality

$$\tilde{b}(v_1 \otimes v_2) = b(v_1, v_2), \tag{5.84}$$

where $v_1 \otimes v_2 := \otimes(v, w)$, holds for all $v_1 \in \mathcal{V}_1$ and $v_2 \in \mathcal{V}_2$.

The tensor product space and the corresponding bilinear mapping can be constructed in various ways[4]. The resulting vector space is uniquely determined up to isomorphisms. Additionally, the definition and construction for the tensor product of more than two vector spaces can be performed analogously.

As illustrated in the following example, in the tensor product space a for every bilinear form a linear form can be found.

[4]For the construction of the tensor product we refer the reader to Halmos (1974).

Example 84. (The Kronecker product) *For finite dimensional vector spaces $\mathbb{R}^m$ and $\mathbb{R}^n$, a matrix $A \in \mathbb{R}^{m \times n}$ exists for every bilinear form $b : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$ with*

$$b(x, y) = x' A y, \text{ for all } x \in \mathbb{R}^m \text{ and } y \in \mathbb{R}^n. \tag{5.85}$$

Let m be equal to three, n be equal to two and b the bilinear form defined by the matrix $B \in \mathbb{R}^{3 \times 2}$

$$b(\begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}, \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}) = \begin{pmatrix} v_1 & v_2 & v_3 \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}, \tag{5.86}$$

for $v \in \mathbb{R}^3$ and $w \in \mathbb{R}^2$. By the definition of the matrix multiplication $b(v, w)$ can be written as a summation

$$b(v, w) = \sum_{i=1}^{3} v_i \sum_{j=1}^{2} B_{ij} w_j. \tag{5.87}$$

Using the distributivity of multiplication and addition and the commutativity of the multiplication in $\mathbb{R}$, $b(v, w)$ can be written as

$$b(v, w) = \sum_{i=1}^{3} \sum_{j=1}^{2} B_{ij} v_i w_j, \tag{5.88}$$

which already indicates, that at least one vector space that is isomorphic to the tensor product space of $\mathbb{R}^3$ and $\mathbb{R}^2$ exists, namely a subspace of $\mathbb{R}^{3 \times 2}$. The bilinear function $\otimes$, that maps $\mathbb{R}^3 \times \mathbb{R}^2$ to that space, is called Kronecker product and defined as

$$\begin{aligned} \otimes : \mathbb{R}^{m \times n} \times \mathbb{R}^{p \times r} : (A, B) &\mapsto A_{ij} \cdot B, \\ i \in \{1 \dots m\}, j &\in \{1 \dots n\}. \end{aligned} \tag{5.89}$$

The Kronecker product of $v \in \mathbb{R}^3$ and $w \in \mathbb{R}^2$ is

$$v \otimes w = \begin{pmatrix} v_1 w_1 & v_1 w_2 \\ v_2 w_1 & v_2 w_2 \\ v_3 w_1 & v_3 w_2 \end{pmatrix}, \tag{5.90}$$

which is an element of $\mathbb{R}^{3 \times 2}$. As a consequence, the linear function $\tilde{b}$ corresponding to b can easily be specified

$$\tilde{b} : \mathbb{R}^3 \otimes \mathbb{R}^2 \to \mathbb{R} : v \otimes w \mapsto \langle B, v \otimes w \rangle_{Frobenius}, \tag{5.91}$$

where $v \in \mathbb{R}^3$, $w \in \mathbb{R}^2$ and $\langle \cdot, \cdot \rangle_{Frobenius}$ is the Frobenius inner product.

Chapter 6

Structured Object Imputation

6.1 Introduction

Imputation terms the process of filling up missing data in a data set, for example in a data matrix of a statistical survey. In statistics imputation has been an important topic for decades. One of the reasons for the importance of imputation in statistics is that many tools of statistical data analysis, for example statistical tests, factor analysis, regression analysis and all the other machine learning methods that we have seen so far, require the data set to be complete to a certain extend.

According to Enders (2010), traditional imputation methods range from simple approaches like *Listwise Deletion*, where instances with missing attributes are omitted, *Arithmetic Mean Imputation*, where the mean of the corresponding attribute is imputed, *Hot-Deck Imputation*, where the latest observed value of the corresponding attribute is imputed, to more sophisticated approaches like *Regression Imputation*, where for every configuration of missing and observed attributes, so-called missing data patterns, a regression function is computed, and *Stochastic Regression Imputation*, which is an extension of Regression Imputation. State-of-the-art approaches go one step further and are mostly based on Maximum Likelihood and Bayesian formulations.

Machine learning methods for the imputation of discrete values based on various classifiers exist as well, for instance Rahman and Davis (2013) compare a classification based imputation approach using different classifiers with Arithmetic Mean Imputation. More recently Kidambi et al. (2013) formulated the *Missing Value Imputation* problem as a Structured Output Problem. The rows of a discrete valued data matrix with missing entries are considered as structured objects, which are composed of inter-related variables, namely the missing and observed entries of the row. In order to impute the missing values a procedure is proposed, that starting from a Mode Imputation iteratively improves the imputed values by re-predicting them using a generalization of a multi-class SVM.

The nature of the missing value problem complicates the usage of standard machine learning methods like regression, classification and structured prediction, since depending on the amount of variables a large number of missing data patterns has to be addressed and the fraction and distribution of the missing values together with the amount of interrelation between them have to be considered. Additionally, the approaches by Kidambi et al. (2013) and Rahman and Davis (2013) both suffer from their iterative nature, because the iterations are computationally expensive, i.e. every iteration involves a training and a prediction step.

For missing data problems, in which the data table obeys certain rules, collaborative filtering

techniques[1] are widely applied. An example might be the recommender system problems, where the data table describes user behavior in terms of user-item interactions. Even though in general the data table for a collaborative filtering problem does not necessarily have to connect users to items, but objects to other objects instead, we stick to the user-item terminology for better readability. The base assumption underlying collaborative filtering techniques is that users sharing their opinion on some items are likely to share their opinion on other items as well, which is basically a reformulation of the base assumption of machine learning, namely that similar inputs shall lead to similar outputs. Therefore, collaborative filtering approaches can be broken down to a similarity measure that allows to identify users with a similar opinion and a mechanism to combine the opinions found that way.

We would like to present an efficient alternative, the work of Szedmak et al. (2014a), which handles the mentioned difficulties in an elegant way while preserving a high extend of generality. The different missing data patterns occurring can be handled simultaneously by reinterpreting the data table appropriately. Large scale datasets can be handled efficiently by distributing the workload among several learners. Additionally, the missing entries of the table may be structured objects.

6.2 Background

6.2.1 Problem Statement

Data table example

x_{11}	$\emptyset$	x_{13}	$\emptyset$
x_{21}	$\emptyset$	x_{23}	x_{24}
x_{31}	$\emptyset$	$\emptyset$	x_{34}
$\vdots$	$\vdots$	$\vdots$	
$\emptyset$	x_{m2}	x_{m3}	x_{m4}.

$\emptyset$ missing items

Figure 6.1: The missing value problem.

At the end of the day all missing value problems can be broken down to the same task: Given a partially observed table, like the one in Figure 6.1, fill in the missing values.

The differences between collaborative filtering problems and statistical imputation problems origin from the procedure underlying the generation of the table. In the collaborative filtering problem the rows of the table correspond to users, the columns to items and the entries capture a relation between users and items, and in many statistical imputation problems the rows of the table correspond to instances (e.g. participants of a survey), the columns to experiments carried out on the instances (e.g. different questions of that survey) and the entries contain the results of the experiments carried out on an instance (e.g. the answer a user gave to a question). When considering the missing value problem as an extension of the supervised learning problem, in which no explicit distinction between input and output variables is made and the missing values can occur anywhere, the rows can be interpreted as instances, the columns as different object classes (e.g. image, textual description, class label) and the entries as the descriptions of the instances in terms of the different object classes.

[1]For more detailed information we refer the interested reader to Su and Khoshgoftaar (2009).

6.2.2 Intuitive Approach

	$\mathcal{X}^{(1)}$	$\mathcal{X}^{(2)}$	$\mathcal{X}^{(3)}$	$\mathcal{X}^{(4)}$
1	x_1^1	$\emptyset$	x_1^3	$\emptyset$
2	x_2^1	$\emptyset$	x_2^3	x_2^4
3	x_3^1	$\emptyset$	$\emptyset$	x_3^4
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
m	$\emptyset$	x_m^2	x_m^3	x_m^4

$$\Rightarrow \quad \begin{array}{c} f_{(1,3)\to(2)} : \mathcal{X}^{(1)} \times \mathcal{X}^{(3)} \to \mathcal{X}^{(2)} \\ f_{(1,3)\to(4)} : \mathcal{X}^{(1)} \times \mathcal{X}^{(3)} \to \mathcal{X}^{(4)} \\ f_{(1,4)\to(2)} : \mathcal{X}^{(1)} \times \mathcal{X}^{(4)} \to \mathcal{X}^{(2)} \\ f_{(1,4)\to(3)} : \mathcal{X}^{(1)} \times \mathcal{X}^{(4)} \to \mathcal{X}^{(3)} \\ \vdots \\ f_{(2,3,4)\to(1)} : \mathcal{X}^{(2)} \times \mathcal{X}^{(3)} \times \mathcal{X}^{(4)} \to \mathcal{X}^{(1)} \end{array}$$

Figure 6.2: The missing value problem can be transformed into multiple supervised learning problems by learning one function per missing data pattern.

Judging from Figure 6.1 a good idea, especially when interpreting the missing value problem as a generalization of the supervised learning problem, might be to decompose the problem of predicting the missing values into several different supervised learning tasks. This can either be achieved by learning one function for every combination of observed object classes available, like illustrated in Figure 6.2, or by learning one function for every pair of object classes. Unfortunately, this simple approach suffers from several issues: If, for instance, the number of object classes is large, a large number of functions will have to be learned. Depending on the fraction of missing elements, the training data available for each individual function can shrink dramatically with the number of functions that have to be learned. Therefore, probably a fundamentally different approach might be necessary to treat the problem appropriately.

6.2.3 Relational Learning Perspective

$\mathcal{A}\backslash\mathcal{B}$	b_1	b_2	b_3	b_4
a_1	$x_{a_1b_1}$	$\emptyset$	$x_{a_1b_3}$	$\emptyset$
a_2	$x_{a_2b_1}$	$\emptyset$	$x_{a_2b_3}$	$x_{a_2b_4}$
a_3	$x_{a_3b_1}$	$\emptyset$	$\emptyset$	$x_{a_3b_4}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
a_m	$\emptyset$	$x_{a_mb_2}$	$x_{a_mb_3}$	$x_{a_mb_4}$

$\Rightarrow$

$\mathcal{A}$	$\mathcal{B}$	$F(a,b)$
a_1	b_1	$x_{a_1b_1}$
a_1	b_3	$x_{a_1b_3}$
a_2	b_1	$x_{a_2b_1}$
a_2	b_3	$x_{a_2b_3}$
a_2	b_4	$x_{a_3b_4}$
$\vdots$	$\vdots$	$\vdots$
a_m	b_2	$x_{a_mb_2}$
a_m	b_3	$x_{a_mb_3}$
a_m	b_4	$x_{a_mb_4}$
a_1	b_2	$\emptyset$
$\vdots$	$\vdots$	$\vdots$
a_m	b_1	$\emptyset$

Figure 6.3: Reinterpretation of the table. A table can be interpreted as the observation of a relation between elements of two sets.

Note that a table can be always interpreted as the observation of a function F defined on the Cartesian product of two sets $\mathcal{A}$ and $\mathcal{B}$, representing a relation between elements of the set $\mathcal{A}$ and elements of the set $\mathcal{B}$

$$F : \mathcal{A} \times \mathcal{B} \to \mathcal{X}, \tag{6.1}$$

as done in Figure 6.3.

By reinterpreting the data table that way, the missing value problem can be formulated as a supervised learning problem. Let $\mathcal{D} \subset \mathcal{A} \times \mathcal{B}$ denote the pairs (a, b) for which x_{ab} is observed, i.e.

$$\mathcal{D} := \{(a, b) \in \mathcal{A} \times \mathcal{B} : x_{ab} \in \mathcal{X} \text{ and } x_{ab} \neq \emptyset\}, \tag{6.2}$$

where we denote missing entries in the table by $\emptyset$. Accordingly, the training set can be defined as

$$\{((a, b), x_{ab}) \in (\mathcal{A} \times \mathcal{B}) \times \mathcal{X} : (a, b) \in \mathcal{D}\} \tag{6.3}$$

and the test set as

$$\{((a, b), \emptyset) \in (\mathcal{A} \times \mathcal{B}) \times (\mathcal{X} \cup \{\emptyset\}) : (a, b) \in (\mathcal{A} \times \mathcal{B}) \backslash \mathcal{D}\}. \tag{6.4}$$

Depending on the nature of $\mathcal{X}$ any suitable learning algorithm, i.e. a learning algorithm that can learn a function between the sets $\mathcal{A} \times \mathcal{B}$ and $\mathcal{X}$, can be chosen. For example in recommender systems the elements of $\mathcal{X}$ frequently is a subset of the natural numbers representing ratings and the objective of the learning task is to learn the "user-rates-item" relation, but in general the elements of $\mathcal{X}$ can also be binary, categorical, real valued, vector valued or structured, representing a relation indicating whether there is a connection, a connection of a certain type, corresponding to a probability measure, a vector valued measure or to something even fancier, respectively. In the following we will consider a structured learning method to learn the relations, since it can capture the simpler cases as well.

Learning a n-ary Relation

Using the structured learning framework, learning a $\mathcal{X}$-valued n-ary relation between the sets $\mathcal{A}_1, \ldots, \mathcal{A}_n$

$$F : \mathcal{A}_1 \times \cdots \times \mathcal{A}_n \to \mathcal{X}, \tag{6.5}$$

can be performed by finding a compatibility function in a joint representation space $\mathcal{H}_\Psi$

$$\mathscr{C}_w : \mathcal{A}_1 \times \cdots \times \mathcal{A}_n \times \mathcal{X} \to \mathbb{R} : (a_1, \ldots, a_n, x) \mapsto \langle w, \Psi(a_1, \ldots, a_n, x) \rangle . \tag{6.6}$$

Consequently, the evaluation of the learned relation becomes

$$F(a_1, \ldots, a_n) \approx \arg\max_{x \in \mathcal{X}} \mathscr{C}_w(a_1, \ldots, a_n, x). \tag{6.7}$$

6.2.4 Feature Representation

A feature representation in the spirit of collaborative filtering methods, where objects are described purely by their associated entries of the table, typically has the form

$$\phi_{\mathcal{A}}^{(relational)} : \mathcal{A} \to \mathcal{H}_{\phi_{\mathcal{A}}^{(relational)}} : a \mapsto \tilde{\phi}\Big((x_{ab})_{\{(a,b):b \in \mathcal{B}\}}\Big). \tag{6.8}$$

Note that $\tilde{\phi} : \prod_{b \in \mathcal{B}} \mathcal{X} \mapsto \mathcal{H}_{\phi_{\mathcal{A}}^{(relational)}}$ is a mapping from a tuple, in this case representing a row, to a Hilbert space. In addition, features of the objects themselves, so-called contend-based features

$$\phi_{\mathcal{A}}^{(content-based)} : \mathcal{A} \to \mathcal{H}_{\phi_{\mathcal{A}}^{(content-based)}}, \tag{6.9}$$

can be considered. Figure 6.4 highlights the difference between content-based features, which can be thought of as intrinsic properties of the objects, and relational features, which can be thought of as interactions between object pairs.

		age,gender,...	(15,male,...)	(32,female,...)		(45,female,...)	(25,male,...)
genre,subgenre,...	$\mathcal{A}\backslash\mathcal{B}$	b_1	b_2	$\dots$		b_{n-1}	b_n
$(sports, extreme, \dots)$	a_1	$x_{a_1b_1}$	$\emptyset$	$\dots$		$x_{a_1b_{n-1}}$	$\emptyset$
$(action, comedy, \dots)$	a_2	$x_{a_2b_1}$	$\emptyset$	$\dots$		$x_{a_2b_{n-1}}$	$x_{a_2b_n}$
$(action, thriller, \dots)$	a_3	$x_{a_3b_1}$	$\emptyset$	$\dots$		$\emptyset$	$x_{a_3b_n}$
	$\vdots$	$\vdots$	$\vdots$			$\vdots$	$\vdots$
$(crime, drama, \dots)$	a_m	$\emptyset$	$x_{a_mb_2}$	$\dots$		$x_{a_mb_{n-1}}$	$x_{a_mb_n}$

Figure 6.4: Content based and relational features illustrated in the example of a movie recommendation system. The rows correspond to movies and the columns to users. Every user is characterized by content based features like age or gender and by relational features like the set of ratings made by the user. Movies are characterized analogously, for every movie there are content based features like the genre or subgenre of the movie and relational features like the set of ratings obtained by the movie.

Both feature representations can be combined, in the collaborative filtering literature methods using such a feature representation are referred to as hybrid methods. One way to combine the feature representation is to use the tensor product

$$\phi_{\mathcal{A}}^{(hybrid)} : \mathcal{A} \to \mathcal{H}_{\phi_{\mathcal{A}}^{(content-based)}} \otimes \mathcal{H}_{\phi_{\mathcal{A}}^{(relational)}}. \tag{6.10}$$

A feature representation for the objects indexing the columns can be defined analogously. Lastly, a feature space mapping

$$\psi : \mathcal{X} \to \mathcal{H}_{\psi} \tag{6.11}$$

for the entries themselves is necessary in the general case.

After specifying the feature representations the learning procedure needs to find a bilinear mapping, or, using the tensor product representation, a linear mapping from the feature representation of objects indexing the rows and the objects indexing the columns to the representation of the relation.

6.3 Relational Learning Using MMR

6.3.1 Problem Formulation

Let's assume that suitable Hilbert space mappings for the elements of the sets $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{X}$ are given by $\phi_{\mathcal{A}}$, $\phi_{\mathcal{B}}$, and ψ. Recall that using the MMR formalism learning the compatibility function

$$\mathscr{C}_w : \mathcal{A} \times \mathcal{B} \times \mathcal{X} \to \mathbb{R} : (a, b, x) \mapsto \langle w, \phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b) \otimes \psi(x) \rangle \tag{6.12}$$

is performed by finding a *linear* mapping between the two spaces $\mathcal{H}_{\phi_{\mathcal{A}}} \otimes \mathcal{H}_{\phi_{\mathcal{B}}}$ and $\mathcal{H}_{\psi}$. This linear mapping can be seen as linear approximation of the $\mathcal{X}$-valued relation

$$F : \mathcal{A} \times \mathcal{B} \to \mathcal{X}, \tag{6.13}$$

by writing the multilinear approximation of the relation

$$\tilde{F} : \mathcal{H}_{\phi_{\mathcal{A}}} \times \mathcal{H}_{\phi_{\mathcal{B}}} \to \mathcal{H}_{\psi}, \tag{6.14}$$

as the corresponding linear function in the tensor product space

$$\tilde{\tilde{F}} : \mathcal{H}_{\phi_{\mathcal{A}}} \otimes \mathcal{H}_{\phi_{\mathcal{B}}} \to \mathcal{H}_{\psi} : \phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b) \mapsto W(\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b)). \tag{6.15}$$

Remark 85. (Learning n-ary relations) *Practically the same learning scheme can be applied when learning a $\mathcal{X}$-valued n-ary relation between the sets $\mathcal{A}_1, \ldots, \mathcal{A}_n$*

$$F : \mathcal{A}_1 \times \cdots \times \mathcal{A}_n \to \mathcal{X}. \tag{6.16}$$

Given that $\phi_{\mathcal{A}_1}, \ldots, \phi_{\mathcal{A}_n}$ are the corresponding feature space mappings, the multilinear approximation of the relation becomes

$$\tilde{F} : \mathcal{H}_{\phi_{\mathcal{A}_1}} \times \cdots \times \mathcal{H}_{\phi_{\mathcal{A}_n}} \to \mathcal{H}_{\psi}, \tag{6.17}$$

for which a linear mapping from the tensor product $\mathcal{H}_{\phi_{\mathcal{A}_1}} \otimes \cdots \otimes \mathcal{H}_{\phi_{\mathcal{A}_n}}$ to $\mathcal{H}_{\psi}$

$$\tilde{\tilde{F}} : \mathcal{H}_{\phi_{\mathcal{A}_1}} \otimes \cdots \otimes \mathcal{H}_{\phi_{\mathcal{A}_n}} \to \mathcal{H}_{\psi} : \phi_{\mathcal{A}_1}(a_1) \otimes \cdots \otimes \phi_{\mathcal{A}_n}(a_n) \mapsto W(\phi_{\mathcal{A}_1}(a_1) \otimes \cdots \otimes \phi_{\mathcal{A}_n}(a_n)) \tag{6.18}$$

exists.

Therefore, after a proper problem formulation the MMR is used to find the linear mapping $\tilde{\tilde{F}}$. For binary relations the resulting optimization problem is

$$\boxed{\begin{array}{ll} \min & \|W\|^2_{Frobenius} + C \sum_{i=1}^{O} \xi_i \\ \text{w.r.t.} & W : \mathcal{H}_{\phi_{\mathcal{A}}} \otimes \mathcal{H}_{\phi_{\mathcal{B}}} \to \mathcal{H}_{\psi}, \xi \in \mathbb{R}^O, \\ \text{s.t.} & \left\langle \psi(x_{a_i b_i}), W(\phi_{\mathcal{A}}(a_i) \otimes \phi_{\mathcal{B}}(b_i)) \right\rangle_{\mathcal{H}_{\psi}} \geq 1 - \xi_i, \\ & \xi_i \geq 0, \text{ for } i \in \{1, \ldots, O\}, \end{array}} \tag{6.19}$$

where the set $\{x_{a_1 b_1}, \ldots, x_{a_O b_O}\}$ denotes all the observed entries of the table.

6.3.2 Kernel Version

In the following let $k_{\mathcal{A}}(a_i, a_j) := \langle \phi_{\mathcal{A}}(a_i), \phi_{\mathcal{A}}(a_j) \rangle$, $k_{\mathcal{B}}(b_i, b_j) := \langle \phi_{\mathcal{B}}(b_i), \phi_{\mathcal{B}}(b_j) \rangle$ and $k_{\mathcal{X}}(x_i, x_j) := \langle \psi(x_i), \psi(x_j) \rangle$ denote the kernels corresponding to the representation spaces.

Proposition 86. (Relational-MMR kernel version) *The dual problem of Optimization Problem 6.19 is given by*

$$\boxed{\begin{array}{ll} \min & \frac{1}{2} \sum_{i=1}^{O} \sum_{j=1}^{O} \alpha_i \alpha_j k_{\mathcal{A}}(a_i, a_j) k_{\mathcal{B}}(b_i, b_j) k_{\mathcal{X}}(x_i, x_j) - \mathbf{1}'\alpha \\ w.r.t. & \alpha \\ s.t. & 0 \leq \alpha_i \leq C, i \in \{1, \ldots, O\}. \end{array}} \tag{6.20}$$

Proof. The proof is identical to the proof of Proposition 80. Additionally, the Identity 5.19 was used to obtain

$$\begin{aligned} k_{\mathcal{A}\times\mathcal{B}}((a_i, b_i), (a_j, b_j)) &= \langle \phi_{\mathcal{A}}(a_i) \otimes \phi_{\mathcal{B}}(b_i), \phi_{\mathcal{A}}(a_j) \otimes \phi_{\mathcal{B}}(b_j) \rangle && \text{(by definition)} \\ &= \langle \phi_{\mathcal{A}}(a_i), \phi_{\mathcal{A}}(a_j) \rangle \langle \phi_{\mathcal{B}}(b_i), \phi_{\mathcal{B}}(b_j) \rangle && \text{(Identity 5.19)} \\ &= k_{\mathcal{A}}(a_i, a_j) k_{\mathcal{B}}(b_i, b_j) && \text{(by definition).} \end{aligned} \tag{6.21}$$

□

Proposition 87. (Relational-MMR predictor) *The pre-image problem of the relational-MMR takes the form*

$$x^*(a,b) = \arg\max_{x\in\mathcal{X}} \sum_{i=1}^{O} \alpha_i k_{\mathcal{A}}(a_i,a) k_{\mathcal{B}}(b_i,b) k_{\mathcal{X}}(x_i,x) \tag{6.22}$$

Proof. The parametrization of the learned mappings takes the form

$$W = \sum_{i=1}^{O} \alpha_i (\phi_{\mathcal{A}}(a_i) \otimes \phi_{\mathcal{B}}(b_i) \otimes \psi(x_i)), \tag{6.23}$$

yielding the prediction function

$$\begin{aligned}
x^*(a,b) &= \arg\max_{x\in\mathcal{X}} \langle \psi(x), W(\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b)) \rangle && \text{(pre-image problem)}\\
&= \arg\max_{x\in\mathcal{X}} \left\langle \psi(x), \left(\sum_{i=1}^{O} \alpha_i (\phi_{\mathcal{A}}(a_i) \otimes \phi_{\mathcal{B}}(b_i) \otimes \psi(x_i)) \right) (\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b)) \right\rangle && \text{(Equation 6.23)}\\
&= \arg\max_{x\in\mathcal{X}} \sum_{i=1}^{O} \alpha_i \langle \psi(x), (\phi_{\mathcal{A}}(a_i) \otimes \phi_{\mathcal{B}}(b_i) \otimes \psi(x_i))(\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b)) \rangle && \text{(bilinearity i.p.)}\\
&= \arg\max_{x\in\mathcal{X}} \sum_{i=1}^{O} \alpha_i \left\langle \psi(x), \psi(x_i)(\phi_{\mathcal{A}}(a_i) \otimes \phi_{\mathcal{B}}(b_i))'(\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b)) \right\rangle && \text{(Kronecker product)}\\
&= \arg\max_{x\in\mathcal{X}} \sum_{i=1}^{O} \alpha_i \langle \psi(x), \psi(x_i) \langle \phi_{\mathcal{A}}(a_i) \otimes \phi_{\mathcal{B}}(b_i), \phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b) \rangle \rangle && (u'v = \langle u, v \rangle)\\
&= \arg\max_{x\in\mathcal{X}} \sum_{i=1}^{O} \alpha_i \langle \phi_{\mathcal{A}}(a_i) \otimes \phi_{\mathcal{B}}(b_i), \phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b) \rangle \langle \psi(x), \psi(x_i) \rangle && \text{(bilinearity i.p.)}\\
&= \arg\max_{x\in\mathcal{X}} \sum_{i=1}^{O} \alpha_i k_{\mathcal{A}}(a_i,a) k_{\mathcal{B}}(b_i,b) k_{\mathcal{X}}(x_i,x) && \text{(kernel trick)}
\end{aligned} \tag{6.24}$$

evaluated on the pair (a,b). □

The view on kernel functions as similarity functions, e.g. motivated by Equation 4.64, allows an intuitive interpretation of the prediction function

$$x^*(a,b) = \arg\max_{x\in\mathcal{X}} \sum_{i=1}^{O} \alpha_i \underbrace{k_{\mathcal{A}}(a_i,a)}_{\text{row score}} \underbrace{k_{\mathcal{B}}(b_i,b)}_{\text{column score}} \underbrace{k_{\mathcal{X}}(x_i,x)}_{\text{entry score}}. \tag{6.25}$$

The optimizer of the pre-image problem x^* maximizes the weighted combination of the agreement with all observed entries of the table where the agreement with entries of similar rows and columns is emphasized due to the row- and column-kernels.

6.3.3 Solving the Optimization Problem Using Frank-Wolfe

Optimization Problem 6.20, the dual problem of the MMR, is a constrained quadratic optimization problem. Since the optimization only depends on the dual parameter vector $\alpha := (\alpha_1, \cdots, \alpha_O)'$ the objective function

$$g(\alpha) = \frac{1}{2} \sum_{i=1}^{O} \sum_{j=1}^{O} \alpha_i \alpha_j \underbrace{k_{\mathcal{A}}(a_i,a_j) k_{\mathcal{B}}(b_i,b_j) k_{\mathcal{X}}(x_i,x_j)}_{=:K_{ij}} - \mathbf{1}'\alpha \tag{6.26}$$

can be written compactly by using vector- and matrix multiplications

$$\begin{aligned} g(\alpha) &= \frac{1}{2}\sum_{i=1}^{O}\sum_{j=1}^{O}\alpha_i K_{ij}\alpha_j - \mathbf{1}'\alpha && \text{(Equation 6.26)} \\ &= \frac{1}{2}\alpha' K\alpha - \mathbf{1}'\alpha && \text{(matrix-vector multiplication).} \end{aligned} \tag{6.27}$$

Therefore, Optimization Problem 6.20 can be equivalently written as

$$\boxed{\begin{array}{ll} \min & \frac{1}{2}\alpha' K\alpha - \mathbf{1}'\alpha =: g(\alpha) \\ \text{w.r.t.} & \alpha \in \mathbb{R}^{|\mathcal{D}|} \\ \text{s.t.} & 0 \le \alpha_i \le C, i \in \{1,\dots,O\}\,. \end{array}} \tag{6.28}$$

Requirements

The objective function g is a convex differentiable function and the constraints restrict the domain of the optimization problem to an O-dimensional hypercube $[0,C]^O$, which is a convex compact subset of $\mathbb{R}^O$.

In order to recognize that g is a convex function, the fact that K is a symmetric positive semidefinite matrix and Proposition 88 are used. The symmetry and positive semidefiniteness of the matrix K follow from the symmetry and positive semidefiniteness of the joint kernel function $k((a_i,b_i,x_i),(a_j,b_j,x_j)) = k_{\mathcal{A}}(a_i,a_j)k_{\mathcal{B}}(b_i,b_j)k_{\mathcal{X}}(x_i,x_j)$ comprising its entries.

Proposition 88. (Convexity of quadratic form) *For a symmetric positive semidefinite (p.s.d.) matrix $Q \in \mathbb{R}^{n\times n}$ the function*

$$h(x) := \frac{1}{2}x'Qx + c'x \tag{6.29}$$

is convex.

Proof. The function h is convex if the inequality

$$h(tx + (1-t)y) \le th(x) + (1-t)h(y) \tag{6.30}$$

holds for all $x, y \in \mathbb{R}^n$ and for all $t \in [0,1]$. By the expansion of Definition 6.29 in Inequality 6.30, we get

$$\frac{1}{2}(tx+(1-t)y)'Q(tx+(1-t)y) + c'(tx+(1-t)y) \le t(\frac{1}{2}x'Qx + c'x) + (1-t)(\frac{1}{2}y'Qy + c'y), \tag{6.31}$$

where it is easy to observe that the linear terms cancel each other.

Therefore, consideration of the left hand side of Inequality 6.31 and estimation from above with the right hand side

$$\begin{aligned} &\frac{1}{2}(tx+(1-t)y)'Q(tx+(1-t)y) \\ &= \frac{1}{2}(y+t(x-y))'Q(y+t(x-y)) \\ &= \frac{1}{2}(y'Qy + t(x-y)'Qy + ty'Q(x-y) + t^2(x-y)'Q(x-y)) && \text{(bilinerity of i.p.)} \\ &= \frac{1}{2}y'Qy + t(x-y)'Qy + \frac{1}{2}\underbrace{t^2}_{\ge t}\underbrace{(x-y)'Q(x-y)}_{\ge 0} && \text{(symmetry of Q \& i.p.)} \\ &\ge \frac{1}{2}y'Qy + t(x-y)'Qy + \frac{1}{2}t(x-y)'Q(x-y) && \text{(p.s.d. of Q)} \\ &= \frac{1}{2}(1-t)y'Qy + \frac{1}{2}tx'Qx && \text{(symmetry of Q \& i.p.)} \end{aligned} \tag{6.32}$$

yields the convexity of h. □

Consequently, the requirements of the Frank-Wolfe algorithm, see Appendix 6.A, which is an iterative algorithm designed to solve constrained optimization problems, are satisfied and it can be applied.

Solution of the Linear Sub-problem

The only thing missing for the application of the Frank-Wolfe algorithm is the solution of the linear sub-problem

$$\boxed{\begin{array}{ll} \min & \langle K\alpha - \mathbf{1}, s\rangle = \nabla g(\alpha)' s \\ \text{w.r.t.} & s \in \mathbb{R}^{|\mathcal{D}|} \\ \text{s.t.} & 0 \le s_i \le C, i \in \{1, \dots, |\mathcal{D}|\}, \end{array}} \tag{6.33}$$

which is required in every step.

For the proof of the following proposition we need the supremum norm and the 1-norm, which is a special case of the p-norm.

Definition 89. (Supremum norm) *The supremum norm of a vector in $\mathbb{R}^d$ is defined as*

$$\|\cdot\|_\infty : \mathbb{R}^d \mapsto [0, \infty) : x \mapsto \sup_{i \in \{1,\dots,d\}} |x_i|. \tag{6.34}$$

Definition 90. (p-norm) *For $1 \le p < \infty$ the p-norm of a vector in $\mathbb{R}^d$ is defined as*

$$\|\cdot\|_p : \mathbb{R}^d \mapsto [0, \infty) : x \mapsto \left(\sum_{i=1}^{d} |x_i|^p\right)^{\frac{1}{p}}. \tag{6.35}$$

Proposition 91. (MMR sub-problem solution) *The i-th component of a minimizer s^* of Optimization Problem 6.33 is*

$$s_i^* = \frac{C}{2} - sign((\nabla g(\alpha))_i)\frac{C}{2} = \begin{cases} 0 & , \text{ for } (\nabla g(\alpha))_i > 0 \\ C & , \text{ for } (\nabla g(\alpha))_i < 0 \\ C/2 & , \text{ for } (\nabla g(\alpha))_i = 0, \end{cases} \tag{6.36}$$

in which sign denotes the signum function. Note that in the case $(\nabla g(\alpha))_i = 0$ the i-th component of s^* is not uniquely determined.

Proof. In order to prove Equation 6.33 it suffices to realize that the constrained domain

$$\mathcal{M} := \{\alpha \in \mathbb{R}^{|\mathcal{D}|} : \forall i \in \{1, \dots, |\mathcal{D}|\} : 0 \le \alpha_i \le C\} \tag{6.37}$$

shifted by the vector $\mathbf{1}\frac{-C}{2}$ is a multiple of the unit ball of the supremum norm, i.e.

$$\begin{aligned} \mathcal{M} - \mathbf{1}\frac{C}{2} &= \{\beta \in \mathbb{R}^{|\mathcal{D}|} : \beta = \alpha - \mathbf{1}\frac{C}{2}, \alpha \in \mathcal{M}\} \\ &= \{\beta \in \mathbb{R}^{|\mathcal{D}|} : \forall i \in \{1, \dots, |\mathcal{D}|\} : -\frac{C}{2} \le \beta_i \le \frac{C}{2}\} \\ &= \{\beta \in \mathbb{R}^{|\mathcal{D}|} : \|\beta\|_\infty \le \frac{C}{2}\} && \text{(Definition 6.34)} \\ &=: B^\infty_{\frac{C}{2}}(\mathbf{0}), \end{aligned} \tag{6.38}$$

and to follow the steps of the proof for optimization over the l_p-ball by Jaggi (2012).

The first equality in Derivation 6.38 implies that for every element $s \in \mathcal{M}$ there is an element $t \in B^{\infty}_{\frac{C}{2}}(\mathbf{0})$ with $s = t + \mathbf{1}\frac{C}{2}$. Therefore, the minimization of $\langle \nabla g(\alpha), s \rangle$ w.r.t. $s \in \mathcal{M}$ can be written as minimization of $\left\langle \nabla g(\alpha), t + \mathbf{1}\frac{C}{2} \right\rangle$ w.r.t. $t \in B^{\infty}_{\frac{C}{2}}(\mathbf{0})$ which doesn't depend on $\left\langle \nabla g(\alpha), \mathbf{1}\frac{C}{2} \right\rangle$. After omitting $\left\langle \nabla g(\alpha), \mathbf{1}\frac{C}{2} \right\rangle$, changing the minimization problem into a maximization problem and defining u as $-\nabla g(\alpha)$, Optimization Problem 6.33 can be equivalently rewritten as

$$\boxed{\begin{array}{ll} \max & \langle u, t \rangle \\ \text{w.r.t.} & t \in B^{\infty}_{\frac{C}{2}}(\mathbf{0}). \end{array}} \tag{6.39}$$

The estimation from above using the Hölder's inequality[2] yields

$$\begin{aligned} \langle u, t \rangle &\leq \sum_{i=1}^{|\mathcal{D}|} |u_i t_i| && \text{(triangle inequality)} \\ &\leq \|u\|_1 \|t\|_\infty && \text{(Hölder's inequality)} \\ &\leq \|u\|_1 \cdot \frac{C}{2} && \text{(for } t \in B^{\infty}_{\frac{C}{2}}(\mathbf{0})). \end{aligned} \tag{6.40}$$

The derivation

$$\begin{aligned} \|u\|_1 \cdot \frac{C}{2} &= \sum_{i=1}^{|\mathcal{D}|} |u_i| \frac{C}{2} && \text{(Definition 6.35)} \\ &= \sum_{i=1}^{|\mathcal{D}|} u_i sign(u_i) \frac{C}{2} && \text{(Definition of } |\cdot|) \end{aligned} \tag{6.41}$$

and back-substitution of u leads to the following representation for the maximizer

$$t_i^* := sign(-(\nabla g(\alpha))_i) \frac{C}{2}. \tag{6.42}$$

Consequently, $s^* \in \mathcal{M}$ is given by Equation 6.36. □

Computational Complexity

Typically, the cardinality of the observed sub-domain is large, even for sparsely filled data tables. A one thousand times one thousand table has one million entries and in practice data tables are much larger than that. For example, the Netflix Price Dataset published by Bennett et al. (2007) contains information about 480,189 rows (users) and 17,770 columns (movies) in the form of 2,817,131 (user, movie, rating) triplets.

Therefore, a constrained quadratic optimization problem, parametrized by a very large matrix needs to be solved. According to Appendix 6.A, the computational costs for achieving that the error between the k-th iterate and the optimal solution α^* is in $\mathcal{O}(\frac{1}{k})$, i.e.

$$|g(\alpha^{(k)}) - g(\alpha^*)| \in \mathcal{O}(\frac{1}{k}), \tag{6.43}$$

are k Frank-Wolfe iterations. The computational costs of one iteration are dominated by the computation of the gradient $\nabla g(\alpha^{(k)})$ and the solution of the linear sub-problem.

The gradient of the objective function g is

$$\nabla g(\alpha) = K\alpha - \mathbf{1}, \tag{6.44}$$

[2]For further details and the proof of the Hölder inequality, we refer the interested reader to Trèves (1967).

therefore, its computation corresponds roughly to a matrix-vector multiplication, which costs are in $\mathcal{O}(O^2)$. Using Proposition 91, the computational costs of the linear sub-problem are in $\mathcal{O}(O)$. As a consequence, the computational complexity of one iteration is $\mathcal{O}(O^2)$, which is too large for many practical applications.

6.4 Maximum Margin Multi Valued Mappings (MMMVM)

The MMMVM, developed by Sandor Szedmak, is a structured output method that addresses missing value problems by interpreting the data table as the observation of a potentially "structured object valued"-relation. In order to overcome the computational problems occurring, if a structured output method is naively applied to the task of learning a relation, a swarm of weakly coupled learners is introduced. In addition to the computational benefits, the utilization of a swarm of learners instead of one single learner allows for the exploitation of local trends.

The underlying idea stems from manifolds, where the potentially complex structure of a manifold can be described by an atlas consisting of several charts, i.e. mappings from subsets of the manifold to a hyperplane. For a smooth manifold those charts agree on overlaps. When learning a relation from a partially observed table we are in a similar situation. Assuming the data lives on a manifold, the learning task could be solved by finding the inverse mapping for every mapping in the atlas. In practice, we don't have an atlas, which is why this approach cannot be applied directly. However, if the learners are simply assigned to subsets of the domain, for example to the rows or the columns, and forced to agree on overlapping areas of their domains we will be in a similar situation.

Ghazanfar et al. (2011) and Ghazanfar et al. (2012) build a recommender system based on the MMMVM. Szedmak et al. (2014b) utilize the MMMVM to learn affordances of actions applied to pairs of objects, for instance the outcome of a stack-action applied to pairs of objects is learned. Krivić et al. (2015) demonstrate the correspondence between relations and graphs by using the MMMVM to learn missing edges in partially-known graphs.

6.4.1 Notation

For mathematical convenience let $\mathcal{D}_{a-}$ and $\mathcal{D}_{-b}$ denote the set of observed columns in the row indexed by $a \in \mathcal{A}$ and the set of observed rows in the column indexed by $b \in \mathcal{B}$ respectively,

$$\mathcal{D}_{a-} := \{b \in \mathcal{B} : (a,b) \in \mathcal{D}\} \tag{6.45}$$

and

$$\mathcal{D}_{-b} := \{a \in \mathcal{A} : (a,b) \in \mathcal{D}\}. \tag{6.46}$$

Note that $\mathcal{D}$ can be written as

$$\begin{aligned} \mathcal{D} &= \{(a,b) \in \mathcal{A} \times \mathcal{B} : a \in \mathcal{A} \text{ and } b \in \mathcal{D}_{a-}\} \\ &= \{(a,b) \in \mathcal{A} \times \mathcal{B} : b \in \mathcal{B} \text{ and } a \in \mathcal{D}_{-b}\}. \end{aligned} \tag{6.47}$$

Additionally for a vector v in $\mathbb{R}^{|\mathcal{D}|}$, let

$$v_a := (v_{ab})'_{b \in \mathcal{D}_{a-}} \tag{6.48}$$

denote the part of the vector

$$v := (v_a)'_{a \in \mathcal{A}} \tag{6.49}$$

corresponding to $\mathcal{D}_{a-}$ and

$$v_b := (v_{ab})'_{a \in \mathcal{D}_{b-}} \tag{6.50}$$

denote the part of v corresponding to $\mathcal{D}_{b-}$.

6.4.2 Defining the Swarm of Learners

One way to overcome the difficulties that arise, when applying MMR directly to the relational dataset, is to distribute the available data among various cooperating learners. A properly configured swarm of learners can capture complicated data manifolds, while preserving computational efficiency.

For example we could attach one learner to each row, i.e.

$$\tilde{F}_a : \mathcal{H}_{\phi_\mathcal{B}} \to \mathcal{H}_\psi : \phi_\mathcal{B}(b) \mapsto W_a(\phi_\mathcal{A}(a) \otimes \phi_\mathcal{B}(b)), \forall a \in \mathcal{A} : \exists b \in \mathcal{B} \ s.t. \ (a,b) \in \mathcal{D}. \tag{6.51}$$

The learner $\tilde{F}_a$ takes the feature representation of a column $\phi_\mathcal{B}(b) \in \mathcal{H}_{\phi_\mathcal{B}}$ and maps it to an element in the label space $W_a(\phi_\mathcal{A}(a) \otimes \phi_\mathcal{B}(b)) \in \mathcal{H}_\psi$ corresponding to the entry x_{ab} in the table. Without further modifications the resulting swarm of learners would be independent, meaning that the prediction $\tilde{F}_a(\phi_\mathcal{B}(b))$ for entry x_{ab} and the prediction $\tilde{F}_{a'}(\phi_\mathcal{B}(b))$ for entry $x_{a'b}$ would not affect each other. Although computational efficiency is gained that way an important source of information is lost.

For many practical applications the predictions $\tilde{F}_a(\phi_\mathcal{B}(b))$ and $\tilde{F}_{a'}(\phi_\mathcal{B}(b))$ should influence each other. Just think of the Netflix Price dataset mentioned earlier. Clearly, in many cases the rating of movie b by user a tells us about something about the rating of movie b by user a'. If movie b is a brilliant movie it will be likely that both ratings x_{ab} and $x_{a'b}$ are high. Similarly, if movie b is bad it will be likely that both ratings are low. In other words, it is fair to assume that ratings of the same movie are consistent with each other to a certain degree.

In order to incorporate this kind of interaction between the learners an appropriate coupling rule has to be applied. Szedmak et al. (2015) couple the learners by forcing them to share the same slack. Following that line, one slack variable for every column with observed entries, instead of having one slack variable for every observed entry in the table, is used. The resulting optimization problem is

$$\boxed{\begin{array}{ll} \min & \sum_{a\in\mathcal{A}} \|W_a\|^2_{Frobenius} + C \sum_{b\in\mathcal{B}} \xi_b \\ \text{w.r.t.} & W_a : \mathcal{H}_{\phi_\mathcal{A}} \otimes \mathcal{H}_{\phi_\mathcal{B}} \to \mathcal{H}_\psi, \xi \in \mathbb{R}^{|\mathcal{B}|}, \\ \text{s.t.} & \left\langle \psi(x_{ab}), W_a(\phi_\mathcal{A}(a) \otimes \phi_\mathcal{B}(b)) \right\rangle_{\mathcal{H}_\psi} \geq 1 - \xi_b, \text{ for } a \in \mathcal{A} \text{ and } b \in \mathcal{D}_{a-}, \\ & \xi_b \geq 0, \text{ for } b \in \mathcal{B}. \end{array}} \tag{6.52}$$

This way *the predictions of the learners cannot vary independently* for shared columns, since for a fixed column $b \in \mathcal{B}$ the optimization problem requires the slack ξ_b attached to it to be minimized and that the inequalities

$$\begin{aligned} &\forall a \in \mathcal{A} : \left\langle \psi(x_{ab}), W_a(\phi_\mathcal{A}(a) \otimes \phi_\mathcal{B}(b)) \right\rangle_{\mathcal{H}_\psi} \geq 1 - \xi_b \\ \Leftrightarrow\ &\forall a \in \mathcal{A} : \xi_b \geq \underbrace{1 - \left\langle \psi(x_{ab}), W_a(\phi_\mathcal{A}(a) \otimes \phi_\mathcal{B}(b)) \right\rangle_{\mathcal{H}_\psi}}_{\text{error by learner } \tilde{F}_a} \end{aligned} \tag{6.53}$$

are satisfied. Therefore, in order to minimize ξ_b the errors have to be uniformly minimized.

Remark 92. (Utilizing local trends) *Due to the fact, that learners are assigned to rows and constraints to the columns it an be useful to reorganize the table before applying the MMMVM. For example, in an image reconstruction task one could like to utilize the local smoothness of pixel values. In order to do so one learner could be assigned to every image patch and one constraint to every intensity interval.*

In general, two separating partitions of $\mathcal{A} \times \mathcal{B}$ are required, i.e. two collections of subsets of $\mathcal{A} \times \mathcal{B}$ $\mathscr{R}$ and $\mathscr{C}$ satisfying

1. $\bigcup_{\mathcal{R}\in\mathscr{R}} \mathcal{R} = \mathcal{A}\times\mathcal{B}$,
2. $\bigcup_{\mathcal{C}\in\mathscr{C}} \mathcal{C} = \mathcal{A}\times\mathcal{B}$,
3. $\forall \mathcal{R}\in\mathscr{R}, \mathcal{C}\in\mathscr{C} : \mathcal{R}\cap\mathcal{C}$ *contains at most one element and*
4. $\forall (a,b)\in\mathcal{A}\times\mathcal{B} : \exists!\ (\mathcal{R},\mathcal{C})\in\mathscr{R}\times\mathscr{C}$ *with* $\mathcal{R}\cap\mathcal{C} = (a,b)$.

Utilizing those two separating partitions a new table with one row per subset $\mathcal{R}\in\mathscr{R}$ and one column per subset $\mathcal{C}\in\mathscr{C}$ can be obtained. The entry in row $\mathcal{R}\in\mathscr{R}$ and column $\mathcal{C}\in\mathscr{C}$ for which the equality $\mathcal{R}\cap\mathcal{C} = \{(a,b)\}$ holds is x_{ab}.

6.4.3 Kernel Version

In the following let $k_{\mathcal{A}}(a_i,a_j) := \langle\phi_{\mathcal{A}}(a_i),\phi_{\mathcal{A}}(a_j)\rangle$, $k_{\mathcal{B}}(b_i,b_j) := \langle\phi_{\mathcal{B}}(b_i),\phi_{\mathcal{B}}(b_j)\rangle$ and $k_{\mathcal{X}}(x_i,x_j) := \langle\psi(x_i),\psi(x_j)\rangle$ denote the kernels corresponding to the representation spaces.

Proposition 93. (MMMVM kernel version) *The dual problem of Optimization Problem 6.52 is given by*

$$\boxed{\begin{array}{ll} \min & \frac{1}{2}\sum_{a\in\mathcal{A}}\sum_{b,b'\in\mathcal{D}_{a-}} \alpha_{ab}\alpha_{ab'}k_{\mathcal{A}}(a,a)k_{\mathcal{B}}(b,b')k_{\mathcal{X}}(x_{ab},x_{ab'}) - \sum_{(a,b)\in\mathcal{D}}\alpha_{ab} \\ w.r.t. & \alpha_{ab}\in\mathbb{R} \text{ for } (a,b)\in\mathcal{D} \\ s.t. & \sum_{a\in\mathcal{D}_{-b}}\alpha_{ab}\le C,\ \text{for } b\in\mathcal{B} \\ & \alpha_{ab}\ge 0,\ \text{for } (a,b)\in\mathcal{D}. \end{array}} \tag{6.54}$$

Proof. In order to obtain the kernel version the Lagrange dual function needs to be considered. The Lagrange dual function g is defined as the infimum of the Lagrangian with respect to the parameters of the objective function and gives us a lower bound for the objective function f. Let's refer to the objective function as f

$$f(W,\xi) := \frac{1}{2}\sum_{a\in\mathcal{A}}\|W_a\|^2_{Frobenius} + C\sum_{b\in\mathcal{B}}\xi_b, \tag{6.55}$$

where W denotes the tuple $(W_a)_{a\in\mathcal{A}}$. The corresponding Lagrangian function can be obtained by converting the greater-than-equal constraints into lower-than-equal constraints, by multiplying with minus one and using the definition of the Lagrangian function (Definition 2.69), resulting in

$$L(W,\xi,\alpha,\beta) := f(W,\xi) - \sum_{(a,b)\in\mathcal{D}}\alpha_{ab}\left(\langle\psi(x_{ab}),W_a(\phi_{\mathcal{A}}(a)\otimes\phi_{\mathcal{B}}(b)\rangle - 1 + \xi_b\right) - \sum_{b\in\mathcal{B}}\beta_b\xi_b. \tag{6.56}$$

The Lagrange dual function

$$g(\alpha,\beta) := \inf_{W,\xi} L(W,\xi,\alpha,\beta) \tag{6.57}$$

provides a lower bound on the objective function $g(\alpha,\beta)\le f(W,\xi)$. The infimum can be found by setting the derivatives with respect to W and ξ to zero:

$$\frac{\partial L}{\partial W_a} \overset{!}{=} 0, \forall a\in\mathcal{A} \tag{6.58}$$

$$\frac{\partial L}{\partial \xi} \overset{!}{=} 0 \tag{6.59}$$

When using the Frobenius norm $\|W_a\|^2_{Frobenius} = tr(W_a' W_a)$, the derivative with respect to W_a becomes

$$\frac{\partial L}{\partial W_a} = W_a - \sum_{(a,b)\in\mathcal{D}} \alpha_{ab}\psi(x_{ab})(\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b))', \tag{6.60}$$

because of the already known derivation rules

$$\frac{\partial tr(W_a' W_a)}{\partial W_a} = 2W_a \tag{6.61}$$

and

$$\frac{\partial \langle \psi(x_{ab}), W_a(\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b)) \rangle}{\partial W_a} = \psi(x_{ab})(\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b))'. \tag{6.62}$$

Therefore, W_a can be expressed in terms of α

$$W_a = \sum_{(a,b)\in\mathcal{D}} \alpha_{ab}\psi(x_{ab})(\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b))'. \tag{6.63}$$

The derivative with respect to ξ_b is

$$\frac{\partial L}{\partial \xi_b} = C - \sum_{a\in\mathcal{D}_{-b}} \alpha_{ab} - \beta_b. \tag{6.64}$$

By rearranging the terms and using the fact that $\beta_b \geq 0$, the Lagrange function is minimized with respect to ξ_b when the inequality

$$\sum_{a\in\mathcal{D}_{-b}} \alpha_{ab} = C - \underbrace{\beta_b}_{\geq 0} \leq C \tag{6.65}$$

is satisfied. The Lagrange dual function now can be written explicitly by substituting Equation 6.63 and Equation 6.65 into Equation 6.56. After simplification we obtain

$$g(\alpha) = -\frac{1}{2} \sum_{a\in\mathcal{A}} \sum_{b,b'\in\mathcal{D}_{a-}} \alpha_{ab}\alpha_{ab'} \left\langle \phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b), \phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b') \right\rangle \langle \psi(x_{ab}), \psi(x_{ab'}) \rangle + \sum_{(a,b)\in\mathcal{D}} \alpha_{ab}, \tag{6.66}$$

where the parameter β does not occur any longer. Since the dual is a lower bound on the objective function f it has to be maximized to find the best α, which is equivalent to minimizing $(-1)g(\alpha)$, resulting in the Optimization Problem 6.54. □

Proposition 94. (MMMVM Predictor) *The MMMVM prediction function takes the form*

$$x^*(a,b) = \arg\max_{x\in\mathcal{X}} \sum_{b'\in\mathcal{D}_{a-}} \alpha_{ab'} k_{\mathcal{B}}(b',b) k_{\mathcal{X}}(x_{ab'}, x). \tag{6.67}$$

Proof. Expression 6.67 is obtained by substituting W_a with Expression 6.63 in

$$x^*(a,b) = \arg\max_{x\in\mathcal{X}} \langle \psi(x), W_a(\phi_{\mathcal{A}}(a) \otimes \phi_{\mathcal{B}}(b)) \rangle. \tag{6.68}$$

After performing analogous simplifications as in Derivation 6.24 the identity

$$x^*(a,b) = \arg\max_{x\in\mathcal{X}} \sum_{b'\in\mathcal{D}_{a-}} \alpha_{ab'} k_{\mathcal{A}}(a,a) k_{\mathcal{B}}(b',b) k_{\mathcal{X}}(x_{ab'}, x) \tag{6.69}$$

is obtained. Omitting $k_{\mathcal{A}}(a,a)$, which is a constant term with respect to x and thereby does not influence the maximization, results in Expression 6.67. □

Again the prediction function admits an intuitive interpretation

$$x^*(a,b) = \arg\max_{x\in\mathcal{X}} \sum_{b'\in\mathcal{D}_{a-}} \alpha_{ab'} \underbrace{k_{\mathcal{B}}(b',b)}_{column\ score} \underbrace{k_{\mathcal{X}}(x_{ab'},x)}_{entry\ score}. \tag{6.70}$$

The optimizer of the pre-image problem x^* maximizes the weighted combination of the agreement with all observed entries in the same row, where the agreement with entries of similar columns is emphasized due to the column-kernels.

Additionally, the fact that the prediction of an element in row a only depends on the part of α corresponding to the learner assigned to that row allows every row to follow an individual trend. This property is particularly appealing, when applying the MMMVM on a collaborative filtering type of problem, where the rows could correspond to users for instance.

6.4.4 Solving the Optimization Problem Using Frank-Wolfe

Let's have a look at Optimization Problem 6.54, the dual problem of the MMMVM, and compare it to Optimization Problem 6.20. First of all, we simplify the expression consisting of the nested sums in the objective function of Optimization Problem 6.54

$$\begin{aligned}
&\sum_{a\in\mathcal{A}} \sum_{b,b'\in\mathcal{D}_{a-}} \alpha_{ab}\alpha_{ab'} k_{\mathcal{A}}(a,a) k_{\mathcal{B}}(b,b') k_{\mathcal{X}}(x_{ab},x_{ab'}) \\
&= \sum_{a\in\mathcal{A}} k_{\mathcal{A}}(a,a) \sum_{b,b'\in\mathcal{D}_{a-}} \alpha_{ab}\alpha_{ab'} \underbrace{k_{\mathcal{B}}(b,b') k_{\mathcal{X}}(x_{ab},x_{ab'})}_{=:K^a_{bb'}} && \text{(pull out } k_{\mathcal{A}}(a,a)) \\
&= \sum_{a\in\mathcal{A}} k_{\mathcal{A}}(a,a) \sum_{b,b'\in\mathcal{D}_{a-}} \alpha_{ab} K^a_{bb'} \alpha_{ab'} && \text{(substitution)} \\
&= \sum_{a\in\mathcal{A}} k_{\mathcal{A}}(a,a)\ \alpha_a' K^a \alpha_a && \text{(matrix-vector mult.)},
\end{aligned} \tag{6.71}$$

where $K^a \in \mathbb{R}^{|\mathcal{D}_{a-}|\times|\mathcal{D}_{a-}|}$ captures all the kernel evaluations associated to learner W_a and α_a denotes the corresponding vector of dual parameters $(\alpha_{ab})'_{b\in\mathcal{D}_{a-}}$. Note that $k_{\mathcal{A}}(a,a)$ only acts as a scaling factor, for mathematical convenience we assume normalized feature vectors in the following, i.e.

$$\forall a \in \mathcal{A} : 1 \overset{!}{=} \|\phi_{\mathcal{A}}(a)\|_2^2 \quad (= k_{\mathcal{A}}(a,a)). \tag{6.72}$$

After using some linear algebra, the nested sums can be written as

$$\begin{aligned}
&\sum_{a\in\mathcal{A}} k_{\mathcal{A}}(a,a)\ \alpha_a' K^a \alpha_a \\
&= \sum_{a\in\mathcal{A}} \alpha_a' K^a \alpha_a && (\forall a \in \mathcal{A} : k_{\mathcal{A}}(a,a) = 1) \\
&= \alpha' K \alpha,
\end{aligned} \tag{6.73}$$

where $\alpha \in \mathbb{R}^{|\mathcal{D}|}$ is defined as the concatenation of all α_a's

$$\alpha := (\alpha_a)'_{a\in\mathcal{A}} \tag{6.74}$$

and $K \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{D}|}$ is defined as a block-diagonal matrix containing the matrices K^a, $for\ a \in \mathcal{A}$, in the diagonal

$$K := \begin{pmatrix} K^{a_1} & 0 & 0 & \cdots & 0 \\ 0 & K^{a_2} & 0 & \cdots & 0 \\ 0 & 0 & \ddots & & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & K^{a_{|\mathcal{A}|}} \end{pmatrix}. \tag{6.75}$$

Note that in Equation 6.75 we wrote $\mathcal{A}$ as $\{a_1, \ldots, a_{|\mathcal{A}|}\}$ and additionally, the zeros are matrices of proper dimension filled with zeros, for instance the first zero-matrix in the first row is of dimension $|\mathcal{D}_{a_1-}| \times |\mathcal{D}_{a_2-}|$.

Therefore, Optimization Problem 6.54 can be equivalently written as

$$\boxed{\begin{array}{ll} \min & \frac{1}{2}\alpha' K\alpha - \mathbf{1}'\alpha =: g(\alpha) \\ \text{w.r.t.} & \alpha \in \mathbb{R}^{|\mathcal{D}|} \\ \text{s.t.} & \sum_{a\in\mathcal{D}_{-b}} \alpha_{ab} \leq C, \text{ for } b \in \mathcal{B} \\ & \alpha_{ab} \geq 0, \text{ for } (a,b) \in \mathcal{D}. \end{array}} \tag{6.76}$$

Requirements

The objective function g is of the same type as the one in Optimization Problem 6.28. Thus, it is a convex differentiable function and according to the following proposition the constraints restrict the domain to a compact convex set.

Proposition 95. (Compactness and convexity of the domain) *Let $\mathcal{M}$ denote the restricted domain*

$$\mathcal{M} := \{\alpha \in \mathbb{R}^{|\mathcal{D}|} : \alpha_{ab} \geq 0, \text{ for } (a,b) \in \mathcal{D} \text{ and } \sum_{a\in\mathcal{D}_{-b}} \alpha_{ab} \leq C, \text{ for } b \in \mathcal{B}\}. \tag{6.77}$$

$\mathcal{M}$ is a compact convex subset of $\mathbb{R}^{|D|}$.

Proof. The *compactness* of $\mathcal{M}$ follows from the relation

$$\mathcal{M} \subset [0, C]^{|\mathcal{D}|}, \tag{6.78}$$

because $[0, C]^{|\mathcal{D}|}$ is compact and subsets of compact sets are compact.

A short derivation shows the *convexity* of the domain. $\mathcal{M}$ is convex if the property

$$\forall \alpha, \tilde{\alpha} \in \mathcal{M} : \forall t \in [0,1] : (1-t)\alpha + t\tilde{\alpha} \in \mathcal{M} \tag{6.79}$$

is satisfied.

Consider arbitrary $\alpha, \tilde{\alpha} \in \mathcal{M}$. Obviously, the property

$$\forall t \in [0,1] : ((1-t)\alpha + t\tilde{\alpha})_{ab} \geq 0 \tag{6.80}$$

holds, since the inequalities $t \geq 0, (1-t) \geq 0, \alpha_{ab} \geq 0$ and $\tilde{\alpha}_{ab} \geq 0$ hold. Additionally, for all $t \in [0,1]$ and for all $b \in \mathcal{B}$ the inequality

$$\sum_{a\in\mathcal{D}_{-b}} ((1-t)\alpha + t\tilde{\alpha})_{ab} \leq C \tag{6.81}$$

holds, because of the simple derivation

$$\sum_{a\in\mathcal{D}_{-b}} ((1-t)\alpha + t\tilde{\alpha})_{ab} = (1-t) \sum_{a\in\mathcal{D}_{-b}} \alpha_{ab} + t \sum_{a\in\mathcal{D}_{-b}} \tilde{\alpha}_{ab} \quad \leq (1-t)C + tC = C. \tag{6.82}$$

□

Consequently, the requirements of the Frank-Wolfe algorithm are fulfilled and it can be applied.

Solution of the Linear Sub-problem

Step 1 of the Frank-Wolfe Algorithm 1 requires the solution of the linear sub-problem

$$\boxed{\begin{array}{ll} \min & \langle K\alpha - \mathbf{1}, s\rangle = \nabla g(\alpha)'s \\ \text{w.r.t.} & s \in \mathbb{R}^{|\mathcal{D}|} \\ \text{s.t.} & \sum_{a\in\mathcal{D}_{-b}} s_{ab} \leq C, \text{ for } b \in \mathcal{B} \\ & s_{ab} \geq 0, \text{ for } (a,b) \in \mathcal{D}. \end{array}} \tag{6.83}$$

Proposition 96. (MMMVM sub-problem solution) *Szedmak et al. (2015) show that the minimizer s^* of Optimization problem 6.83 can be obtained by setting*

$$(s_b^*)_i = \begin{cases} C & \text{if } i = \arg\min_{k\in\{1,\dots,|\mathcal{D}_{-b}|\}} (\nabla g(\alpha))_k \text{ and } (\nabla g(\alpha))_k < 0, \\ 0 & \text{otherwise} \end{cases} \tag{6.84}$$

for all $i \in \{1,\dots,|\mathcal{D}_{-b}|\}$ and for all b in $\mathcal{B}$.

Proof. Thanks to the linearity of $\langle \nabla g(\alpha), \cdot\rangle$ and the nature of its constraints, Optimization Problem 6.83 can be split up into independent sub-problems.

Rewriting the objective function $\nabla g(\alpha)'s$ utilizing Notation 6.50 yields

$$\begin{aligned} \nabla g(\alpha)'s &= \sum_{(a,b)\in\mathcal{D}} (\nabla g(\alpha))_{ab} s_{ab} && \text{(per definition)} \\ &= \sum_{b\in\mathcal{B}} \sum_{a\in\mathcal{D}_{-b}} (\nabla g(\alpha))_{ab} s_{ab} && \text{(Equation 6.47)} \\ &= \sum_{b\in\mathcal{B}} \langle (\nabla g(\alpha))_b, s_b\rangle && \text{(definition i.p.).} \end{aligned} \tag{6.85}$$

Putting the result of Derivation 6.85 back into Optimization Problem 6.83 leads to the optimization problem

$$\boxed{\begin{array}{ll} \min & \sum_{b\in\mathcal{B}} \langle (\nabla g(\alpha))_b, s_b\rangle \\ \text{w.r.t.} & s \in \mathbb{R}^{|\mathcal{D}|} \\ \text{s.t.} & \langle \mathbf{1}, s_b\rangle \leq C, \text{ for } b \in \mathcal{B} \\ & s_{ab} \geq 0, \text{ for } (a,b) \in \mathcal{D}, \end{array}} \tag{6.86}$$

in which every summand can be considered individually as there are no dependencies between them. Therefore, solving Optimization Problem 6.83 is equivalent to solving the optimization problem

$$\boxed{\begin{array}{ll} \min & \langle (\nabla g(\alpha))_b, s_b\rangle \\ \text{w.r.t.} & s_b \in \mathbb{R}^{|\mathcal{D}_{-b}|} \\ \text{s.t.} & \langle \mathbf{1}, s_b\rangle \leq C, \\ & s_{ab} \geq 0, \text{ for } a \in \mathcal{D}_{-b} \end{array}} \tag{6.87}$$

for all b in $\mathcal{B}$ independently. According to Szedmak et al. (2015) the solution of Optimization Problem 6.87 is given by Equation 6.84, which seems intuitive as it emphasizes the most negative component of the part of the gradient corresponding to the constraint b as much as possible. □

Computational Complexity

In contrast to Optimization Problem 6.28, in which the complexity of Frank-Wolfe iteration was $\mathcal{O}(|\mathcal{D}|^2)$, the matrix parameterizing Optimization Problem 6.76 is a block-diagonal matrix. The

combination of the block-diagonal structure of K with the sparsity of the optimal solution of the linear sub-problem allows for an efficient computation of the gradient.

Szedmak et al. (2015) show that the complexity of the gradient computation can be reduced to linear complexity, i.e. to $\mathcal{O}(|\mathcal{D}|)$ operations, by using the update rule

$$(\nabla g(\alpha^{(k)}))_a := \gamma K^a s_a^* + (1-\gamma)(\nabla g(\alpha^{(k-1)}))_a, \tag{6.88}$$

and that s^* can be computed in linear time. The estimate for the complexity of the gradient computation is obtained by bounding the size of the K^a blocks in a clever way.

Therefore, the worst-case computational complexity of one Frank-Wolfe iteration is $\mathcal{O}(|\mathcal{D}|)$, which is a significant improvement compared to the $\mathcal{O}(|\mathcal{D}|^2)$ complexity of the relational MMR.

6.5 Application Example - Missing Edges in Multiplex Networks

> "A multiplex network is a graph defined over a set of nodes linked by different types of relations." (Pujari and Kanawati, 2015)

In this section a sub-problem of a robot planning task, in which a robot needs to plan its actions to achieve a specific goal, is considered. Such a goal could be to build a tower of maximum height composed of a subset of objects in the environment. In order to plan its actions, the robot needs information about its environment, in particular how it can interact with pairs of objects. These possible interactions between the robot and the pairs of objects are called affordances. An example of an affordance for the "stack action" of a robot could be: "$object_i$ can be stacked on $object_j$". Therefore, the objective of the robot's machine learning module is to infer the missing affordances of a partially known affordance table.

6.5.1 Details about the Dataset

The dataset contains affordances for all pairs of a set of 82 objects. For every object pair there are four different types of affordances. One of these could correspond to the "stack-ability of objects" and could have the manifestations: "$object_1$ can be placed on $object_2$","$object_2$ can be placed on $object_1$" and "$object_1$ and $object_2$ cannot be stacked"[3]. Three of the affordance types have three different manifestations and the remaining one has four different manifestations. As a result, the output space of interest contains $3^3 \cdot 4^1 = 108$ different affordance vectors.

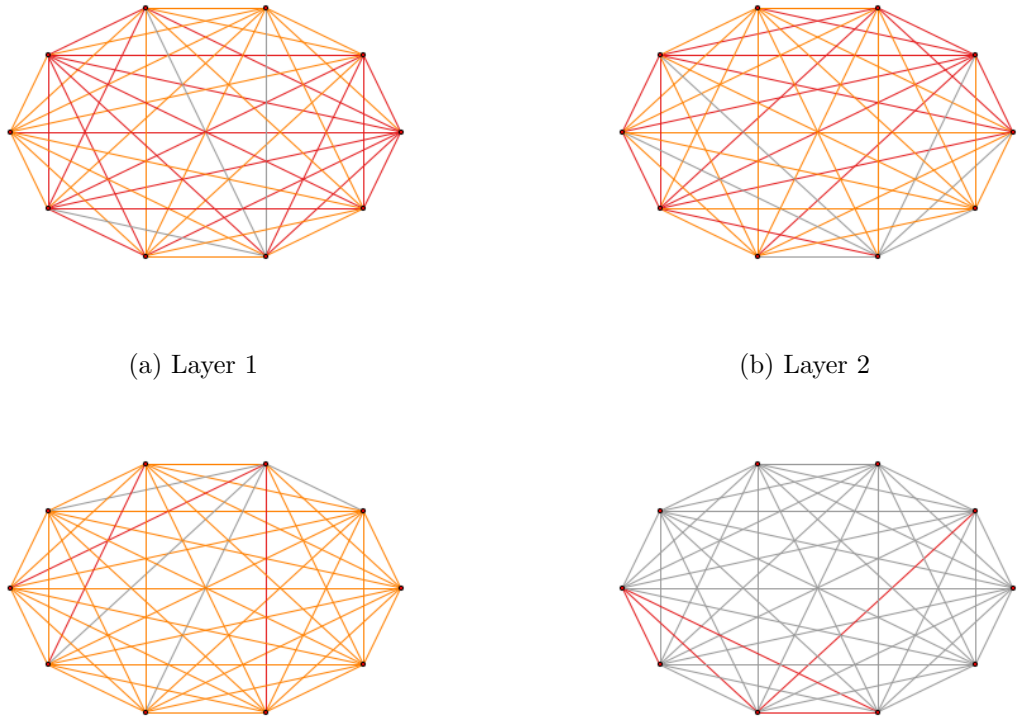
The described dataset can be interpreted as a multiplex network, in which nodes correspond to objects and edges to affordances between them. Figure 6.1 shows a layer-wise depiction of a sub-network composed by ten objects.

6.5.2 Application of the MMR & MMMVM

In order to address the learning problem with the relational MMR or the MMMVM, the multiplex network is represented as a vector valued table. The entries of the table are interpreted as structured objects. In this example the sets indexing the rows $\mathcal{A}$ and the columns $\mathcal{B}$ are identical, namely, $\mathcal{A} = \mathcal{B} = \{Object_1, \ldots, Object_{82}\}$ and the entries of the table are elements of $\mathcal{X}$ which is a subset of $\mathbb{R}^4$.

The only requirement for the application of the relational MMR and MMMVM is the choice of kernel functions and the penalty parameter C. For the sake of simplicity normalized polynomial

[3] Unfortunately, this information is not available for us.



(a) Layer 1

(b) Layer 2

(c) Layer 3

(d) Layer 4

Figure 6.1: A layer-wise depiction of a subset of the multiplex network. The red circles correspond to objects and colored edges to different interaction types.

$$k_{poly}(x,y) = \frac{(\langle x,y\rangle + b)^p}{\sqrt{(\langle x,x\rangle + b)^p \cdot (\langle y,y\rangle + b)^p}} \tag{6.89}$$

and radial basis function kernels

$$k_{rbf}(x,y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}}, \tag{6.90}$$

in which $\langle \cdot, \cdot \rangle$ is the scalar product of $\mathbb{R}^d$ and $\| \cdot \|$ the corresponding norm, are considered.

As a consequence, the kernel functions

$$k_{\mathcal{X}}(x_1, x_2) := k_t(x_1, x_2), \tag{6.91}$$

$$k_{\mathcal{A}}(a_1, a_2) := k_t(r_{a_1}, r_{a_2}) \tag{6.92}$$

and

$$k_{\mathcal{B}}(b_1, b_2) := k_t(c_{b_1}, c_{b_2}), \tag{6.93}$$

in which $r_a := (x_{ab})_{b \in \mathcal{D}_{a-}}$ is the concatenation of all vectors in row a, $c_b := (x_{ab})_{a \in \mathcal{D}_{-b}}$ is the concatenation of all vectors in column b and t is a placeholder for $poly$ or rbf, are obtained. The remaining parameters b, p, σ and C can be found by cross-validation or simply by trial and error.

6.5.3 Experimental Setup and Results

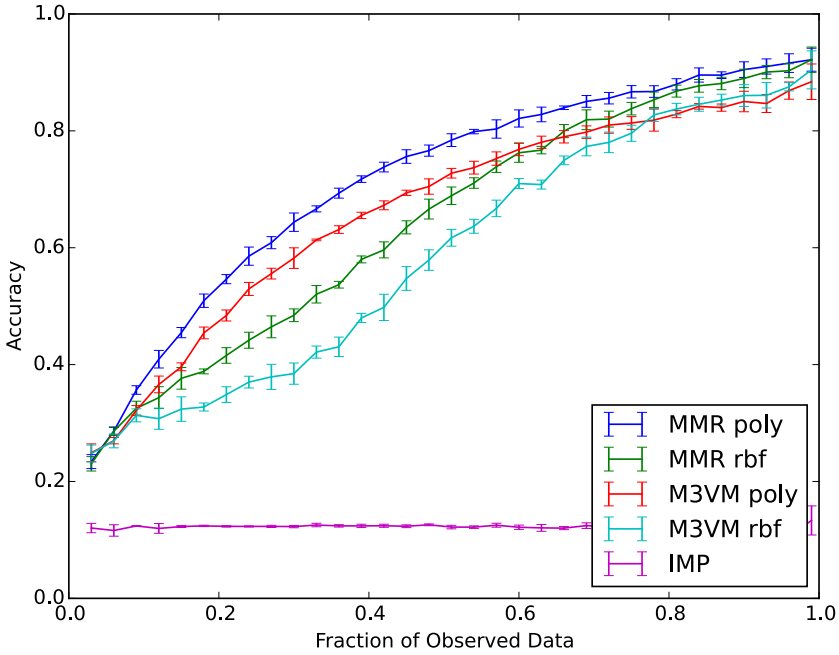
The relational MMR and the MMMVM were applied on different fractions of the table in order to predict the missing entries. Fractions ranging from three percent to 99 percent of the entries were considered. Each random split into observed and missing entries was repeated five times.

Figure 6.2 depicts the comparison of the relational MMR and the MMMVM using polynomial and radial basis function kernels. For the polynomial row and column kernels the parameters $b = 1$ and $p = 15$ and for the polynomial entry kernels the parameters $b = 1$ and $p = 4$ were used. For the radial basis function row, column and entry kernels the parameter $\sigma = 1$ was used. Additionally, as a baseline method a mode imputation was used.

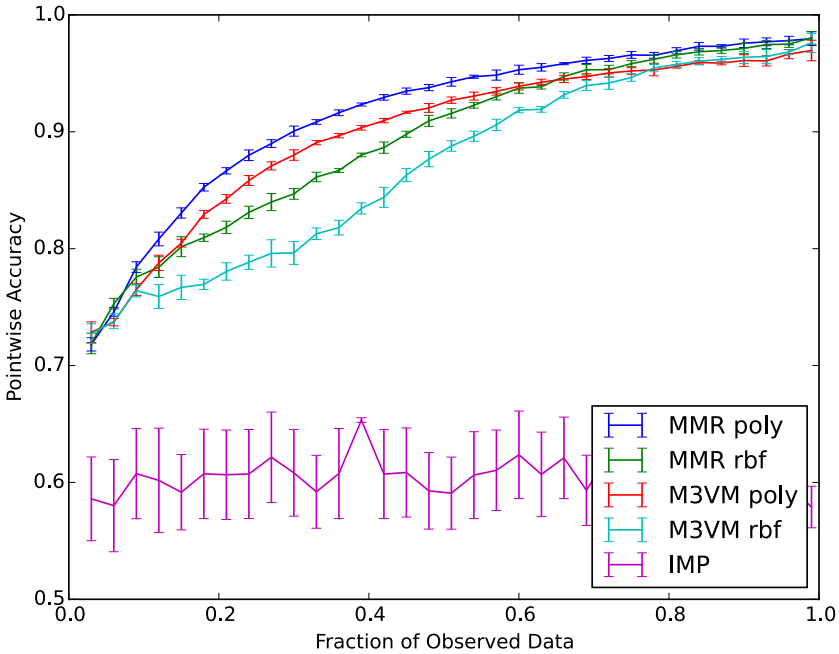
Two performance measures were considered:

- *Accuracy* - the fraction of correctly predicted affordance vectors.
- *Pointwise Accuracy* - the fraction of correctly predicted affordance vector entries.

It is to be noted, that in this comparison all four methods behave similarly. This observation becomes particularly clear considering the pointwise accuracy plot. Nevertheless, for the parameters used the polynomial kernels worked slightly better than the radial basis function kernels. Equally, the relational MMR behaved slightly better than the MMMVM. Using the polynomial kernels for both the relational MMR and the MMMVM already twenty percent of the edges suffice to recover more than eighty percent of the missing missing edges in the multiplex network.



(a) Accuracy



(b) Pointwise Accuracy

Figure 6.2: Relational MMR and MMMVM were evaluated on different fractions of observed data using various kernels. The *blue* line corresponds to the relational MMR using polynomial kernels, the *green* line to the relational MMR using radial basis function kernels, the *red* line to the MMMVM using polynomial kernels, the *light blue* line to the MMMVM using radial basis function kernels and the *pink* line to a "most frequent value"-imputation. The error bars depict the standard deviation over five repetitions of randomly splitting the data.

Appendix

6.A The Frank-Wolfe Algorithm

6.A.1 Problem Statement

For a compact convex subset $\mathcal{D}$ of a vector space and a convex differentiable real-valued function $f : \mathcal{D} \to \mathbb{R}$ the conditional gradient algorithm developed by Frank and Wolfe (1956) solves the optimization problem

$$\boxed{\begin{array}{ll} \min & f(x) \\ \text{w.r.t.} & x \in \mathcal{D}. \end{array}} \tag{6.94}$$

6.A.2 Algorithm

The underlying idea of the optimization procedure is to utilize the fact that the local error of the linear Taylor approximation centered at the point $\tilde{x}$

$$f(x) = f(\tilde{x}) + \nabla f(\tilde{x})'(x - \tilde{x}) + \underbrace{\mathcal{O}(\|x - \tilde{x}\|^2)}_{error\ magnitude} \tag{6.95}$$

is small.
Therefore, in order to solve Optimization Problem 6.94 an element $x^{(k)}$ of $\mathcal{D}$ is iteratively moved towards the minimizer s^* of the optimization problem

$$\boxed{\begin{array}{ll} \min & f(x^{(k)}) + \nabla f(x^{(k)})'(s - x^{(k)}) \\ \text{w.r.t.} & s \in \mathcal{D}, \end{array}} \tag{6.96}$$

i.e. $x^{(k+1)} := x^{(k)} + \gamma(s^* - x^{(k)})$. Note that the terms without dependency on s can be omitted, resulting in the linear programming problem

$$\boxed{\begin{array}{ll} \min & \nabla f(x^{(k)})'s \\ \text{w.r.t.} & s \in \mathcal{D}. \end{array}} \tag{6.97}$$

Linear programming problems are well studied and general purpose solvers such as the simplex method introduced by Dantzig (1951) exist.

In summary, the above considerations lead to Algorithm 1, in which different step-size de-

termination policies such as a line search for the optimal step-size would work as well.

Algorithm 1: Frank and Wolfe (1956)

Let $x^{(0)} \in \mathcal{D}$

for $k \in \{0, \ldots, K-1\}$ **do**

1. *Find direction:*
   $s^* := \arg\min_{s \in \mathcal{D}} \left\langle \nabla f(x^{(k)}), s \right\rangle$
2. *Determine step-size:*
   $\gamma := \frac{2}{k+2}$
3. *Update:*
   $x^{k+1} := x^{(k)} + \gamma(s^* - x^{(k)})$

end

.

6.A.3 Computational Complexity

According to Frank and Wolfe (1956) and Jaggi (2012), the iterates $x^{(k)}$ of Algorithm 1 satisfy

$$|f(x^{(k)}) - f(x^*)| \in \mathcal{O}(\frac{1}{k}), \tag{6.98}$$

in which x^* denotes the optimal solution for Optimization Problem 6.94.
Therefore, the sole unknown for the determination of the computational complexity of the Frank-Wolfe algorithm is the cost of one iteration. In Algorithm 1 the cost of one iteration is dominated by the computation of the gradient and the solution of the linear sub-problem, since the determination of the step-size and the update of the iterate are trivial operations.

Chapter 7

Conclusion

We have seen that the *kernel trick* is a powerful tool that allows for the solution of learning problems of various generalities in a uniform way. For example, the dual problem of the MMR is identical to the one of the SVM, except for the fact that in the MMR a more general output kernel than a linear one can be used. Thanks to the connections between *reproducing kernel Hilbert spaces* (RKHS), *positive definite kernels* and *feature space mappings* a rich toolset for the design of *reproducing kernels* and, in further consequence, for the choice of hypothesis spaces has been obtained. In addition, we have derived the *representer theorem* for the real valued case, which states that the optimal solution of the *regularized risk minimization problem* using a RKHS as hypothesis space can be represented exclusively in terms of the training data, and informally extended it to the structured output case by utilizing the notion of *joint kernels*. Therefore, potentially infinite dimensional hypothesis functions admit a finite dimensional representation.
Furthermore, we have addressed the *missing value problem* using the same techniques, by reinterpreting and thereby reformulating it as a *supervised learning problem*. Additionally, the proposed reformulation allows for the avoidance of problems – in particular, the problem of multiple missing value patterns – occurring when directly approaching the missing value problem and for seamless extension to the structured object valued case.
Since the *missing value problem* is a very general problem, it is of high relevance for various fields. For instance, in the pharmaceutical analysis of medicaments significant financial savings are achieved by predicting interactions between molecules, and thereby identifying pairs of molecules, for which an expensive measuring procedure is worthwhile. In this thesis an analogous problem, namely the problem of predicting object pair affordances in the scope of a robot learning task has been studied, and the proposed kernel methods have achieved decent results. Already with a small fraction of observed interactions most of the unknown interactions have been predicted correctly. In order to identify the "hard cases", i.e. the pairs of objects – or analogously the pairs of molecules – for which the a measurement would be worthwhile, a measurement of certainty about a prediction is necessary. One way to include such a measure into the proposed kernel methods, is to map the output space into a probabilistic one, and to use the probability of an output as its certainty.
Bearing in mind that in this thesis only finite output spaces have been considered, the solution of the pre-image problems required in the prediction step has been trivial. However, in many practical applications this is not the case. Therefore, further research towards more general structured output spaces, in particular infinite dimensional ones would be interesting in order to obtain a general learning framework. Despite this weakness, we have revised a fairly general learning framework, namely the MMMVM.

We live in a time, in which despite the flood of data, knowledge has remained a scarce resource. Understanding machine learning methods on a low level could help to shrink this gap

between data and knowledge. Of course, in that process machine learning methods should be rather regarded as helpful tool than as panacea, since there is a difference between separating and understanding the data. Researchers of the corresponding fields may be able to gain insights about their research question, by investigating which aspects of the data were the separating ones for a decently working machine learning algorithm applied on a related task.

Bibliography

N.I. Akhiezer and I.M. Glazman. *Theory of Linear Operators in Hilbert Space*. Dover Books on Mathematics. Dover Publications, 1993. ISBN 9780486677484. URL https://books.google.de/books?id=GTWMqiuvOAQC.

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950. URL http://dx.doi.org/10.2307/1990404.

K. Astikainen, L. Holm, E. Pitkänen, S. Szedmak, and J. Rousu. Towards structured output prediction of enzyme function. In *BMC Proceedings, 2(Suppl 4):S2*. 2008.

Gükhan H. Bakir, Thomas Hofmann, Bernhard Schölkopf, Alexander J. Smola, Ben Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data (Neural Information Processing)*. The MIT Press, 2007. ISBN 0262026171.

James Bennett, Stan Lanning, and Netflix Netflix. The netflix prize. In *In KDD Cup and Workshop in conjunction with KDD*, 2007.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521833787.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, September 1995. ISSN 0885-6125. doi: 10.1023/A:1022627411411. URL http://dx.doi.org/10.1023/A:1022627411411.

Corinna Cortes, Mehryar Mohri, and Jason Weston. A general regression framework for learning string-to-string mappings. 2006.

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

G. B. Dantzig. *Maximization of a Linear Function of Variables Subject to Linear Inequalities, in Activity Analysis of Production and Allocation*, chapter XXI. Wiley, New York, 1951.

C.K. Enders. *Applied Missing Data Analysis*. Methodology in the social sciences. Guilford Press, 2010. ISBN 9781606236390. URL https://books.google.at/books?id=MN8ruJd2tvgC.

Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956. ISSN 1931-9193. doi: 10.1002/nav.3800030109. URL http://dx.doi.org/10.1002/nav.3800030109.

Thomas Gärtner and Shankar Vembu. On structured output training: hard cases and an efficient alternative. *Machine Learning*, 76(2):227–242, 2009. ISSN 1573-0565. doi: 10.1007/s10994-009-5129-3. URL `http://dx.doi.org/10.1007/s10994-009-5129-3`.

Christel Geiss and Stefan Geiss. An introduction to probability theory, 2014. URL `https://www.jyu.fi/maths/en/research/stochastics/lecture-notes-for-stochastics-1/probability-1.pdf`.

Mustansar Ali Ghazanfar, Sandor Szedmak, and Adam Prugel-Bennett. Incremental kernel mapping algorithms for scalable recommender systems. In *Proceedings of the 2011 IEEE 23rd International Conference on Tools with Artificial Intelligence*, ICTAI '11, pages 1077–1084, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-0-7695-4596-7. doi: 10.1109/ICTAI.2011.183. URL `http://dx.doi.org/10.1109/ICTAI.2011.183`.

Mustansar Ali Ghazanfar, Adam PrüGel-Bennett, and Sandor Szedmak. Kernel-mapping recommender system algorithms. *Inf. Sci.*, 208:81–104, November 2012. ISSN 0020-0255. doi: 10.1016/j.ins.2012.04.012. URL `http://dx.doi.org/10.1016/j.ins.2012.04.012`.

Valery Glivenko and Francesco Cantelli. Sulla determinazione empirica della legge di probabilita. *Giornale dell'Istituto Italiano degli Attuari*, (Bd. 4):221–424, 1933. ISSN 0021-2482. URL `https://books.google.at/books?id=QPaFAAAAIAAJ`.

Chunhui Gu. Reproducing kernel hilbert spaces, 2008. URL `http://people.eecs.berkeley.edu/~bartlett/courses/281b-sp08/7.pdf`.

Jacques Hadamard. Sur les problèmes aux dérivés partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52, 1902.

Paul R. Halmos. *Finite-Dimensional Vector Spaces*. 1974. ISBN 0-387-90093-4. Reprint of the Second edition published by Van Nostrand, Princeton, NJ, 1958.

Tobias Hell and Lukas Neumann. Topologie und funktionalanalysis, 2012. URL `https://numerical-analysis.uibk.ac.at/images/User-Data/Tobias-Hell/Analysis4SS12.pdf`.

Ralf Herbrich. *Learning Kernel Classifiers: Theory and Algorithms*. MIT Press, Cambridge, MA, USA, 2001. ISBN 026208306X.

David Hilbert. Grundzüge einer allgemeinen theorie der linearen integralrechnungen. (erste mitteilung). *Nachrichten von der Gesellschaft der Wissenschaften zu GÃűttingen, Mathematisch-Physikalische Klasse*, 1904:49–91, 1904. URL `http://eudml.org/doc/58572`.

David Hilbert. Grundzüge einer allgemeinen theorie der linearen integralgleichungen. In A. Pietsch, editor, *Integralgleichungen und Gleichungen mit unendlich vielen Unbekannten*, volume 11 of *Teubner-Archiv zur Mathematik*, pages 8–171. Vieweg+Teubner Verlag, 1989. ISBN 978-3-322-00681-3. doi: 10.1007/978-3-322-84410-1_1. URL `http://dx.doi.org/10.1007/978-3-322-84410-1_1`.

Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. A review of kernel methods in machine learning, 2006.

John E. Hopcroft and Jeffrey D. Ullman. *Introduction To Automata Theory, Languages, And Computation*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1990. ISBN 020102988X.

Martin Jaggi. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. *submitted*, 2012.

Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Recommender Systems: An Introduction*. Cambridge University Press, New York, NY, USA, 1st edition, 2010. ISBN 0521493366, 9780521493369.

C. R. Johnson. Matrix Completion Problems: A Survey. 1990.

Rahul Kidambi, Vinod Nair, Sundararajan Sellamanickam, and S. Sathiya Keerthi. A structured prediction approach for missing value imputation. *CoRR*, abs/1311.2137, 2013. URL `http://arxiv.org/abs/1311.2137`.

Senka Krivić, Sandor Szedmak, Hanchen Xiong, and Justus Piater. Learning missing edges via kernels in partially-known graphs. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2015. URL `https://iis.uibk.ac.at/public/papers/Krivic-2015-ESANN.pdf`.

H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, Calif., 1951. University of California Press. URL `http://projecteuclid.org/euclid.bsmsp/1200500249`.

Harold W. Kuhn. Nonlinear programming: a historical view. *ACM Sigmap Bulletin*, pages 6–18, 1982. doi: 10.1145/1111278.1111279.

John Lafferty, Xiaojin Zhu, and Yan Liu. Kernel conditional random fields: Representation and clique selection. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 64–, New York, NY, USA, 2004. ACM. ISBN 1-58113-838-5. doi: 10.1145/1015330.1015337. URL `http://doi.acm.org/10.1145/1015330.1015337`.

Christoph H. Lampert and Matthew B. Blaschko. Structured prediction by joint kernel support estimation. *Mach. Learn.*, 77(2-3):249–269, December 2009. ISSN 0885-6125. doi: 10.1007/s10994-009-5111-0. URL `http://dx.doi.org/10.1007/s10994-009-5111-0`.

Roderick J A Little and Donald B Rubin. *Statistical Analysis with Missing Data*. John Wiley & Sons, Inc., New York, NY, USA, 1986. ISBN 0-471-80254-9.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. Text classification using string kernels. *J. Mach. Learn. Res.*, 2:419–444, March 2002. ISSN 1532-4435. doi: 10.1162/153244302760200687. URL `http://dx.doi.org/10.1162/153244302760200687`.

Olvi L. Mangasarian. Exact 1-norm support vector machines via unconstrained convex differentiable minimization. *JOURNAL OF MACHINE LEARNING RESEARCH*, 7:1517–1530, 2006.

J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 209(441-458):415–446, 1909. ISSN 0264-3952. doi: 10.1098/rsta.1909.0016.

Charles A. Micchelli and Massimiliano A. Pontil. On learning vector-valued functions. *Neural Comput.*, 17(1):177–204, January 2005. ISSN 0899-7667. doi: 10.1162/0899766052530802. URL `http://dx.doi.org/10.1162/0899766052530802`.

Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997. ISBN 0070428077, 9780070428072.

Sebastian Nowozin and Christoph H. Lampert. Structured learning and prediction in computer vision. *Found. Trends. Comput. Graph. Vis.*, 6(3–4):185–365, March 2011. ISSN 1572-2740. doi: 10.1561/0600000033. URL `http://dx.doi.org/10.1561/0600000033`.

Karl Pearson. Notes on regression and inheritance in the case of two parents. pages 240–242, 1895. URL `https://books.google.at/books?id=60aL0zlT-90C`.

Tomaso Poggio and Lorenzo Rosasco. Machine learning: a regularization approach, mit-9.520 lectures notes, 2015.

Tomaso Poggio, Ryan Rifkin, Sayan Mukherjee, and Partha Niyogi. General conditions for predictivity in learning theory. *Nature*, 428(6981):419–422, Mar 2004. ISSN 0028-0836. doi: 10.1038/nature02341. URL `http://dx.doi.org/10.1038/nature02341`.

Manisha Pujari and Rushed Kanawati. Link prediction in multiplex networks. *Networks and Heterogeneous Media*, 10(1):17–35, 2015. ISSN 1556-1801. doi: 10.3934/nhm.2015.10.17. URL `http://aimsciences.org/journals/displayArticlesnew.jsp?paperID=10837`.

M. Mostafizur Rahman and Darryl N. Davis. Fuzzy unordered rules induction algorithm used as missing value imputation methods for k-mean clustering on real cardiovascular data, 2013.

W. Rudin. *Functional Analysis*. International series in pure and applied mathematics. McGraw-Hill, 2006. ISBN 9780070619883. URL `https://books.google.de/books?id=l7XFfDmjp5IC`.

Craig Saunders, Alexander Gammerman, and Volodya Vovk. Ridge regression learning algorithm in dual variables. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 515–521, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-556-8. URL `http://dl.acm.org/citation.cfm?id=645527.657464`.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521813972.

Morton Slater. Lagrange multipliers revisited. In Giorgio Giorgi and Tinne Hoff Kjeldsen, editors, *Traces and Emergence of Nonlinear Programming*, pages 293–306. Springer Basel, 2014. ISBN 978-3-0348-0438-7. doi: 10.1007/978-3-0348-0439-4_14. URL `http://dx.doi.org/10.1007/978-3-0348-0439-4_14`.

Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009:4:2–4:2, January 2009. ISSN 1687-7470. doi: 10.1155/2009/421425. URL `http://dx.doi.org/10.1155/2009/421425`.

S. Szedmak, J. Shawe-Taylor, and E. Parado-Hernandez. Learning via linear operators: Maximum margin regression. In *PASCAL Research Reports, http://eprints.pascal-network.org/*. 2005.

S. Szedmak, E. Ugur, and J. Piater. Knowledge propagation and relation learning for predicting action effects. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 623–629, Sept 2014a. doi: 10.1109/IROS.2014.6942624.

Sandor Szedmak, Emre Ugur, and Justus Piater. Knowledge Propagation and Relation Learning for Predicting Action Effects. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 623–629. IEEE, 09 2014b. doi: 10.1109/IROS.2014.6942624. URL `https://iis.uibk.ac.at/public/papers/Szedmak-2014-IROS.pdf`.

Sandor Szedmak, Senka Krivić, and Hanchen Xiong. Learning Interrelations via Incomplete Multivalued Mappings. 2015.

Xin Lu Tan. Notes on reproducing kernel hilbert space. 2014.

B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *NIPS 2003*. 2003.

A.N. Tikhonov and V.I.A. Arsenin. *Solutions of ill-posed problems*. Scripta series in mathematics. Winston, 1977. ISBN 9780470991244. URL `https://books.google.at/books?id=ECrvAAAAMAAJ`.

François Trèves. Topological vector spaces, distributions and kernels. New York-London: Academic Press 1967. XVI, 565 p. (1967)., 1967.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6(Sep): 1453–1484, 2005a.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, 6: 1453–1484, December 2005b. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1046920.1088722`.

V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

Peter Wagner. Skriptum zur vorlesung funktionenanalysis, 2004.

Grace Wahba. Reproducing kernel hilbert spaces-two brief reviews. 2003. URL `http://www.stat.wisc.edu/techreports/tr1079.pdf`.

Jason Weston, Olivier Chapelle, André Elisseeff, Bernhard Schölkopf, and Vladimir Vapnik. Kernel Dependency Estimation. In Suzanna Becker, Sebastian Thrun, Klaus Obermayer, Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *NIPS*, pages 873–880. MIT Press, 2002. ISBN 0-262-02550-7. URL `http://dblp.uni-trier.de/rec/bibtex/conf/nips/WestonCESV02`.

Jason Weston, Gökhan Bakir, Olivier Bousquet, Tobias Mann, William Stafford Noble, and Bernhard Schölkopf. Joint kernel maps. In Gökhan H. Bakir, Thomas Hofmann, Bernhard Schölkopf, Alexander J. Smola, Ben Taskar, and S. V. N. Vishwanathan, editors, *Predicting Structured Data (Neural Information Processing)*, chapter 4, pages 80–96. The MIT Press, 2007. ISBN 0262026171.

Hanchen Xiong, Sandor Szedmak, and Justus Piater. Scalable, Accurate Image Annotation with Joint SVMs and Output Kernels. *Neurocomputing*, 2015. URL `https://iis.uibk.ac.at/public/papers/Xiong-2015-NEUCOM.pdf`. To appear.

Peng Yifan. Illustration of the maximum margin hyperplane. `http://blog.pengyifan.com/tikz-example-svm-trained-with-samples-from-two-classes/`. Accessed: 2015-12-11.