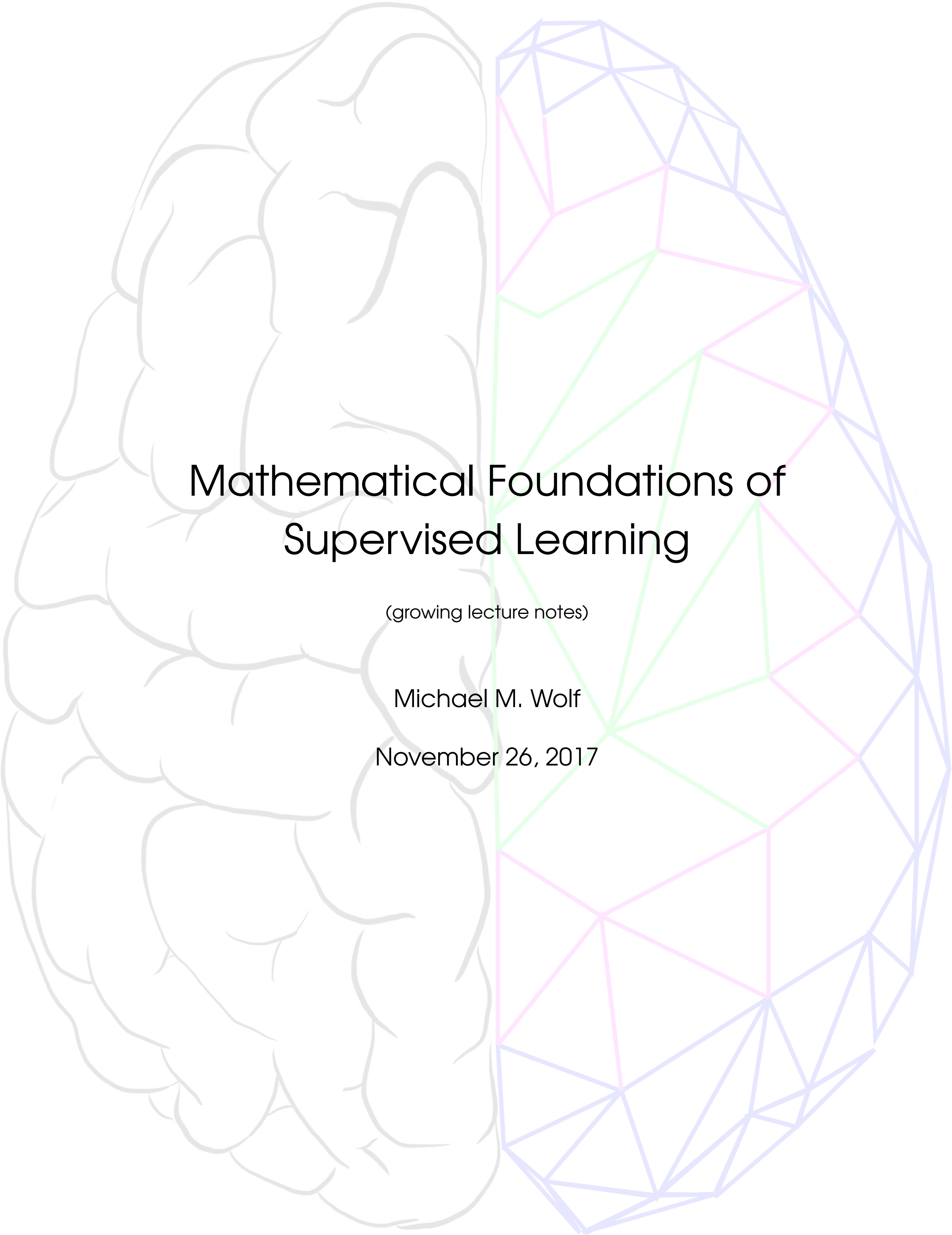


Mathematical Foundations of Supervised Learning

(growing lecture notes)

Michael M. Wolf

November 26, 2017

Contents

Introduction

These are (incomplete but hopefully growing) lecture notes of a course taught first in summer 2016 at the department of mathematics at the Technical University of Munich. The course is meant to be a concise introduction to the mathematical results of the field.

What is *machine learning*? Machine learning is often considered as part of the field of artificial intelligence, which in turn may largely be regarded as a subfield of computer science. The aim of machine learning is to exploit data in order to devise complex models or algorithms in an automated way. So machine learning is typically used whenever large amounts of data are available and when one aims at a computer program that is (too) difficult to program 'directly'. Standard examples are programs that recognize faces, handwriting or speech, drive cars, recommend products, translate texts or play Go. These are hard to program from scratch so that one uses machine learning algorithms that produce such programs from large amounts of data.

Two main branches of the field are *supervised learning* and *unsupervised learning*. In supervised learning a learning algorithm is a device that receives 'labeled training data' as input and outputs a program that predicts the label for unseen instances and thus generalizes beyond the training data. Examples of sets of labeled data are emails that are labeled 'spam' or 'no spam' and medical histories that are labeled with the occurrence or absence of a certain disease. In these cases the learning algorithm's output would be a spam filter and a diagnostic program, respectively.

In contrast, in unsupervised learning there is no additional label attached to the data and the task is to identify and/or model patterns in the data. Unsupervised learning is for instance used to compress information or to organize data. Whereas unsupervised learning is more descriptive, supervised learning is more predictive. In the following, we will exclusively deal with the latter.

A first coarse classification of supervised learning algorithms is in terms of the chosen *representation*, which determines the basic structure of the generated programs. Common ones are:

- Decision trees

- Nearest neighbors

- Neural networks
- Support vector machines and kernel methods

These representations, though quite different in nature, have two important things in common: they enable *optimization* and they form *universal hierarchies*.

The fact that their structure enables optimization is crucial in order to identify an instance (i.e., a program) that fits the data and presumably performs well regarding future predictions. This optimization is typically done in a greedy manner.

Forming a universal hierarchy means that the representation contains more and more refined levels that, in principle, are capable of representing every possibility or at least approximating every possibility to arbitrary accuracy.

Only few such representations are known and the above examples (together with variations on the theme and combinations thereof) already seem to cover most of the visible universe.

We will focus on the last two of the mentioned representations, neural networks and support vector machines, which are arguably the most sophisticated and most powerful ones. To begin with, however, we will have a closer look at the general statistical framework of supervised learning theory.

Chapter 1

Learning Theory

1.1 Statistical framework

In this section we set up the standard statistical framework for supervised learning theory.

Input of the learning algorithm is the *training data* that is a finite sequence $S = \big((x_1, y_1), \ldots, (x_n, y_n)\big)$ of pairs from $\mathcal{X} \times \mathcal{Y}$. y_i is called the *label* corresponding to x_i.

Output of the learning algorithm is a function $h : \mathcal{X} \to \mathcal{Y}$, called *hypothesis*, that aims at predicting $y \in \mathcal{Y}$ for arbitrary $x \in \mathcal{X}$, especially for those not contained in the training data. Formally, a learning algorithm can thus be seen as a map $\mathcal{A} : \cup_{n\in\mathbb{N}}(\mathcal{X} \times \mathcal{Y})^n \to \mathcal{Y}^{\mathcal{X}}$. We will denote its range, i.e., the set of functions that can be output and thus be represented by the learning algorithm, by $\mathcal{F}$. From a computer science perspective the learning algorithm is an algorithm that, upon input of the training data S, outputs a computer program described by $h_S := \mathcal{A}(S) \in \mathcal{F}$.

Probabilistic assumption. The pairs (x_i, y_i) are treated as values of random variables (X_i, Y_i) that are identically and independently distributed according to some probability measure P over $\mathcal{X} \times \mathcal{Y}$. We will throughout assume that the corresponding σ-algebra is a product of Borel σ-algebras w.r.t. the usual topologies. All considered functions will be assumed to be Borel functions. Expectations w.r.t. P and P^n will be denoted by $\mathbb{E}$ and $\mathbb{E}_S$, respectively. If we want to emphasize that, for instance, S is distributed according to P^n we will use the more explicit notation $\mathbb{E}_{S\sim P^n}$. Similarly, probabilities of events A and B w.r.t. P and P^n will be denoted by $\mathbb{P}[A]$ and $\mathbb{P}_S[B]$, respectively. It is throughout assumed that P does not only govern the distribution of the training data, but also the one of future, yet unseen instances of data points.

Goal of the learning algorithm is to find a good hypothesis h w.r.t. a suitably chosen *loss function* $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ that measures how far $h(x)$ is from the respective y. The smaller the average loss, called *risk*[1] and given by

$$R(h) := \int_{\mathcal{X}\times\mathcal{Y}} L\big(y, h(x)\big)dP(x,y), \tag{1.1}$$

the better the hypothesis. The challenge is, that the probability measure P is unknown, only the training data S is given. Hence, the task of the learning algorithm is to minimize the risk without being able to evaluate it directly.

Depending on whether $\mathcal{Y}$ is continuous or discrete one distinguishes two types of learning problems with different loss functions:

Regression

Regression deals with continuous $\mathcal{Y}$. The most common loss function in the case $\mathcal{Y} = \mathbb{R}$ is the quadratic loss $L\big(y, h(x)\big) = |y - h(x)|^2$ leading to the L_2-*risk* also known as *mean squared error* $R(h) = \mathbb{E}\left[|Y - h(X)|^2\right]$. For many reasons this is a mathematically convenient choice. One of them is that the function that minimizes the risk can be handled:

Theorem 1.1: Regression function minimizes L_2-risk

In the present context let $h : \mathcal{X} \to \mathcal{Y} = \mathbb{R}$ be a Borel function and assume that $\mathbb{E}\left[Y^2\right]$ and $\mathbb{E}\left[h(X)^2\right]$ are both finite. Define the *regression function* as conditional expectation $r(x) := \mathbb{E}(Y|X = x)$. Then the L_2-risk of h can be written as

$$R(h) = \mathbb{E}\left[|Y - r(X)|^2\right] + \mathbb{E}\left[|h(X) - r(X)|^2\right]. \tag{1.2}$$

Note: The first term on the r.h.s. in Eq.(1.2) vanishes if there is a deterministic relation between x and y, i.e., if $P(y|x) \in \{0,1\}$. In general, it can be regarded as unavoidable inaccuracy that is due to noise or due to the lack of information content in X about Y. The second term contains the dependence on h and is simply the squared L_2-distance between h and the regression function r. Minimizing the risk thus means minimizing the distance to the regression function.

Proof. (sketch) Consider the real Hilbert space $L_2(\mathcal{X}\times\mathcal{Y}, P)$ with inner product $\langle\psi,\phi\rangle := \mathbb{E}\left[\psi\phi\right]$. h can be considered as an element of the closed subspace of functions that only depend on x and are constant w.r.t. y. The function r also represents an element of that subspace and since the conditional expectation[2] is, by construction, the orthogonal projection into that subspace, we have $\langle y - r, h - r\rangle = 0$. With this, Pythagoras' identity yields the desired result

$$||y - h||^2 = ||y - r||^2 + ||h - r||^2.$$

□

[1] The risk also runs under the name *out-of-sample error* or *generalization error*.

[2] If there is a probability density $p(x,y)$, the conditional expectation is given by $E(Y|X = x) = \int_{\mathcal{Y}} y\ p(x,y)/p(x)dy$, if the marginal $p(x)$ is non-zero. For a general treatment of conditional expectations see for instance [15], Chap.23.

Classification

Classification deals with discrete $\mathcal{Y}$, in which case a function from $\mathcal{X}$ to $\mathcal{Y}$ is also called a *classifier*. The most common loss function in this scenario is the *0-1 loss* $L(y, y') = 1 - \delta_{y,y'}$ so that the corresponding risk is nothing but the error probability $R(h) = \mathbb{P}\left[h(X) \neq Y\right] = \mathbb{E}\left[\mathbb{1}_{h(X)\neq Y}\right]$. We will at the beginning often consider *binary classification* where $\mathcal{Y} = \{-1, 1\}$. The error probability in binary classification is minimized by the *Bayes classifier*

$$b(x) := \operatorname{sgn}\big(\mathbb{E}\left[Y|X = x\right]\big). \tag{1.3}$$

1.2 Error decomposition

How can the learning algorithm attempt to minimize the risk $R(h)$ over its accessible hypotheses $h \in \mathcal{F}$ without knowing the underlying distribution P? There are two helping hands. The first one is prior knowledge. This can for instance be hidden in the choice of $\mathcal{F}$ and the way the learning algorithm chooses a hypothesis from this class. Second, although $R(h)$ cannot be evaluated directly, the average loss can be evaluated on the data S, which leads to the *empirical risk*, also called *in-sample error*

$$\hat{R}(h) := \frac{1}{n}\sum_{i=1}^{n} L\big(y_i, h(x_i)\big). \tag{1.4}$$

The approach of minimizing $\hat{R}$ is called *empirical risk minimization* (ERM). In particular if $|\mathcal{Y}| < \infty$, then there always exists a minimizer $\hat{h} \in \mathcal{F}$ that attains $\inf_{h\in\mathcal{F}} \hat{R}(h) = \hat{R}(\hat{h})$ since the functions are only evaluated at a finite number of points, which effectively restricts $\mathcal{F}$ to a finite space.

In general, ERM is a computationally hard task—an issue that we will discuss in greater detail in the following chapters, where specific representations are chosen. At this point, we will make the idealizing assumption that ERM can be performed. Keeping in mind, however, that only in few cases an efficient algorithm or a closed form solution for ERM is known, like in the following examples.

Example 1.1 (Linear regression). Let $\mathcal{X} \times \mathcal{Y} = \mathbb{R}^d \times \mathbb{R}$ and $\mathcal{F} := \{h : \mathcal{X} \to \mathcal{Y} \mid \exists v \in \mathbb{R}^d : h(x) = \langle v, x\rangle\}$ be the class of linear functions. The minimizer of the empirical risk (w.r.t. the quadratic loss)

$$\hat{R}(v) := \frac{1}{n}\sum_{i=1}^{n} \big(y_i - \langle v, x_i\rangle\big)^2, \tag{1.5}$$

can be determined by realizing that the condition $\nabla\hat{R}(v) = 0$ can be rewritten as linear equation $Av = b$, where $A := \sum_i x_i x_i^T$ and $b := \sum_i y_i x_i$. This is solved by $v = A^{-1}b$ where the inverse is computed on range(A).

Example 1.2 (Polynomial regression). Let $\mathcal{X} \times \mathcal{Y} = \mathbb{R} \times \mathbb{R}$ and $\mathcal{F} := \{h : \mathbb{R} \to \mathbb{R} \mid \exists a \in \mathbb{R}^{m+1} : h(x) = \sum_{k=0}^m a_k x^k\}$ be the set of polynomials of degree m. In order to find the ERM w.r.t. the quadratic loss, define $\psi : \mathbb{R} \to \mathbb{R}^{m+1}$, $\psi(x) := (1, x, x^2, \dots, x^m)$. Then the empirical risk can be written as

$$\frac{1}{n}\sum_{i=1}^n \Big(y_i - \sum_{k=0}^m a_k x_i^k\Big)^2 = \frac{1}{n}\sum_{i=1}^n \big(y_i - \langle a, \psi(x_i)\rangle\big)^2.$$

Hence, it is of the form in Eq.(1.5) and one can proceed exactly as in the case of linear regression in Exp.1.1. Note that instead of using the monomial basis in the components of ψ, we could as well use a Fourier basis, Wavelets or other basis functions and again follow the same approach.

If n, the size of the training data set, is sufficiently large, one might hope that $\hat{R}(h)$ is not too far from $R(h)$ so that ERM would come close to minimizing the risk. The formalization and quantification of this hope is essentially the content of the remaining part of this chapter.

To this end, and also for a better understanding of some of the main issues in supervised machine learning, it is useful to look at the following error decompositions.

Let $R^* := \inf_h R(h)$ be the so-called *Bayes risk*, where the infimum is taken over all measurable functions $h : \mathcal{X} \to \mathcal{Y}$, and let $R_{\mathcal{F}} := \inf_{h\in\mathcal{F}} R(h)$ quantify the optimal performance of a learning algorithm with range $\mathcal{F}$. Assume further that a hypothesis $\hat{h} \in \mathcal{F}$ minimizes the empirical, i.e., $\hat{R}(\hat{h}) \leq \hat{R}(h)\ \forall h \in \mathcal{F}$. Then we can decompose the difference between the risk of a hypothesis h and the optimal Bayes risk as

$$R(h) - R^* = \underbrace{\big(R(h) - R(\hat{h})\big)}_{\text{optimization error}} \quad + \quad \underbrace{\big(R(\hat{h}) - R_{\mathcal{F}}\big)}_{\text{estimation error}} \quad + \quad \underbrace{\big(R_{\mathcal{F}} - R^*\big)}_{\text{approximation error}} \quad . \tag{1.6}$$

The *approximation error* does neither depend on the hypothesis nor on the data. It quantifies how well the hypothesis class $\mathcal{F}$ is suited for the problem under consideration. The *optimization error* depends on how good the optimization thet led to hypothesis h is relative to ideal empirical risk minimization. The *estimation error* measures how well the empirical risk minimizer $\hat{h}$ performs relative to a true risk minimizer in $\mathcal{F}$. By the law of large numbers the estimation error is expected to decrease with the size n of the training data set and to vanish asymptotically. The estimation error can be bounded by:

$$\begin{aligned} R\big(\hat{h}\big) - R_{\mathcal{F}} &= R\big(\hat{h}\big) - \hat{R}\big(\hat{h}\big) + \sup_{h\in\mathcal{F}} \Big(\hat{R}\big(\hat{h}\big) - R(h)\Big) \\ &\leq 2\sup_{h\in\mathcal{F}} \Big|\hat{R}(h) - R(h)\Big|. \end{aligned} \tag{1.7}$$

Bounds on difference between the risk and the empirical risk (or, using synonyms, between the out-of-sample error and the in-sample error) are called *generalization bounds.* They quantify how well the hypothesis generalizes from

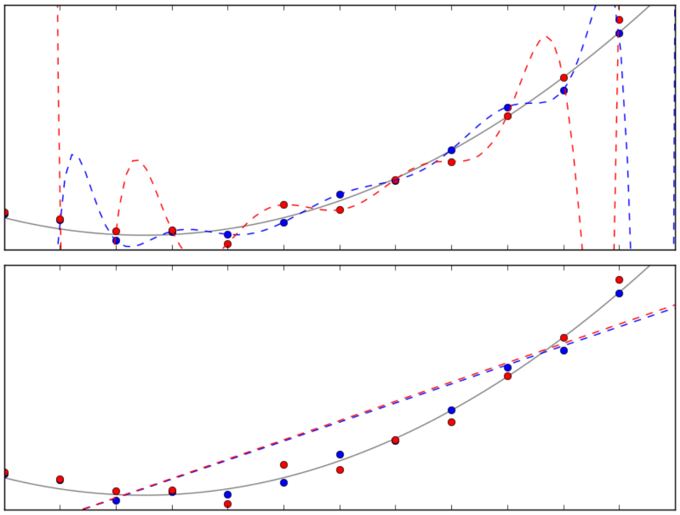


Figure 1.1: Bias-variance trade-off in polynomial regression: two samples (red and blue points) are drawn from the same noisy version of a quadratic function (gray). If we fit high degree polynomials (top image), there is a large *variance* from sample to sample, but a small *bias* in the sense that the average curve asymptotically matches the underlying distribution well. If affine functions are used instead (bottom image), the variance is reduced at the cost of a large bias.

the observed data to unseen cases. Uniform generalization bounds, as desired by Eq.(1.7), will be derived in the following sections.

Let us for the moment assume that the learning algorithm performs ideal ERM so that the optimization error vanishes. Then we are typically faced with a trade-off between the estimation error and the approximation error: while aiming at a smaller approximation error suggests to take a richer hypothesis class $\mathcal{F}$, the data required to keep the estimation error under control unfortunately turns out to grow rapidly with the size or complexity of $\mathcal{F}$ (cf. following sections). A closely related trade-off runs under the name *bias-variance trade-off*. It has its origin in a refinement of the decomposition in Thm.1.1 and is exemplified in Fig.1.1.

Theorem 1.2: Noise-bias-variance decomposition

In the setup of Thm.1.1 consider a fixed learning algorithm that outputs a hypothesis h_S upon input of $S \in (\mathcal{X} \times \mathcal{Y})^n$. Regard S as a random variable, distributed according to P^n and define $\bar{h}(x) := \mathbb{E}_S\left[h_S(x)\right]$ the expected prediction for a fixed x. If the expected risk $\mathbb{E}_S\left[R(h_S)\right]$ is finite, then it is equal to

$$\underbrace{\mathbb{E}\left[|Y-r(X)|^2\right]}_{\text{noise}}+\underbrace{\mathbb{E}\left[|\bar{h}(X)-r(X)|^2\right]}_{\text{bias}^2}+\underbrace{\mathbb{E}\left[\mathbb{E}_S\left[|h_S(X)-\bar{h}(X)|^2\right]\right]}_{\text{variance}} \tag{1.8}$$

Proof. We take the expectation $\mathbb{E}_S$ of Eq.(1.2) when applied to h_S and observe that the first term on the r.h.s. is independent of S. For the second term we obtain

$$\begin{aligned}\mathbb{E}_S\left[\mathbb{E}\left[|h_S(X)-r(X)|^2\right]\right] &= \mathbb{E}\left[\mathbb{E}_S\left[|h_S(X)-\bar{h}(X)+\bar{h}(X)-r(X)|^2\right]\right]\\ &= \mathbb{E}\left[|\bar{h}(X)-r(X)|^2\right]\\ &\quad+\mathbb{E}\left[\mathbb{E}_S\left[|h_S(X)-\bar{h}(X)|^2\right]\right]\\ &\quad+2\mathbb{E}\left[\mathbb{E}_S\left[\big(h_S(X)-\bar{h}(X)\big)\big(\bar{h}(X)-r(X)\big)\right]\right].\end{aligned}$$

The term in the last line vanishes since $\big(\bar{h}(X)-r(X)\big)$ is independent of S and $\mathbb{E}_S\left[\big(h_S(X)-\bar{h}(X)\big)\right]=0$. □

As can bee seen in the example of polynomial regression in Fig.1.1, if we increase the size of $\mathcal{F}$, then the squared bias is likely to decrease while the variance will typically increase (while the noise is unaffected).

There is a third incarnation of the phenomenon behind a dominating variance or estimation error that is called *overfitting*. All together these are consequences of choosing $\mathcal{F}$ too large so that it contains exceedingly complex hypotheses, which might be chosen by the learning algorithm.[3]

As long as ideal ERM is considered, the three closely related issues just discussed all ask for a balanced choice of $\mathcal{F}$. In order to achieve this and to get confidence in the quality of the choice many techniques have been developed. First of all, the available labeled data is split into two disjoint sets, *training data* and *test data*. While the former is used to train/learn/optimize and eventually output a hypothesis h_S, the latter is used to evaluate the performance of h_S. There is sometimes a third separate set, the *validation data*, that is used to tune free parameters of the learning algorithm. In many cases, however, training data is too precious to set aside a separate validation sample and then validation is done on the training data by a technique called *cross-validation*.

In order to prevent the learning algorithm from choosing overly complex hypotheses, ERM is often modified in practice. One possibility, called *structural risk minimization*, is to consider a sequence of hypotheses classes $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \mathcal{F}_3 \subset \ldots$ of increasing complexity and to optimize the empirical error plus a penalty term that takes into account the complexity of the underlying class.

A smoother variant of this idea is called *regularization*, where a single hypotheses class $\mathcal{F}$ is chosen together with a *regularizer*, i.e., a complexity penalizing function $\varrho : \mathcal{F} \to \mathbb{R}_+$, and one minimizes the *regularized empirical risk*

[3] Formally, one defines a hypothesis $h \in \mathcal{F}$ to be *overfitting* if there exists a hypothesis $h' \in \mathcal{F}$ such that $\hat{R}(h) < \hat{R}(h')$, but $R(h) > R(h')$. That is, h overfits the data in the sense that the empirical error is overly optimistic.

$\hat{R}(h) + \varrho(h)$. If $\mathcal{F}$ is embedded in a vector space, a very common scheme is *Tikhonov regularization*, where $\varrho(h) := ||Ah||^2$ for some linear map A, which is often simply a multiple of the identity. The remaining free parameter is then chosen for instance by cross-validation.

In the end, however, choosing a good class $\mathcal{F}$ and/or a good way to pick not too complex hypotheses from $\mathcal{F}$ is to some extent an art. In practice, one makes substantial use of heuristics and of explicit or implicit prior knowledge about the problem. In Sec.1.4 we will see a formalization of the fact that there is no a priori best choice.

A central goal of statistical learning theory is to provide generalization bounds. The simplest way to obtain such bounds in practice, is to look at a test set, which will be done in the remaining part of this section. Deriving generalization bounds without a test set is more delicate, but from a theoretical point of view desirable. It is usually based on two ingredients: (i) a so-called *concentration inequality*, which can be seen as a non-asymptotic version of the law of large numbers, and (ii) a bound on a relevant property of the learning algorithm. This property could be the size or complexity of its range, the stability, compressibility, description length or memorization-capability of the algorithm. All these lead to different generalization bounds and some of them will be discussed in the following sections. The traditionally dominant approach is to consider only the range $\mathcal{F}$ of the algorithm. This seems well justified as long as idealized ERM is considered (as ERM treats all hypotheses in $\mathcal{F}$ equally) and we will have a closer look at various themes along this line until Sec.1.10 (incl.). In Sec.1.11-1.13 we will exploit more details and other properties of the learning algorithms and discuss approaches in which the range $\mathcal{F}$ plays essentially no role anymore. This class of approaches is potentially better suited to deal with the fact that in practice learning algorithms often deviate from their ERM-type idealization.

Generalization bound from test error

Before we discuss how generalization bounds can be obtained prior to looking at the test error, we will look at the test error and ask what kind of generalization bounds can be obtained from it. To this end, we will assume that there is a test set $T \in (\mathcal{X} \times \mathcal{Y})^m$ which has been kept aside so that the hypothesis h, which has been picked by the learning algorithm depending on a training data set S, is statistically independent of T. More precisely, we assume that the elements of both, S and T, are distributed identically and independently, governed by a probability measure P on $\mathcal{X} \times \mathcal{Y}$, and that h may depend on S, but not on T. Testing the hypothesis on T then leads to the empirical test error

$$\hat{R}_T(h) := \frac{1}{|T|} \sum_{(x,y)\in T} L\big(y, h(x)\big).$$

If $R = R(h)$ is the error probability, i.e., the $0-1$ loss is considered, then the empirical test error $\hat{R}_T = \hat{R}_T(h)$ is a multiple of $1/m$ and we can express the probability that it is at most k/m in terms of the cumulative binomial distribution

$$\mathbb{P}_{T\sim P^m}\left[\hat{R}_T \leq \frac{k}{m}\right] = \sum_{j=0}^{k} \binom{m}{j} R^j (1-R)^{m-j} =: \mathrm{Bin}(m,k,R). \tag{1.9}$$

Since we want to deduce a bound on R from $\hat{R}_T$, we have to invert this formula and introduce

$$\mathrm{Binv}(m,k,\delta) := \max\left\{\, p \in [0,1] \mid \mathrm{Bin}(m,k,p) \geq \delta\right\}. \tag{1.10}$$

This leads to:

Theorem 1.3: Clopper-Pearson bound

Let $R(h)$ be the error probability of a hypothesis $h : \mathcal{X} \to \mathcal{Y}$. With probability at least $1-\delta$ over an i.i.d. draw of a test set $T \in (\mathcal{X}\times\mathcal{Y})^m$:

$$R(h) \leq \mathrm{Binv}\big(m, m\hat{R}_T(h), \delta\big) \leq \hat{R}_T(h) + \sqrt{\frac{\ln\frac{1}{\delta}}{2m}}. \tag{1.11}$$

Moreover, if $\hat{R}_T(h) = 0$, then

$$R(h) \leq \frac{\ln\frac{1}{\delta}}{m}. \tag{1.12}$$

Proof. (Sketch) By the definition of Binv, $R(h) \leq \mathrm{Binv}(m, m\hat{R}_T(h))$ holds with probability at least $1-\delta$. In order to further bound this in a more explicit way, we exploit that the cumulative binomial distribution can be bounded by $\mathrm{Bin}(m,k,p) \leq \exp\left[-2m(p-k/m)^2\right]$. Inserting this into the definition of Binv, we get

$$\begin{aligned} \mathrm{Binv}(m,k,\delta) &\leq \max\left\{p \mid \exp\left[-2m(p-k/m)^2\right] \geq \delta\right\} \\ &= \frac{k}{m} + \sqrt{\frac{\ln\frac{1}{\delta}}{2m}}, \end{aligned} \tag{1.13}$$

which leads to Eq.(1.11). Following the same reasoning, Eq.(1.12) is obtained from the bound $\mathrm{Bin}(m,0,p) = (1-p)^m \leq e^{-pm}$. □

The bound on $R(h)$ in terms of Binv is optimal, by definition. Of course, in practice Binv should be computed numerically in order to get the best possible bound. The explicit bounds given in Thm.1.3, however, display the right asymptotic behavior, which we will also find in the generalization bounds that are expressed in terms of the training error in the following sections: while in general the difference between risk and empirical risk is inversely proportional to the square root of the sample size, this square root can be dropped under special assumptions.

1.3 PAC learning bounds

Since we consider the training data to be random, we have to take into account the possibility to be unlucky with the data in the sense that it may not be a fair sample of the underlying distribution. Hence, useful bounds for instance on $|\hat{R}(h) - R(h)|$ will have to be probabilistic. What we can reasonably hope for, is that, under the right conditions, we obtain guarantees of the form

$$\mathbb{P}_S\left[|\hat{R}(h) - R(h)| \geq \epsilon\right] \leq \delta.$$

Bounds of this form are the heart of the *probably approximately correct* (PAC) learning framework. The bounds in this context are *distribution-free*. That is, ϵ and δ do not depend on the underlying probability measure, which is unknown in practice. The simplest bound of this kind concerns cases where a deterministic assignment of labels is assumed that can be perfectly described within the chosen hypotheses class:

Theorem 1.4: PAC bound for deterministic, realizable scenarios

Let $\epsilon \in (0, 1)$, consider the error probability as risk function and assume:

1. There exists a function $f : \mathcal{X} \to \mathcal{Y}$ that determines the labels, i.e., $\mathbb{P}[Y = y|X = x] = \delta_{y,f(x)}$ holds $\forall (x, y) \in \mathcal{X} \times \mathcal{Y}$.
2. For any $S = \big((x_i, f(x_i))\big)_{i=1}^n \in (\mathcal{X} \times \mathcal{Y})^n$ the considered learning algorithm returns a hypothesis $h_S \in \mathcal{F}$ for which $\hat{R}(h_S) = 0$.

Then $\mathbb{P}_S\big[R(h_S) > \epsilon\big] \leq |\mathcal{F}|(1 - \epsilon)^n$ and for any $\delta > 0$, $n \in \mathbb{N}$ one has

$$n \geq \frac{1}{\epsilon} \ln \frac{|\mathcal{F}|}{\delta} \quad \Rightarrow \quad \mathbb{P}_S\left[|\hat{R}(h_S) - R(h_S)| > \epsilon\right] \leq \delta.$$

Proof. We assume that $|\mathcal{F}| < \infty$ since the statements are trivial or empty otherwise. First observe that for any hypothesis

$$\begin{aligned} \mathbb{P}_S\big[\hat{R}(h) = 0\big] &= \mathbb{P}_S\big[\forall i \in \{1, \dots, n\} : h(X_i) = f(X_i)\big] \\ &= \prod_{i=1}^n \mathbb{P}\big[h(X_i) = f(X_i)\big] \;=\; \big(1 - \mathbb{P}[h(X) \neq f(X)]\big)^n \\ &= \big(1 - R(h)\big)^n, \end{aligned} \tag{1.14}$$

where the i.i.d. assumption is used in the second line. With $\hat{R}(h_S) = 0 \wedge h_S \in \mathcal{F}$ we can bound

$$\begin{aligned} \mathbb{P}_S\left[|\hat{R}(h_S) - R(h_S)| > \epsilon\right] &= \mathbb{P}_S\left[R(h_S) > \epsilon\right] \\ &\leq \mathbb{P}_S\left[\exists h \in \mathcal{F} : \hat{R}(h) = 0 \wedge R(h) > \epsilon\right] \\ &\leq \sum_{h \in \mathcal{F} : R(h) > \epsilon} \mathbb{P}_S\left[\hat{R}(h) = 0\right] \;\leq\; |\mathcal{F}|(1 - \epsilon)^n, \end{aligned}$$

where in the third line we first used the union bound, which can be applied since $|\mathcal{F}| < \infty$, and then we exploited Eq.(1.14). In addition, the number of terms in the sum $\sum_{h\in\mathcal{F}:R(h)>\epsilon}$ was simply bounded by $|\mathcal{F}|$. From here the final implication stated in the theorem can be obtained via $|\mathcal{F}|(1-\epsilon)^n \leq |\mathcal{F}|e^{-\epsilon n} =: \delta$ and solving for n. □

In the following we will relax the assumptions that were made in Thm.1.4 more and more. Many of the PAC learning bounds then rely on the following Lemma, which was proven in [14]:

Lemma 1.1 (Hoeffding's inequality)**.** *Let $Z_1, \ldots, Z_n$ be real independent random variables whose values are contained in intervals $[a_i, b_i] \supseteq \mathrm{range}[Z_i]$. Then for every $\epsilon > 0$ it holds that*

$$\mathbb{P}\left[\sum_{i=1}^{n} Z_i - \mathbb{E}\left[Z_i\right] \geq \epsilon\right] \leq \exp\left[-\frac{2\epsilon^2}{\sum_{i=1}^{n}(a_i - b_i)^2}\right]. \qquad (1.15)$$

A useful variant of this inequality can be obtained as a simple corollary: the probability $\mathbb{P}\left[\left|\sum_{i=1}^{n} Z_i - \mathbb{E}\left[Z_i\right]\right| \geq \epsilon\right]$ can be bounded by two times the r.h.s. of Eq.(1.15). This can be seen by first observing that Eq.(1.15) remains valid when replacing Z_i by $-Z_i$ and then adding the obtained inequality to the initial one.

We will now use Hoeffding's inequality to prove a PAC learning bound without assuming that there is a deterministic assignments of labels that can be perfectly described within $\mathcal{F}$. In the literature, this scenario is often called the *agnostic* case (as opposed to the *realizable* case considered in the previous theorem).

Theorem 1.5: PAC bound for countable, weighted hypotheses

Consider a countable hypothesis class $\mathcal{F}$ and a loss function whose values are contained in an interval of length $c \geq 0$. Let p be any probability distribution over $\mathcal{F}$ and $\delta \in (0, 1]$ any confidence parameter. Then with probability at least $(1-\delta)$ w.r.t. repeated sampling of sets of training data of size n we have

$$\forall h \in \mathcal{F} : \left|\hat{R}(h) - R(h)\right| \leq c\sqrt{\frac{\ln\frac{1}{p(h)} + \ln\frac{2}{\delta}}{2n}}. \qquad (1.16)$$

Note: The bound is again independent of the underlying probability measure P. It should also be noted that $p(h)$ can not depend on the training data and is merely used in order to allow the level of approximation to depend on the hypothesis. In particular, $p(h)$ can not be interpreted as a probability with which the hypothesis h is chosen.

Proof. Let us first consider a fixed $h \in \mathcal{F}$ and apply Hoeffding's inequality to the i.i.d. random variables $Z_i := L(Y_i, h(X_i))/n$. Setting $\epsilon := c\sqrt{\left(\ln\frac{2}{p(h)\delta}\right)/2n}$

we obtain

$$\mathbb{P}_S\left[\left|\hat{R}(h)-R(h)\right|\geq\epsilon\right]\leq p(h)\delta. \tag{1.17}$$

In order to bound the probability that for any of the h's the empirical average deviates from the mean, we exploit the union bound and arrive at

$$\mathbb{P}_S\left[\exists h\in\mathcal{F}:\left|\hat{R}(h)-R(h)\right|\geq\epsilon\right]\leq\sum_{h\in\mathcal{F}}\mathbb{P}_S\left[\left|\hat{R}(h)-R(h)\right|\geq\epsilon\right]\leq\sum_{h\in\mathcal{F}}p(h)\delta=\delta.$$

□

The ϵ in Eq.(1.17) depends on the hypothesis h. The smaller the weight $p(h)$, the larger the corresponding ϵ. Hence, effectively, the above derivation provides reasonable bounds only for a finite number of hypotheses. If $\mathcal{F}$ itself is finite, we can choose $p(h):=1/|\mathcal{F}|$ and rewrite the theorem so that it yields a bound for the size of the training set that is sufficient for a PAC learning guarantee:

Corollary 1.2. *Consider a finite hypothesis space $\mathcal{F}$, $\delta\in(0,1]$, $\epsilon>0$ and a loss function whose range is contained in an interval of length $c\geq 0$. Then $\forall h\in\mathcal{F}:|\hat{R}(h)-R(h)|\leq\epsilon$ holds with probability at least $1-\delta$ over repeated sampling of training sets of size n, if*

$$n\ \geq\ \frac{c^2}{2\epsilon^2}\left(\ln|\mathcal{F}|+\ln\frac{2}{\delta}\right). \tag{1.18}$$

Due to Eq.(1.7) this also guarantees that $R(\hat{h})-R_{\mathcal{F}}\leq 2\epsilon$, providing a quantitative justification of ERM. Consequently, in a deterministic scenario where a function $f:\mathcal{X}\to\mathcal{Y}$ determines the 'true' label, we have $R(\hat{h})\leq 2\epsilon$, if $f\in\mathcal{F}$.

Unfortunately, for infinite $\mathcal{F}$ the statement of the corollary becomes void — a drawback that will to a large extent be corrected in the following sections. Before going there, however, we will discuss why it is nevertheless essential to restrict the hypotheses class $\mathcal{F}$.

If $\mathcal{F}=\mathcal{Y}^{\mathcal{X}}$ with all sets finite, then Eq.(1.18) provides a PAC guarantee essentially only if n exceeds $|\mathcal{X}|$. The latter means, however, that the learning algorithm has basically already seen all instances in the training data. The next theorem shows that this is indeed necessary for PAC learning if $\mathcal{F}=\mathcal{Y}^{\mathcal{X}}$, i.e., if the hypotheses class is not constrained.

1.4 No free lunch

If we are given part of a sequence, say 2 4 8 16, without further assumption about an underlying structure, we can not infer the next number. As Hume phrased it (first published anonymously in 1739): *there is nothing in any object, consider'd in itself, which can afford us a reason for drawing a conclusion beyond it.* The necessity of prior information in machine learning is put in a nutshell by the 'no-free-lunch theorem', of which one version is the following:

Theorem 1.6: No-free-lunch

Let $\mathcal{X}$ and $\mathcal{Y}$ both be finite and so that $|\mathcal{X}|$ exceeds the size n of the training set S. For any $f : \mathcal{X} \to \mathcal{Y}$ define $R_f(h) := \mathbb{P}\left[h(X) \neq f(X)\right]$ where the probability is taken w.r.t to a uniform distribution of X over $\mathcal{X}$. Then for every learning algorithm the expected risk averaged uniformly over all functions $f \in \mathcal{Y}^{\mathcal{X}}$ fulfills

$$\mathbb{E}_f\left[\mathbb{E}_S\left[R_f\big(h_S\big)\right]\right] \geq \left(1 - \frac{1}{|\mathcal{Y}|}\right)\left(1 - \frac{n}{|\mathcal{X}|}\right). \tag{1.19}$$

Note: Here it is understood that f determines the joint distribution $P(x,y) = \delta_{y,f(x)}/|\mathcal{X}|$. Consequently, the training data has the form $\big((x_i, f(x_i))\big)_{i=1}^n$.

Proof. Denote by $\mathcal{X}_S$ the subset of $\mathcal{X}$ appearing in the training data S. If necessary, add further elements to $\mathcal{X}_S$ until $|\mathcal{X}_S| = n$. We can write

$$\begin{aligned}\mathbb{E}_f\left[\mathbb{E}_S\left[R_f\big(h_S\big)\right]\right] &= \frac{1}{|\mathcal{X}|}\mathbb{E}_f\left[\mathbb{E}_S\left[\sum_{x\in\mathcal{X}} \mathbb{1}_{h_S(x)\neq f(x)}\right]\right] &(1.20)\\ &\geq \frac{1}{|\mathcal{X}|}\mathbb{E}_f\left[\mathbb{E}_S\left[\sum_{x\notin\mathcal{X}_S} \mathbb{1}_{h_S(x)\neq f(x)}\right]\right]. &(1.21)\end{aligned}$$

While inside $\mathcal{X}_S$ the value of $f(x)$ is determined by S, for $x \notin \mathcal{X}_S$ all $|\mathcal{Y}|$ values are possible and equally likely, so that $h_S(x) \neq f(x)$ holds with probability $1 - 1/|\mathcal{Y}|$ w.r.t. a uniform distribution over f's that are consistent with S. The remaining factor is due to $\sum_{x\notin\mathcal{X}_S} = |\mathcal{X}| - n$. □

Let us compare this with random guessing. The risk, i.e., the average error probability, of random guessing in the above scenario is $1 - 1/|\mathcal{Y}|$. Thm.1.6 only leaves little room for improvement beyond this—an additional factor $(1-n/|\mathcal{X}|)$. This factor reflects the fact that the learning algorithm has already seen the training data, which is at most a fraction $n/|\mathcal{X}|$ of all cases. Regarding the unseen cases, however, all learning algorithms are the same on average and perform no better than random guessing. Note that the above proof also allows us to derive an upper bound in addition to the lower bound in Eq.(1.19). To this end, observe that the difference between Eqs.(1.20) and (1.21) is at most $n/|\mathcal{X}|$. Hence, in the limit $n/|\mathcal{X}| \to 0$ the average error probability is exactly the one for random guessing, irrespective of what learning algorithm has been chosen.

This sobering result also implies that there is no order among learning algorithms. If one learning algorithm beats another on some functions, the converse has to hold on other functions. This result, as well as similar ones, has to be put into perspective, however, since not all functions are equally relevant.

The no-free-lunch theorem should not come as a surprise. In fact, it is little more than a formalization of a rather obvious claim within our framework: if one is given n values of a sequence of independently, identically and uniformly distributed random variables, then predicting the value of the $(n+1)$'st can not be better than random guessing. If prediction is to be better than chance, then additional structure is required. The inevitable a priori information about this structure can be incorporated into machine learning in different ways. In the approach we focus on in this course, the a priori information is reflected in the choice of the hypotheses class $\mathcal{F}$. In addition, hypotheses in $\mathcal{F}$ may effectively be given different weight, for instance resulting from SRM, regularization or a Bayesian prior distribution over $\mathcal{F}$. At the same time, this approach is *distribution-independent* in the sense that it makes no assumption about the distribution P that governs the data. An alternative approach (which we will not follow) would be to put prior information into P, for instance by assuming a parametric model for P.

1.5 Growth function

Starting in this section, we aim at generalizing the PAC bounds derived in Sec.1.3 to beyond finite hypotheses classes. The first approach we will discuss essentially replaces the cardinality of $\mathcal{F}$ by the corresponding *growth function*.

Definition 1.3 (Growth function)**.** *Let $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a class of functions with finite target space $\mathcal{Y}$. For every subset $\Xi \subseteq \mathcal{X}$ define the restriction of $\mathcal{F}$ to Ξ as $\mathcal{F}|_\Xi := \{f \in \mathcal{Y}^\Xi \mid \exists F \in \mathcal{F}\ \forall x \in \Xi : f(x) = F(x)\}$. The* growth function Γ *assigned to $\mathcal{F}$ is then defined for all $n \in \mathbb{N}$ as*

$$\Gamma(n) := \max_{\Xi \subseteq \mathcal{X} : |\Xi| = n} \left| \, \mathcal{F}|_\Xi \, \right|.$$

For later convenience, we will in addition set $\Gamma(0) := 1$.

That is, the growth function describes the maximal size of $\mathcal{F}$ when restricted to a domain of n points. Thus, by definition $\Gamma(n) \leq |\mathcal{Y}|^n$.

Example 1.3 (Threshold functions). Consider the set of all threshold functions $\mathcal{F} \subseteq \{-1,1\}^{\mathbb{R}}$ defined by $\mathcal{F} := \{x \mapsto \operatorname{sgn}[x-b]\}_{b \in \mathbb{R}}$. Given a set of distinct points $\{x_1, \ldots, x_n\} = \Xi \subseteq \mathbb{R}$, there are $n+1$ functions in $\mathcal{F}|_\Xi$ corresponding to $n+1$ possible ways of placing b relative to the x_i's. Hence, in this case $\Gamma(n) = n+1$.

Theorem 1.7: PAC bound via growth function

Consider a hypothesis class $\mathcal{F}$ with finite target space $\mathcal{Y}$ and a loss function whose range is contained in an interval $[0,c]$. Let $\delta \in (0,1]$. With probability at least $(1-\delta)$ w.r.t. repeated sampling of training data of size n we have

$$\forall h \in \mathcal{F}: \; |R(h) - \hat{R}(h)| \leq c \sqrt{\frac{8 \ln\left(\Gamma(2n)\frac{4}{\delta}\right)}{n}}. \tag{1.22}$$

Note: this implies a non-trivial bound whenever the growth function grows subexponentially. Loosely speaking, in this case every additional data point adds information about the risk $R(h)$. This is in particular true if $|\mathcal{F}| < \infty$, since $\forall n : \Gamma(n) \leq |\mathcal{F}|$, but it does not require a finite hypotheses class as already seen in example 1.3.

Proof. Let S and S' be i.i.d. random variables with values in $(\mathcal{X} \times \mathcal{Y})^n$ distributed according to some product probability measure P^n. For every value of S' denote by $\hat{R}'(h)$ the corresponding empirical risk of a hypothesis $h \in \mathcal{F}$. By virtue of the triangle inequality, if $|R(h) - \hat{R}(h)| > \epsilon$ and $|R(h) - \hat{R}'(h)| < \frac{\epsilon}{2}$, then $|\hat{R}'(h) - \hat{R}(h)| > \frac{\epsilon}{2}$. Expressed in terms of indicator functions this is

$$\mathbb{1}_{|R(h)-\hat{R}(h)|>\epsilon} \, \mathbb{1}_{|R(h)-\hat{R}'(h)|<\frac{\epsilon}{2}} \;\leq\; \mathbb{1}_{|\hat{R}'(h)-\hat{R}(h)|>\frac{\epsilon}{2}}. \tag{1.23}$$

Let us assume that $n \geq 4c^2\epsilon^{-2}\ln 2$, which will be justified later by a particular choice of ϵ. Taking the expectation value w.r.t. S' in Eq.(1.23) affects the second and third term. The former can be bounded using Hoeffding's inequality together with the assumption on n, which leads to

$$\mathbb{E}_{S'}\left[\mathbb{1}_{|R(h)-\hat{R}'(h)|<\frac{\epsilon}{2}}\right] \geq 1 - 2\exp\left[-\frac{\epsilon^2 n}{2c^2}\right] \geq \frac{1}{2}.$$

For the expectation value of the last term in Eq.(1.23) we use

$$\mathbb{E}_{S'}\left[\mathbb{1}_{|\hat{R}'(h)-\hat{R}(h)|>\frac{\epsilon}{2}}\right] \leq \mathbb{P}_{S'}\left[\exists h \in \mathcal{F} : |\hat{R}'(h) - \hat{R}(h)| > \frac{\epsilon}{2}\right].$$

Inserting both bounds into Eq.(1.23) gives

$$\mathbb{1}_{|R(h)-\hat{R}(h)|>\epsilon} \;\leq 2\,\mathbb{P}_{S'}\left[\exists h \in \mathcal{F} : |\hat{R}'(h) - \hat{R}(h)| > \frac{\epsilon}{2}\right].$$

As this holds for all $h \in \mathcal{F}$, we can replace the left hand side by $\mathbb{1}_{\exists h\in\mathcal{F}:|R(h)-\hat{R}(h)|>\epsilon}$. Taking the expectation w.r.t. S then leads to

$$\mathbb{P}_S\left[\exists h \in \mathcal{F} : |R(h) - \hat{R}(h)| > \epsilon\right] \leq 2\mathbb{P}_{S,S'}\left[\exists h \in \mathcal{F} : |\hat{R}'(h) - \hat{R}(h)| > \frac{\epsilon}{2}\right].$$

Note that the r.h.s. involves only empirical quantities. This implies that every function h is only evaluated on at most $2n$ points, namely those appearing in S and S'. Since restricted to $2n$ points there are at most $\Gamma(2n)$ functions, our aim is now to exploit this together with the union bound and to bound the remaining factor with Hoeffding's inequality. To this end, observe that we can write

$$
\begin{aligned}
&2\mathbb{P}_{S,S'}\left[\exists h\in\mathcal{F}:|\hat{R}'(h)-\hat{R}(h)|>\frac{\epsilon}{2}\right] \;= \qquad\qquad (1.24)\\
&2\mathbb{E}_{SS'}\left[\mathbb{P}_\sigma\left[\exists h\in\mathcal{F}\;:\;\frac{1}{n}\Big|\sum_{i=1}^n\Big(L\big(Y_i,h(X_i)\big)-L\big(Y_i',h(X_i')\big)\Big)\sigma_i\Big|>\frac{\epsilon}{2}\right]\right],
\end{aligned}
$$

where $\mathbb{P}_\sigma$ denotes the probability w.r.t. uniformly distributed $\sigma\in\{-1,1\}^n$. Eq.(1.24) is based on the fact that multiplication with $\sigma_i=-1$ amounts to interchanging $(X_i,Y_i)\leftrightarrow(X_i',Y_i')$, which has no effect since the random variables are independently and identically distributed. The advantage of this step is that we can now apply our tools inside the expectation value $\mathbb{E}_{SS'}$ where S and S' are fixed. Then $h\in\mathcal{F}\big|_{S\cup S'}$ is contained in a finite function class, so that we can use the union bound followed by an application of Hoeffding's inequality to arrive at

$$
\begin{aligned}
\mathbb{P}_S\left[\exists h\in\mathcal{F}:|R(h)-\hat{R}(h)|>\epsilon\right] &\leq 4\mathbb{E}_{SS'}\left[\big|\mathcal{F}|_{S\cup S'}\big|\right]\exp\left[-\frac{n\epsilon^2}{8c^2}\right]\\
&\leq 4\Gamma(2n)\exp\left[-\frac{n\epsilon^2}{8c^2}\right] \qquad (1.25)
\end{aligned}
$$

The result then follows by setting the final expression in Eq.(1.25) equal to δ and solving for ϵ. The previously made assumption on n then becomes equivalent to $\delta\leq 2\sqrt{2}\Gamma(2n)$, which is always fulfilled as $\delta\in(0,1]$. □

Note that we have proven a slightly stronger result, in which the growth function $\Gamma(2n)$ is replaced by $\mathbb{E}_{SS'}\left[\big|\mathcal{F}|_{S\cup S'}\big|\right]$. The logarithm of this expectation value is called *VC-entropy*. The VC-entropy, however, depends on the underlying probability distribution P and is thus difficult to estimate in general. The growth function, though independent of P, is still difficult to estimate. The following section will distill its remarkable essence for the binary case ($|\mathcal{Y}|=2$), where Γ turns out to exhibit a simple dichotomic behavior.

For later use, let us state the behavior of the growth function w.r.t. compositions:

Lemma 1.4 (Growth functions under compositions)**.** *Consider function classes $\mathcal{F}_1\subseteq\mathcal{Y}^{\mathcal{X}},\mathcal{F}_2\subseteq\mathcal{Z}^{\mathcal{Y}}$ and $\mathcal{F}:=\mathcal{F}_2\circ\mathcal{F}_1$. The respective growth functions then satisfy*

$$\Gamma(n)\leq\Gamma_1(n)\Gamma_2(n).$$

Proof. Fix an arbitrary subset $\Xi\subseteq\mathcal{X}$ of cardinality $|\Xi|=n$. With $\mathcal{G}:=\mathcal{F}_1|_\Xi$ we can write $\mathcal{F}|_\Xi=\bigcup_{g\in\mathcal{G}}\{f\circ g\mid f\in\mathcal{F}_2\}$. So

$$
\begin{aligned}
\big|\mathcal{F}|_\Xi\big| &\leq \big|\mathcal{F}_1|_\Xi\big|\max_{g\in\mathcal{G}}\big|\{f\circ g|f\in\mathcal{F}_2\}\big|\\
&\leq \Gamma_1(n)\max_{g\in\mathcal{G}}\big|\mathcal{F}_2|_{g(\Xi)}\big|\\
&\leq \Gamma_1(n)\Gamma_2(n).
\end{aligned}
$$

□

1.6 VC-dimension

In the case of binary target space ($|\mathcal{Y}| = 2$) there is a peculiar dichotomy in the behavior of the growth function $\Gamma(n)$. It grows at maximal rate, i.e., exponentially and exactly like 2^n, up to some $n = d$ and from then on remains bounded by a polynomial of degree at most d. The number d where this transition occurs, is called the *VC-dimension* of the function class and plays an important role in the theory of binary classification.

Definition 1.5 (Vapnik-Chervonenkis dimension)**.** *The* VC-dimension *of a function class $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ with binary target space $\mathcal{Y}$ is defined as*

$$\mathrm{VCdim}(\mathcal{F}) := \max \left\{ n \in \mathbb{N}_0 \;\middle|\; \Gamma(n) = 2^n \right\}$$

if the maximum exists and $\mathrm{VCdim}(\mathcal{F}) = \infty$ *otherwise.*

That is, if $\mathrm{VCdim}(\mathcal{F}) = d$, then there exists a set $A \subseteq \mathcal{X}$ of d points, such that $\mathcal{F}|_A = \mathcal{Y}^A$ and the VC-dimension is the largest such number.

Example 1.4 (Threshold functions)*.* If $\mathcal{F} = \{\mathbb{R} \ni x \mapsto \mathrm{sgn}[x - b]\}_{b \in \mathbb{R}}$, then $\mathrm{VCdim}(\mathcal{F}) = 1$ as we have seen in example 1.3 that $\Gamma(n) = n + 1$. More specifically, if we consider an arbitrary pair of points $x_1 < x_2$, then the assignment $x_1 \mapsto 1 \wedge x_2 \mapsto -1$ is missing in $\mathcal{F}|_{\{x_1, x_2\}}$. Hence, $\mathrm{VCdim}(\mathcal{F}) < 2$.

Theorem 1.8: VC-dichotomy of growth function

Consider a function class $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ with binary target space $\mathcal{Y}$ and VC-dimension d. Then the corresponding growth function satisfies

$$\Gamma(n) \begin{cases} = \; 2^n, & \text{if } n \leq d. \\ \leq \; \left(\frac{en}{d}\right)^d, & \text{if } n > d. \end{cases} \tag{1.26}$$

Proof. $\Gamma(n) = 2^n$ for all $n \leq d$ holds by definition of the VC-dimension. In order to arrive at the expression for $n > d$, we show that for every subset $A \subseteq \mathcal{X}$ with $|A| = n$ the following is true:

$$\big|\mathcal{F}|_A\big| \leq \Big|\left\{ B \subseteq A \;\middle|\; \mathcal{F}|_B = \mathcal{Y}^B \right\}\Big|. \tag{1.27}$$

If Eq.(1.27) holds, we can upper bound the r.h.s. by $\big|\{B \subseteq A \big| \; |B| \leq d\}\big| = \sum_{i=0}^{d} \binom{n}{i}$, which for $n > d$ in turn can be bounded by

$$\begin{aligned} \sum_{i=0}^{d} \binom{n}{i} \;&\leq\; \sum_{i=0}^{n} \binom{n}{i} \left(\frac{n}{d}\right)^{d-i} \\ &=\; \left(\frac{n}{d}\right)^{d} \left(1 + \frac{d}{n}\right)^{n} \;\leq\; \left(\frac{n}{d}\right)^{d} e^{d}, \end{aligned}$$

where the last step follows from $\forall x \in \mathbb{R} : (1+x) \leq e^x$. Hence, the proof is reduced to showing Eq.(1.27).

This will be done by induction on $|A|$. For $|A| = 1$ it is true (as $B = \emptyset$ always counts). Now assume as induction hypothesis that it holds for all sets of size $n-1$ and that $|A| = n$. Let a be any element of A and define

$$\mathcal{F}' := \big\{h \in \mathcal{F}|_A \;\big|\; \exists g \in \mathcal{F}|_A : h(a) \neq g(a) \wedge (h-g)|_{A\setminus a} = 0\big\}, \quad \mathcal{F}_a := \mathcal{F}'|_{A\setminus a}.$$

Then $|\mathcal{F}|_A| = |\mathcal{F}|_{A\setminus a}| + |\mathcal{F}_a|$ and both terms on the r.h.s. can be bounded by the induction hypothesis. For the first term we obtain

$$\big|\mathcal{F}|_{A\setminus a}\big| \leq \Big|\big\{B \subseteq A \;\big|\; \mathcal{F}|_B = \mathcal{Y}^B \wedge a \notin B\big\}\Big|. \tag{1.28}$$

The second term can be bounded by

$$\begin{aligned} \big|\mathcal{F}_a\big| &= \big|\mathcal{F}'|_{A\setminus a}\big| \;\leq\; \Big|\big\{B \subseteq A \setminus a \;\big|\; \mathcal{F}'|_B = \mathcal{Y}^B\big\}\Big| \\ &= \Big|\big\{B \subseteq A \setminus a \;\big|\; \mathcal{F}'|_{B\cup a} = \mathcal{Y}^{B\cup a}\big\}\Big| \\ &= \Big|\big\{B \subseteq A \;\big|\; \mathcal{F}'|_B = \mathcal{Y}^B \wedge a \in B\big\}\Big| \\ &\leq \Big|\big\{B \subseteq A \;\big|\; \mathcal{F}|_B = \mathcal{Y}^B \wedge a \in B\big\}\Big|, \end{aligned} \tag{1.29}$$

where we use the induction hypothesis in the first line and the step to the second line uses the defining property of $\mathcal{F}'$. Adding the bounds of Eq.(1.28) and Eq.(1.29) then yields the result claimed in Eq.(1.27). □

Now we can plug this bound on the growth function into the PAC bound in Thm.1.7. After a couple of elementary manipulations we then arrive at the following result, which, similar to Cor.1.2, provides a bound on the necessary statistics, but with the VC-dimension d now playing the role of $\ln|\mathcal{F}|$.

Corollary 1.6. *Consider a function class $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ with binary target space and VC-dimension d. Let $(\epsilon, \delta) \in (0,1]^2$ and choose the risk function R to be the error probability. Then $\forall h \in \mathcal{F} : |\hat{R}(h) - R(h)| \leq \epsilon$ holds with probability at least $1-\delta$ over repeated sampling of training sets of size n, if*

$$n \;\geq\; \frac{32}{\epsilon^2}\left[d \ln\left(\frac{8d}{\epsilon^2}\right) + \ln\frac{6}{\delta}\right]. \tag{1.30}$$

Note: the bound in Eq.(1.30) can be slightly improved. In particular, the first logarithm turns out to be unnecessary, cf. Eq.(1.41).

A useful tool for computing VC-dimensions is the following theorem:

Theorem 1.9: VC-dimension for function vector spaces

Let $\mathcal{G}$ be a real vector space of functions from $\mathcal{X}$ to $\mathbb{R}$. Then $\mathcal{F} := \big\{x \mapsto \operatorname{sgn}[g(x)]\big\}_{g\in\mathcal{G}} \subseteq \{-1,1\}^{\mathcal{X}}$ has $\operatorname{VCdim}(\mathcal{F}) = \dim(\mathcal{G})$.

Proof. Let us first prove $\operatorname{VCdim}(\mathcal{F}) \leq \dim(\mathcal{G})$. We can assume $\dim(\mathcal{G}) < \infty$ and argue by contradiction. Let $k = \dim(\mathcal{G})+1$ and suppose that $\operatorname{VCdim}(\mathcal{F}) \geq k$. Then there is a subset $\Xi = \{x_1,\dots,x_k\} \subseteq \mathcal{X}$ such that $\mathcal{F}|_\Xi = \{-1,1\}^\Xi$. Define a map $L : \mathcal{G} \to \mathbb{R}^k$ via $L(g) := \big(g(x_1),\dots,g(x_k)\big)$. L is a linear map whose range has dimension at most $\dim(\mathcal{G})$. Hence, there is a non-zero vector $v \in (\text{range } L)^\perp = \text{ker L}^*$. This means that for all $g \in \mathcal{G}$:

$$0 = \langle L^*(v), g\rangle = \langle v, L(g)\rangle = \sum_{l=1}^{k} v_l \; g(x_l). \tag{1.31}$$

However, if $\mathcal{F}|_\Xi = \{-1,1\}^\Xi$, we can choose g such that $\operatorname{sgn}[g(x_l)] = \operatorname{sgn}[v_l]$ for all $l \in \{1,\dots,k\}$, which would imply $\sum_{l=1}^k v_l \; g(x_l) > 0$.

In order to arrive at $\operatorname{VCdim}(\mathcal{F}) \geq \dim(\mathcal{G})$, it suffices to show that for all $d \leq \dim(\mathcal{G})$ there are points $x_1,\dots,x_d \in \mathcal{X}$ such that for all $y \in \mathbb{R}^d$ there is a $g \in \mathcal{G}$ satisfying $y_j = g(x_j)$ for all j. To this end, consider d linearly independent functions $(g_i)_{i=1}^d$ in $\mathcal{G}$ and define $G(x) := \big(g_1(x),\dots,g_d(x)\big)$. Then $\operatorname{span}\{G(x)\}_{x\in\mathcal{X}} = \mathbb{R}^d$ so that there have to exist d linearly independent vectors $G(x_1),\dots,G(x_d)$. Hence, the $d\times d$ matrix with entries $g_i(x_j)$ is invertible and for all $y \in \mathbb{R}^d$ the system of equations $y_j = \sum_{i=1}^d \gamma_i g_i(x_j)$ has a solution $\gamma \in \mathbb{R}^d$. □

Corollary 1.7 (VC-dimension of half spaces)**.** *The set* $\mathcal{F} := \big\{h : \mathbb{R}^d \to \{-1,1\} \mid \exists (v,b) \in \mathbb{R}^d \times \mathbb{R} : \;\; h(x) = \operatorname{sgn}[\langle v,x\rangle - b]\big\}$, *which corresponds to the set of all half spaces in* $\mathbb{R}^d$, *satisfies*

$$\operatorname{VCdim}(\mathcal{F}) = d+1.$$

Proof. The result follows from the foregoing theorem, when applied to the linear space of functions spanned by $g_i(x) := x_i$ for $i = 1,\dots,d$ and $g_{d+1}(x) := 1$. □

As in the case of half spaces, we can assign a function $f : \mathbb{R}^d \to \{-1,1\}$ to any subset $C \subseteq \mathbb{R}^d$ and vice versa via $f(x) = 1 \Leftrightarrow x \in C$. In this way we can apply the notion of VC-dimension to classes of Borel subsets of $\mathbb{R}^d$. Table 1.2 collects some examples.

Example 1.5 (Axes-aligned rectangles). Consider $\mathcal{C} := \{C \subseteq \mathbb{R}^d | \exists a, b \in \mathbb{R}^d : C = [a_1,b_1] \times \ldots \times [a_d,b_d]\}$ the set of all axes-aligned rectangles in $\mathbb{R}^d$ and let $\mathcal{F} := \{f : \mathbb{R}^d \to \{0,1\} | \exists C \in \mathcal{C} : f(x) = \mathbb{1}_{x\in\mathcal{C}}\}$ be the corresponding class of indicator-functions. For any set of points $A = \{x_1,\dots,x_n\} \subseteq \mathbb{R}^d$ there is a unique smallest rectangle $C_{min} \in \mathcal{C}$ so that $A \subseteq C_{min}$. As long as $n > 2d$

	$\mathcal{X}$	VCdim	see
l_2-balls	$\mathbb{R}^d$	$d+1$	[11]
l_∞-balls	$\mathbb{R}^d$	$\lfloor (3d+1)/2 \rfloor$	[10]
half spaces	$\mathbb{R}^d$	$d+1$	Cor.1.7
axes-aligned rectangles	$\mathbb{R}^d$	$2d$	Exp.1.5
convex k-gons	$\mathbb{R}^2$	$2k+1$	[7]
semialgebraic sets $\mathcal{S}_{k,m}$	$\mathbb{R}^d$	$\leq 2k\binom{m+d}{m}\ln\big((k^2+k)\binom{m+d}{m}\big)$	[6]
$\mathcal{S}_{1,m}$	$\mathbb{R}^d$	$\binom{m+d}{m}$	[6]
$\mathrm{Aff}(C)$ for fixed $C \in \mathcal{S}_{k,m}$	$\mathbb{R}^d$	$\mathcal{O}\big(d^2 \ln(dkm)\big)$	[6]
$\{x \mapsto \operatorname{sgn}\sin[\alpha x] \mid \alpha \in \mathbb{R}\}$	$\mathbb{R}$	∞	Exp.1.6

Figure 1.2: VC-dimension of various classes of functions or corresponding geometric objects. A convex k-gon means a polygon in $\mathbb{R}^2$ that is obtained by intersecting k half spaces. $\mathcal{S}_{k,m}$ is the class of subsets of $\mathbb{R}^d$ that can be obtained as Boolean combination of k sets of the form $f_j^{-1}\big((0,\infty)\big)$ where each $f_j : \mathbb{R}^d \mapsto \mathbb{R}$, $j = 1, \ldots, k$ is a polynomial of maximal degree m. $\mathrm{Aff}(C)$ denotes the class of all affine transformations of C.

we can discard points from A without changing C_{min}. Let $\tilde{A} \subseteq A$ be such a reduced set with $|\tilde{A}| \leq 2d$. Then every $f \in \mathcal{F}|_A$ that assigns a value 1 to all elements of $\tilde{A}$ also assigns 1 to all $A \setminus \tilde{A}$, since those lie inside the same box. Hence, if $n > 2d$, then the function $\tilde{f}(x) := \mathbb{1}_{x \in \tilde{A}}$ is not contained in $\mathcal{F}|_A$ and therefore $\mathrm{VCdim}(\mathcal{F}) \leq 2d$.

To prove equality, consider the extreme points of the d-dimensional hyperoctahedron (i.e., the l_1-unit ball), which are given by all the permutations of $(\pm 1, 0, \ldots, 0)$. Denote them by $x_k^{(+)}$ and $x_k^{(-)}$, $k = 1, \ldots, d$, depending on whether the k'th component is $+1$ or -1. Let f be an arbitrary assignment of values 0 or 1 to these $2d$ points. Then

$$b_k := \frac{1}{2} + f\big(x_k^{(+)}\big) \quad \text{and} \quad a_k := -\frac{1}{2} - f\big(x_k^{(-)}\big)$$

define a rectangle $C \in \mathcal{C}$, which is such that $x_k^{(\pm)} \in C \Leftrightarrow f(x_k^{(\pm)}) = 1$. So, restricted to these $2d$ points, $\mathcal{F}$ still contains all functions and thus $\mathrm{VCdim}(\mathcal{F}) \geq 2d$.

In the examples discussed so far, the VC-dimension was essentially equal to the number of parameters that appear in the definition of the considered hypotheses class. That such a relation is not generally true is shown by the following example:

Example 1.6 (Sine-functions). Consider $\mathcal{F} := \{x \mapsto \operatorname{sgn}\sin(x\alpha) \mid \alpha \in \mathbb{R}\}$ and $A := \{2^{-k} | \ k = 1, \ldots, n\}$. Let f be an arbitrary assignment of values ± 1 to the points $x_k := 2^{-k}$ in A. If we choose

$$\begin{aligned}\alpha &:= \pi\left(1+\sum_{k=1}^{n}\frac{1-f(x_k)}{2}\,2^k\right), \quad \text{we obtain}\\ \alpha\, x_l \bmod 2\pi &= \pi\left(\frac{1-f(x_l)}{2}\right)+\pi\left[2^{-l}+\sum_{k=1}^{l-1}2^{k-l}\left(\frac{1-f(x_k)}{2}\right)\right]\\ &=: \pi\left(\frac{1-f(x_l)}{2}\right)+\pi\; c, \qquad (1.32)\end{aligned}$$

where $c \in (0,1)$. Consequently, $\operatorname{sgn}\sin(\alpha x_l) = f(x_l)$ and thus $\mathcal{F}|_A = \{-1,1\}^A$. Since this holds for all n, we have $\mathrm{VCdim}(\mathcal{F}) = \infty$ despite the fact that there is only a single real parameter involved.

Although the VC-dimension is infinite in this example, there are finite sets B for which $\mathcal{F}|_B \neq \{-1,1\}^B$. Consider for instance $B := \{1,2,3,4\}$ and the assignment $f(1) = f(2) = -f(3) = f(4) = -1$. If $\alpha = 2\pi m - \delta$, $m \in \mathbb{N}$ with $\delta \in [0, 2\pi)$ is to reproduce the first three values, then $\delta \in [\pi/3, \pi/2)$. However, this implies that 4δ is in the range where the sine is positive, so that $f(4) = -1$ cannot be matched.

Let us finally have a closer look at sets of functions from Euclidean space to $\{0,1\}$ that are constructed using Boolean combinations of few elementary, for instance polynomial, relations. In this context, it turns out that VC-dimension and growth function are related to the question how many connected components can be obtained when partitioning Euclidean space using these relations. Loosely speaking, counting functions becomes related to counting cells in the domain space. A central bound concerning the latter problem was derived by Warren for the case of polynomials:

Proposition 1.8 (Warren's upper bound for polynomial arrangements)**.** *Let $\{p_1,\ldots,p_m\}$ be a set of $m \geq k$ polynomials in k variables, each of degree at most d and with coefficients in $\mathbb{R}$. Let $\gamma(k,d,m)$ be the number of connected components of $\mathbb{R}^k \setminus \bigcup_{i=1}^m p^{-1}(\{0\})$ (and for later use, let us define it to be the largest number constructed in this way). Then*

$$\gamma(k,d,m) \leq (4edm/k)^k. \qquad (1.33)$$

With this ingredient, we can obtain the following result. To simplify its statement, predicates are interpreted as functions into $\{0,1\}$, i.e., we identify `TRUE` $= 1$ and `FALSE` $= 0$.

Theorem 1.10: Complexity of semi-algebraic function classes

Let $d,k,m,s \in \mathbb{N}$. Consider a set of s atomic predicates, each of which is given by a polynomial equality or inequality of degree at most d in $m+k$ variables. Let $\Psi : \mathbb{R}^m \times \mathbb{R}^k \to \{0,1\}$ be a Boolean combination of the atomic predicates and $\mathcal{F} := \{\Psi(\cdot,w) \mid w \in \mathbb{R}^k\}$ a class of functions from

$\mathbb{R}^m$ into $\{0,1\}$ with corresponding growth function Γ. Then

$$\Gamma(n) \quad \leq \quad \gamma(k,d,2ns)\ , \tag{1.34}$$

$$VCdim(\mathcal{F}) \quad \leq \quad 2k\log_2(8eds)\ . \tag{1.35}$$

Proof. W.l.o.g. we assume that all polynomial (in-)equalities are comparisons with zero, i.e., of the form $p\#0$ where p is a polynomial and $\# \in \{<,\leq,=,\geq,>\}$. We are going to estimate $|\mathcal{F}|_A|$ for a set $A \subset \mathbb{R}^m$ with cardinality $|A| = n$. Each $a \in A$ corresponds to a predicate $\psi_a : \mathbb{R}^k \to \{0,1\}$ that is defined by $\psi_a(w) := \Psi(a,w)$. Denote by P_a the set of polynomials (in the variables $w_1,\ldots,w_k$) that appear in ψ_a. Then $P := \bigcup_{a\in A} P_a$ has cardinality $|P| \leq ns$. Since different functions in $\mathcal{F}|_A$ correspond to different truth values of the polynomial (in-)equalities, we have that the number of consistent sign-assignments to the polynomials in P is an upper bound on the number of function in $\mathcal{F}|_A$. That is,

$$|\mathcal{F}|_A| \leq \Big|\big\{Q \in \{-1,0,1\}^P \;\big|\; Q(p) = \mathrm{sgn}_0\big(p(w)\big),\ w \in \mathbb{R}^k\big\}\Big|, \tag{1.36}$$

where $\mathrm{sgn}_0 := \mathrm{sgn}$ on $\mathbb{R}\setminus\{0\}$ and $\mathrm{sgn}_0(0) := 0$. For $\epsilon > 0$ define $P' := \{p+\epsilon | p \in P\} \cup \{p-\epsilon | p \in P\}$. Then $|P'| \leq 2|P| \leq 2ns$ and if ϵ is sufficiently small, the number of consistent sign-assignments for P is upper bounded by the number of connected components of $\mathbb{R}^k \setminus \bigcup_{p\in P'} p^{-1}(\{0\})$. Hence, $|\mathcal{F}|_A| \leq \gamma(k,d,|P'|)$, which implies Eq.(1.34).

The bound on the VC-dimension in Eq.(1.35) then combines this result with Prop.1.8. If n equals the VC-dimension of $\mathcal{F}$, then $2^n = \Gamma(n) \leq \gamma(k,d,2ns) \leq (8edns/k)^k$. Here, the last inequality used Prop.1.8 assuming that $2ns \geq k$. Note, however, that if $2ns < k$, then Eq.(1.35) holds trivially. After taking the $\log_2$, we arrive at the inequality

$$n \leq k\log_2(8eds) + k\log_2(n/k).$$

If the second term on the r.h.s. is smaller than the first, Eq.(1.35) follows immediately. If, on the other hand, $n/k \geq 8eds$, then $n \leq 2k\log_2(n/k)$, which in turn implies $n \leq 4k$ and Eq.(1.35) follows, as well. □

Note that the first part of the proof, which relates the growth function to the number of connected components of a particular partitioning of $\mathbb{R}^k$, made no essential use of the fact that the underlying functions are polynomials. That means, all one needs is a sufficiently well-behaved class of functions for which 'cell counting' can be done in the domain space.

An alternative view on the problem is in terms of the computational complexity of Ψ. By assumption, the function Ψ in Thm.1.10 can be computed using few elementary arithmetic operations and conditioning on (in-)equalities. The number of these operations is then related to d and s. A closer analysis of this point of view leads to:

Theorem 1.11: VC-dimension from computational complexity

Assume $\Psi : \mathbb{R}^m \times \mathbb{R}^k \to \{0,1\}$ can be computed by an algorithm that executes at most t of the following operations: (i) basic arithmetic operations $(\times, /, +, -)$ on real numbers, (ii) jumps conditioned on equality or inequality of real numbers, (iii) output 0 or 1. Then $\mathcal{F} := \{\Psi(\cdot, w) \mid w \in \mathbb{R}^k\}$ satisfies

$$VCdim(\mathcal{F}) \leq 2k\big(2t + \log_2 8e\big). \tag{1.37}$$

This follows from Thm.1.10 by realizing that the algorithm corresponds to an algebraic decision tree with at most 2^t leaves and that it can be expressed in terms of $\leq 2^t$ polynomial predicates of degree $\leq 2^t$.

With some effort one can add one further type to the list of operations the algorithm for Ψ is allowed to execute: computation of the exponential function on real numbers. Under these conditions the upper bound then becomes

$$VCdim(\mathcal{F}) = \mathcal{O}\big(t^2 k^2\big). \tag{1.38}$$

1.7 Fundamental theorem of binary classification

In this section we collect the insights obtained so far and use them to prove what may be called the *fundamental theorem of binary classification.* For its formulation, denote by $\text{poly}\big(\frac{1}{\epsilon}, \frac{1}{\delta}\big)$ the set of all function of the form $(0,1] \times (0,1] \ni (\epsilon, \delta) \mapsto \nu(\epsilon, \delta) \in \mathbb{R}_+$ that are polynomial in $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$.

Theorem 1.12: Fundamental theorem of binary classification

Let $\mathcal{F} \subseteq \{-1,1\}^{\mathcal{X}}$ be any hypotheses class and $n = |S|$ the size of the training data set S, which is treated as a random variable, distributed according to some product probability measure P^n. Choose the risk function R to be the error probability. Then the following are equivalent:

1. **(Finite VC-dimension)** $\text{VCdim}(\mathcal{F}) < \infty$.

2. **(Uniform convergence)** There is a $\nu \in \text{poly}\big(\frac{1}{\epsilon}, \frac{1}{\delta}\big)$ so that for all $(\epsilon, \delta) \in (0,1]^2$ and all probability measures P we have

$$n \geq \nu(\epsilon, \delta) \quad \Rightarrow \quad \mathbb{P}_S\left[\exists h \in \mathcal{F}: \ |\hat{R}(h) - R(h)| \geq \epsilon\right] \leq \delta.$$

3. **(PAC learnability)** There is a $\nu \in \text{poly}\big(\frac{1}{\epsilon}, \frac{1}{\delta}\big)$ and a learning algorithm that maps $S \mapsto h_S \in \mathcal{F}$ so that for all $(\epsilon, \delta) \in (0,1]^2$ and all probability measures P we have

$$n \geq \nu(\epsilon, \delta) \quad \Rightarrow \quad \mathbb{P}_S\left[|R(h_S) - R_{\mathcal{F}}| \geq \epsilon\right] \leq \delta. \tag{1.39}$$

4. **(PAC learnability via ERM)** There is a $\nu \in \text{poly}\big(\frac{1}{\epsilon}, \frac{1}{\delta}\big)$ so that for all $(\epsilon, \delta) \in (0,1]^2$ and all probability measures P we have

$$n \geq \nu(\epsilon, \delta) \quad \Rightarrow \quad \mathbb{P}_S\left[|R(\hat{h}) - R_{\mathcal{F}}| \geq \epsilon\right] \leq \delta,$$

where $\hat{h} \in \mathcal{F}$ is an arbitrary empirical risk minimizer.

Proof. 1. ⇒ 2. is the content of Cor.1.6.

2. ⇒ 4.: Assuming uniform convergence, with probability at least $(1-\delta)$ we have that $\forall h \in \mathcal{F}: \; |\hat{R}(h) - R(h)| \leq \frac{\epsilon}{2}$ if $n \geq \nu(\frac{\epsilon}{2}, \delta)$. By Eq.(1.7) this implies $R(\hat{h}) - R_{\mathcal{F}} \leq \epsilon$.

4. ⇒ 3. is obvious since the former is a particular instance of the latter.

3. ⇒ 1. is proven by contradiction: choose $\epsilon = \delta = 1/4$, $n = \nu(\epsilon, \delta)$ and suppose $\text{VCdim}(\mathcal{F}) = \infty$. Then for any $N \in \mathbb{N}$ there is a subset $\Xi \subseteq \mathcal{X}$ of size $|\Xi| = N$ such that $\mathcal{F}|_{\Xi} = \{-1,1\}^{\Xi}$. Applying the no-free-lunch theorem to this space we get that there is an $f : \Xi \to \{-1,1\}$, which defines a probability density $P(x,y) := \mathbb{1}_{x \in \Xi \,\wedge\, f(x)=y}/N$ on $\mathcal{X} \times \{-1,1\}$ with respect to which

$$\mathbb{E}_S\left[R(h_S)\right] \geq \frac{1}{2}\left(1 - \frac{n}{N}\right) \tag{1.40}$$

holds for an arbitrary learning algorithm, given by a mapping $S \mapsto h_S$. Using that $R(h_S)$ is itself a probability and thus bounded by one, we can bound

$$\mathbb{E}_S\left[R(h_S)\right] \leq 1 \cdot \mathbb{P}_S\left[R(h_S) \geq \epsilon\right] + \epsilon\big(1 - \mathbb{P}_S\left[R(h_S) \geq \epsilon\right]\big).$$

Together with Eq.(1.40) and $\epsilon = \frac{1}{4}$ this leads to $\mathbb{P}_S\left[R(h_S) \geq \frac{1}{4}\right] \geq \frac{1}{3} - \frac{2n}{3N}$, which for sufficiently large N contradicts $\delta = \frac{1}{4}$. □

There is also a quantitative version of this theorem. In fact, the VC-dimension does not only lead to a bound on the necessary statistics, it precisely specifies the optimal scaling of ν. Let us denote by $\nu_{\mathcal{F}}$ the pointwise infimum of all functions ν taken i) over all functions for which the implication in Eq.(1.39) is true for all P and all (ϵ, δ) and ii) over all learning algorithms with range $\mathcal{F}$. $\nu_{\mathcal{F}}$ is called the *sample complexity* of $\mathcal{F}$ and it can be shown that

$$\nu_{\mathcal{F}}(\epsilon, \delta) = \Theta\left(\frac{\text{VCdim}(\mathcal{F}) + \ln \frac{1}{\delta}}{\epsilon^2}\right). \tag{1.41}$$

Here, the asymptotic notation symbol Θ means that there are asymptotic upper and lower bounds that differ only by multiplicative constants (that are non-zero and finite).

Note that the scaling in $1/\delta$ is much better than required—logarithmic rather than polynomial. Hence, we could have formulated a stronger version of the fundamental theorem. However, requiring polynomial scaling is what is typically done in the general definition of PAC learnability.

What about generalizations to cases with $|\mathcal{Y}| > 2$? For both, classification ($\mathcal{Y}$ discrete) and regression ($\mathcal{Y}$ continuous), the concept of VC-dimension has been generalized and there exist various counterparts to the VC-dimension with similar implications. For the case of classification, the *graph dimension* d_G and the *Natarajan dimension* d_N are two useful generalizations that lead to quantitative bounds on the sample complexity of a hypotheses class with the error probability as risk function. In the binary case they both coincide with the VC-dimension, while in general $d_N \leq d_G \leq 4.67 d_N \log_2 |\mathcal{Y}|$ (cf. [5]). Known bounds on the sample complexity $\nu_{\mathcal{F}}$ turn out to have still the form of Eq.(1.41) with the only difference that in the upper and lower bound the role of the VC-dimension is played by d_G and d_N, respectively. The logarithmic gap between the two appears to be relevant and leads to the possibility of good and bad ERM learning algorithms (cf. [9]).

In the case of regression, a well-studied counterpart of the VC-dimension is the *fat-shattering dimension.* For particular loss functions (e.g., the squared loss) the above theorem then has a direct analogue, in the sense that under mild assumptions, uniform convergence, finite fat-shattering dimension and PAC learnability are equivalent [3]. In general learning contexts, however, uniform convergence turns out to be a strictly stronger requirement than PAC learnability [22, 9].

1.8 Rademacher complexity

The approaches discussed so far were distribution independent. Growth function and VC-dimension, as well as its various generalizations, depend only on the hypotheses class $\mathcal{F}$ and lead to PAC guarantees that are independent of the probability measures P. In this section we will consider an alternative approach and introduce the *Rademacher complexities.* These will not only depend on $\mathcal{F}$, but also on P or, alternatively, on the empirical distribution given by the data. This approach has several possible advantages compared to what we have discussed before. First, a data dependent approach may, in benign cases, provide better bounds than a distribution-free approach that has to cover the worst case, as well. Second, the approach based on Rademacher complexities allows us to go beyond binary classification and treat more general function classes that appear in classification or regression problems on an equal footing.

Definition 1.9 (Rademacher complexity)**.** *Consider a set of real-valued functions $\mathcal{G} \subseteq \mathbb{R}^{\mathcal{Z}}$ and a vector $z \in \mathcal{Z}^n$. The* empirical Rademacher complexity *of $\mathcal{G}$ w.r.t. z is defined as*

$$\hat{\mathcal{R}}(\mathcal{G}) := \mathbb{E}_\sigma \left[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i g(z_i) \right], \tag{1.42}$$

where $\mathbb{E}_\sigma$ denotes the expectation w.r.t. a uniform distribution of $\sigma \in \{-1,1\}^n$. If the z_i's are considered values of a vector of i.i.d.random variables $Z := (Z_1, \ldots, Z_n)$, each distributed according to a probability measure P on $\mathcal{Z}$, then the Rademacher complexities *of $\mathcal{G}$ w.r.t. P are given by*

$$\mathcal{R}_n(\mathcal{G}) := \mathbb{E}_Z \left[\hat{\mathcal{R}}(\mathcal{G}) \right]. \tag{1.43}$$

Note: The uniformly distributed σ_i's are called *Rademacher variables.* Whenever we want to emphasize the dependence of $\hat{\mathcal{R}}(\mathcal{G})$ on $z \in \mathcal{Z}^n$, we will write $\hat{\mathcal{R}}_z(\mathcal{G})$. Similarly, we occasionally write $\mathcal{R}_{n,P}(\mathcal{G})$ to make the dependence on P explicit. We will tacitly assume that $\mathcal{G}$ is chosen so that the suprema appearing in the definition lead to measurable functions.

The richer the function class $\mathcal{G}$, the larger the (empirical) Rademacher complexity. If we define $g(z) := \big(g(z_1), \ldots, g(z_n)\big)$ and write

$$\hat{\mathcal{R}}(\mathcal{G}) = \frac{1}{n}\mathbb{E}_\sigma \left[\sup_{g\in\mathcal{G}} \langle \sigma, g(z)\rangle\right],$$

we see that the (empirical) Rademacher complexity measures how well the function class $\mathcal{G}$ can 'match Rademacher noise'. If for a random sign pattern σ there is always a function in $\mathcal{G}$ that is well aligned with σ in the sense that $\langle \sigma, g(z)\rangle$ is large, the Rademacher complexity will be large. Clearly, this will become more and more difficult when the number n of considered points is increased.

With the following Lemma, which is a refinement of Hoeffding's inequality, we can show that the Rademacher complexity is close to its empirical counterpart. This will imply that the Rademacher complexity can be estimated reliably from the data and that no additional knowledge about P is required.

Lemma 1.10 (McDiarmid's inequality[19])**.** *Let $(Z_1, \ldots, Z_n) = Z$ be a finite sequence of independent random variables, each with values in $\mathcal{Z}$ and $\varphi : \mathcal{Z}^n \to \mathbb{R}$ a measurable function such that $|\varphi(z) - \varphi(z')| \leq \nu_i$ whenever z and z' only differ in the i'th coordinate. Then for every $\epsilon > 0$*

$$\mathbb{P}\big[\varphi(Z) - \mathbb{E}\left[\varphi(Z)\right] \geq \epsilon\big] \leq \exp\left[-\frac{2\epsilon^2}{\sum_{i=1}^n \nu_i^2}\right]. \tag{1.44}$$

Note: the same inequality holds with $\varphi(Z)$ and $\mathbb{E}\left[\varphi(Z)\right]$ interchanged. This can be seen by replacing φ with $-\varphi$.

Lemma 1.11 (Rademacher vs. empirical Rademacher complexity)**.** *Let $\mathcal{G} \subseteq [a,b]^{\mathcal{Z}}$ be a set of real-valued functions. Then for every $\epsilon > 0$ and any product probability measure P^n on $\mathcal{Z}^n$ it holds that*

$$\mathbb{P}_Z\left[\big(\mathcal{R}_n(\mathcal{G}) - \hat{\mathcal{R}}_Z(\mathcal{G})\big) \geq \epsilon\right] \leq \exp - \frac{2n\epsilon^2}{(b-a)^2}. \tag{1.45}$$

Proof. Define $\varphi : \mathcal{Z}^n \to \mathbb{R}$ as $\varphi(z) := \hat{\mathcal{R}}_z(\mathcal{G})$, which implies $\mathbb{E}\left[\varphi(Z)\right] = \mathcal{R}_n(\mathcal{G})$. Let $z, z' \in \mathcal{Z}^n$ be a pair that differs in only one component. Then $\sup_{g\in\mathcal{G}} \sum_i \sigma_i g(z_i)$ changes by at most $|b-a|$ if we replace z by z'. Consequently,

$$|\varphi(z) - \varphi(z')| = |\hat{\mathcal{R}}_z(\mathcal{G}) - \hat{\mathcal{R}}_{z'}(\mathcal{G})| \leq \frac{|b-a|}{n}, \tag{1.46}$$

and we can apply McDiarmid's inequality to obtain the stated result. □

Now we are prepared for the main result of this section and can prove a PAC-type guarantee based on (empirical) Rademacher complexities:

Theorem 1.13: PAC-type bound via Rademacher complexities

Consider arbitrary spaces $\mathcal{X}, \mathcal{Y}$, a hypotheses class $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$, a loss function $L : \mathcal{Y} \times \mathcal{Y} \to [0, c]$ and define $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$ and $\mathcal{G} := \{(x, y) \mapsto L(y, h(x)) \mid h \in \mathcal{F}\} \subseteq [0, c]^{\mathcal{Z}}$. For any $\delta > 0$ and any probability measure P on $\mathcal{Z}$ we have with probability at least $(1 - \delta)$ w.r.t. repeated sampling of P^n-distributed training data $S \in \mathcal{Z}^n$: all $h \in \mathcal{F}$ satisfy

$$R(h) - \hat{R}(h) \quad \leq \quad 2\mathcal{R}_n(\mathcal{G}) + c\sqrt{\frac{\ln \frac{1}{\delta}}{2n}}, \quad \text{and} \tag{1.47}$$

$$R(h) - \hat{R}(h) \quad \leq \quad 2\hat{\mathcal{R}}_S(\mathcal{G}) + 3c\sqrt{\frac{\ln \frac{2}{\delta}}{2n}}. \tag{1.48}$$

Proof. Defining $\varphi : \mathcal{Z}^n \to \mathbb{R}$ as $\varphi(S) := \sup_{h \in \mathcal{F}} \big(R(h) - \hat{R}(h)\big)$, we can apply McDiarmid's inequality to φ with $\nu_i = \frac{c}{n}$ and obtain

$$\mathbb{P}_S \left[\varphi(S) - \mathbb{E}_S \left[\varphi(S)\right] \geq \epsilon\right] \leq e^{-2n\epsilon^2/c^2}.$$

Setting the r.h.s. equal to δ and solving for ϵ then gives that with probability at least $1 - \delta$ we have

$$\sup_{h \in \mathcal{F}} \big(R(h) - \hat{R}(h)\big) \leq \mathbb{E}_S \left[\varphi(S)\right] + c\sqrt{\frac{\ln \frac{1}{\delta}}{2n}}. \tag{1.49}$$

It remains to upper bound the expectation on the right. To this end, we will again introduce a second sample S' that is an i.i.d. copy of S. Then

$$\begin{aligned}
\mathbb{E}_S \left[\varphi(S)\right] &= \mathbb{E}_S \left[\sup_{h \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{S'} \left[L\big(Y_i', h(X_i')\big) - L\big(Y_i, h(X_i)\big)\right]\right] \\
&\leq \mathbb{E}_{SS'} \left[\sup_{h \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n L\big(Y_i', h(X_i')\big) - L\big(Y_i, h(X_i)\big)\right] \\
&= \mathbb{E}_{SS'} \mathbb{E}_\sigma \left[\sup_{h \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n \sigma_i \Big(L\big(Y_i', h(X_i')\big) - L\big(Y_i, h(X_i)\big)\Big)\right] \\
&\leq 2\, \mathbb{E}_S \mathbb{E}_\sigma \left[\sup_{h \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n \sigma_i L\big(Y_i, h(X_i)\big)\right] \;=\; 2\mathcal{R}_n(\mathcal{G}),
\end{aligned}$$

where between the second and third line we have used that multiplication with $\sigma_i = -1$ amounts to interchanging $(X_i, Y_i) \leftrightarrow (X_i', Y_i')$, which has no effect as these are i.i.d. random variables. This proves Eq.(1.47). In order to obtain Eq.(1.48) note that by Lemma 1.11 with probability at least $1 - \delta/2$ we have

$$\mathcal{R}_n(\mathcal{G}) \leq \hat{\mathcal{R}}(\mathcal{G}) + c\sqrt{\frac{\ln \frac{2}{\delta}}{2n}}.$$

Combining this via the union bound with Eq.(1.47), where the latter is also applied to $\delta/2$ instead of δ, then yields the desired result. □

When applying the previous theorem to the case of binary classification, one can replace the Rademacher complexities of $\mathcal{G}$ by those of the hypotheses class $\mathcal{F}$:

Lemma 1.12 (Rademacher complexities for binary classification)**.** *Consider a hypotheses class $\mathcal{F} \subseteq \{-1,1\}^{\mathcal{X}}$, $L(y,y') := \mathbb{1}_{y\neq y'}$ as loss function and $\mathcal{G} := \{(x,y) \mapsto L(y,h(x)) \mid h \in \mathcal{F}\}$. Denote the restriction of $S = \big((x_i,y_i)\big)_{i=1}^n \in (\mathcal{X}\times\{-1,1\})^n$ to $\mathcal{X}$ by $S_{\mathcal{X}} := (x_i)_{i=1}^n$. For any probability measure P on $\mathcal{X}\times\{-1,1\}$ with marginal p on $\mathcal{X}$ we have*

$$\hat{\mathcal{R}}_S(\mathcal{G}) = \frac{1}{2}\hat{\mathcal{R}}_{S_{\mathcal{X}}}(\mathcal{F}) \quad \textit{and} \quad \mathcal{R}_{n,P}(\mathcal{G}) = \frac{1}{2}\mathcal{R}_{n,p}(\mathcal{F}). \tag{1.50}$$

Proof. The second equation is obtained from the first by taking the expectation value. The first is obtained by exploiting that $L(y,h(x)) = (1-yh(x))/2$. Then

$$\begin{aligned}\hat{\mathcal{R}}_S(\mathcal{G}) &= \mathbb{E}_\sigma\left[\sup_{h\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^n \sigma_i\big(1-y_ih(x_i)\big)/2\right]\\ &= \frac{1}{2}\mathbb{E}_\sigma\left[\sup_{h\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^n \sigma_i h(x_i)\right] \;=\; \frac{1}{2}\hat{\mathcal{R}}_{S_{\mathcal{X}}}(\mathcal{F}),\end{aligned}$$

where we have used that $\mathbb{E}_\sigma[\sigma_i] = 0$ and that the distributions of $-\sigma_i y_i$ and σ_i are the same. □

If, similar to the last part of the proof, we use that σ_i and $-\sigma_i$ are equally distributed, we can write

$$\hat{\mathcal{R}}_{S_{\mathcal{X}}}(\mathcal{F}) = \mathbb{E}_\sigma\left[\sup_{h\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^n -\sigma_i h(x_i)\right] = -\mathbb{E}_\sigma\left[\inf_{h\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^n \sigma_i h(x_i)\right].$$

Hence, computing the empirical Rademacher complexity is an optimization problem similar to empirical risk minimization—so it may be hard. The Rademacher complexity $\mathcal{R}_n$ itself depends on an unknown distribution and is therefore difficult to estimate, as well. However, it can be bounded for instance in the binary case in terms of the growth function or the VC-dimension. More specifically,

$$\mathcal{R}_n(\mathcal{F}) \leq \sqrt{\frac{2\ln\Gamma(n)}{n}} \quad \text{and} \quad \mathcal{R}_n(\mathcal{F}) \leq C\sqrt{\frac{\text{VCdim}(\mathcal{F})}{n}}, \tag{1.51}$$

for some universal constant C. These inequalities will be proven in Cor.1.14 and Cor.1.18.

Before going there, let us collect some properties of the Rademacher complexities and a Lemma, which turn out to be useful for their application and estimation.

Theorem 1.14: Properties of Rademacher complexities

Let $\mathcal{G}, \mathcal{G}_1, \mathcal{G}_2 \subseteq \mathbb{R}^{\mathcal{Z}}$ be classes of real-valued functions on $\mathcal{Z}$ and $z \in \mathcal{Z}^n$. The following holds for the empirical Rademacher complexities w.r.t. z:

1. If $c \in \mathbb{R}$, then $\hat{\mathcal{R}}(c\mathcal{G}) = |c|\hat{\mathcal{R}}(\mathcal{G})$.
2. $\mathcal{G}_1 \subseteq \mathcal{G}_2$ implies $\hat{\mathcal{R}}(\mathcal{G}_1) \leq \hat{\mathcal{R}}(\mathcal{G}_2)$.
3. $\hat{\mathcal{R}}(\mathcal{G}_1 + \mathcal{G}_2) = \hat{\mathcal{R}}(\mathcal{G}_1) + \hat{\mathcal{R}}(\mathcal{G}_2)$.
4. $\hat{\mathcal{R}}(\mathcal{G}) = \hat{\mathcal{R}}(\text{conv } \mathcal{G})$, where conv denotes the convex hull.
5. If $\varphi : \mathbb{R} \to \mathbb{R}$ is $L-$Lipschitz, then $\hat{\mathcal{R}}(\varphi \circ \mathcal{G}) \leq L\, \hat{\mathcal{R}}(\mathcal{G})$.

Proof. (sketch) 1.-3. follow immediately from the definition.

4. follows from the simple observation that

$$\sup_{\lambda \in \mathbb{R}^m_+, ||\lambda||_1 = 1} \sum_{i=1}^n \sum_{l=1}^m \lambda_l \sigma_i g_l(z_i) = \max_{l \in \{1,\ldots,m\}} \sum_{i=1}^n \sigma_i g_l(z_i).$$

5. Define $V := \{v \in \mathbb{R}^n \mid \exists g \in \mathcal{G}\ \forall i : \ v_i = g(z_i)\}$. Then

$$\begin{aligned} n\hat{\mathcal{R}}(\varphi \circ \mathcal{G}) &= \mathbb{E}_\sigma \left[\sup_{v \in V} \sum_{i=1}^n \sigma_i \varphi(v_i) \right] && (1.52) \\ &= \frac{1}{2} \mathbb{E}_{\sigma_2,\ldots,\sigma_n} \left[\sup_{v,v' \in V} \varphi(v_1) - \varphi(v_1') + \sum_{i=2}^n \sigma_i \big(\varphi(v_i) + \varphi(v_i')\big) \right] \\ &\leq \frac{1}{2} \mathbb{E}_{\sigma_2,\ldots,\sigma_n} \left[\sup_{v,v' \in V} L|v_1 - v_1'| + \sum_{i=2}^n \sigma_i \big(\varphi(v_i) + \varphi(v_i')\big) \right] \\ &= \mathbb{E}_\sigma \left[\sup_{v \in V} L\sigma_1 v_1 + \sum_{i=2}^n \sigma_i \varphi(v_i) \right], && (1.53) \end{aligned}$$

where in the last step we used that the absolute value can be dropped since the expression is invariant w.r.t. interchanging $v \leftrightarrow v'$. Repeating the above steps for the other $n-1$ components then leads to the claimed result. □

Remark: sometimes the definition of the (empirical) Rademacher complexity in the literature differs from the one in Eqs.(1.42, 1.43) and the absolute value is taken, i.e., the empirical quantity is defined as $\mathbb{E}_\sigma \left[\sup_g \left| \sum_i \sigma_i g(x_i) \right|\right]$ instead. In this case Thm.1.14 essentially still holds with small variations: then 3. becomes an inequality '$\leq$' and 5. requires in addition that $\varphi(-z) = -\varphi(z)$ (see [1]).

Lemma 1.13 (Massart's Lemma)**.** *Let A be a finite subset of $\mathbb{R}^m$ that is contained in a Euclidean ball of radius r. Then*

$$\mathbb{E}_\sigma \left[\max_{a \in A} \sum_{i=1}^m \sigma_i a_i \right] \leq r\sqrt{2 \ln |A|}, \tag{1.54}$$

where the expectation value is w.r.t. uniformly distributed Rademacher variables $\sigma \in \{-1,1\}^m$.

Proof. W.l.o.g. we can assume that the center of the ball is at the origin since Eq.(1.54) is unaffected by a translation. We introduce a parameter $\lambda > 0$ to be chosen later and first compute an upper bound for the rescaled set λA:

$$\begin{aligned}\mathbb{E}_\sigma\left[\max_{a\in\lambda A}\sum_{i=1}^m \sigma_i a_i\right] &\leq \mathbb{E}_\sigma\left[\ln\sum_{a\in\lambda A} e^{\sigma\cdot a}\right] \leq \ln\mathbb{E}_\sigma\left[\sum_{a\in\lambda A} e^{\sigma\cdot a}\right] \quad (1.55)\\ &= \ln\sum_{a\in\lambda A}\prod_{i=1}^m \frac{e^{a_i}+e^{-a_i}}{2} \quad (1.56)\\ &\leq \ln\sum_{a\in\lambda A} e^{||a||_2^2/2} \leq \frac{1}{2}r^2\lambda^2 + \ln|A|. \quad (1.57)\end{aligned}$$

Here, the first step is most easily understood when taking the exponential on both sides of the inequality for a fixed value of σ. Then the first inequality in Eq.(1.55) reduces to the statement that the maximum over positive numbers can be upper bounded by their sum. The second inequality uses concavity of the logarithm together with Jensen's inequality. Eq. (1.56) uses that the σ_i's are independently and uniformly distributed. The step to Eq.(1.57) exploits that $e^x + e^{-x} \leq 2e^{x^2/2}$ holds for all $x \in \mathbb{R}$. The final inequality then bounds the sum by its maximal element multiplied by the number of terms.

We then obtain the claimed result by inserting $\lambda = \sqrt{2\ln|A|}/r$ into

$$\mathbb{E}_\sigma\left[\max_{a\in A}\sum_{i=1}^m \sigma_i a_i\right] \leq \left(\frac{1}{2}r^2\lambda^2 + \ln|A|\right)/\lambda.$$

□

We can now use this Lemma to prove the claimed relation between the Rademacher complexities and the growth function of a function class:

Corollary 1.14 (Growth function bound on Rademacher complexity)**.** *Let* $\mathcal{Y} \subset \mathbb{R}$ *be a finite set of real numbers of modulus at most* $c > 0$. *The Rademacher complexity of any function class* $\mathcal{G} \subseteq \mathcal{Y}^{\mathcal{X}}$ *can then be bounded in terms of its growth function by*

$$\mathcal{R}_n(\mathcal{G}) \leq c\sqrt{\frac{2\ln\Gamma(n)}{n}}. \quad (1.58)$$

Proof. The statement follows directly from Massart's Lemma (Lem. 1.13) together with the fact that $||z||_2 \leq c\sqrt{n}$ if $z := (g(x_1),\ldots,g(x_n))$ for some $g \in \mathcal{G}$ and $x \in \mathcal{X}^n$. Using $r := c\sqrt{n}$ in Massart's Lemma then gives

$$\begin{aligned}\mathcal{R}_n(\mathcal{G}) &= \mathbb{E}_Z\mathbb{E}_\sigma \sup_{g\in\mathcal{G}}\frac{1}{n}\sum_{i=1}^n \sigma_i g(x_i)\\ &\leq \frac{r}{n}\sqrt{2\ln\Gamma(n)} = c\sqrt{\frac{2\ln\Gamma(n)}{n}}.\end{aligned}$$

□

1.9 Covering numbers

Rademacher complexities, growth function and VC dimension all measure the complexity of infinite function classes. In this section, we will quantify this complexity in a different way: we will discretize the space of functions and consider the minimal number of discretization points that is necessary to approximate any function to a given degree. The obtained covering and packing numbers will then turn out to be a useful tool, in particular, for obtaining bounds on Rademacher complexities.

Definition 1.15 (Coverings and packings). *Let $(\mathcal{M}, d)$ be a pseudometric space*[4]*, $A, B \subseteq \mathcal{M}$ and $\epsilon > 0$.*

- *A is called* ϵ-cover *of B if $\forall b \in B\ \exists a \in A : d(a,b) \leq \epsilon$. It is called an* internal *cover if in addition $A \subseteq B$. The* ϵ-covering number *of B, denoted by $N(\epsilon, B)$, is the smallest cardinality of any ϵ-cover of B. If only internal covers are considered, we will write $N_{in}(\epsilon, B)$.*
- *$A \subseteq B$ is called an* ϵ-packing *of B if $a, b \in A \Rightarrow d(a,b) > \epsilon$. The* ϵ-packing number *of B, denoted by $M(\epsilon, B)$, is the largest cardinality of any ϵ-packing of B.*

Note that by definition $N_{in}(\epsilon, B) \geq N(\epsilon, B)$. In fact, all those numbers are closely related:

Lemma 1.16 (Packing vs. covering). *For every pseudometric space $(\mathcal{M}, d)$ and $B \subseteq \mathcal{M}$:*

$$N(\epsilon/2, B) \geq M(\epsilon, B) \geq N_{in}(\epsilon, B).$$

Proof. Assume that $A \subseteq B$ is a maximal ϵ-packing of B, i.e. such that no more point can be added to A without violating the ϵ-packing property. Then for every $b \in B$ there is an $a \in A$ s.t. $d(a,b) \leq \epsilon$. Hence, A is an internal ϵ-cover of B and therefore $M(\epsilon, B) \geq N_{in}(\epsilon, B)$.

Conversely, let C be a smallest $\epsilon/2$-cover of B, i.e. $|C| = N(\epsilon/2, B)$. If A is an ϵ-packing of B, then the ball $\{b \in B \mid d(c,b) < \epsilon/2\}$ around any $c \in C$ contains at most one element from A: if there were two elements $a, a' \in A$, then $d(a, a') \leq d(a,c) + d(c, a') < \epsilon$ would contradict the ϵ-packing assumption. So $|C| \geq |A|$ and thus $N(\epsilon/2, B) \geq M(\epsilon, B)$. □

Example 1.7 (Norm balls in $\mathbb{R}^d$). Let $||\cdot||$ be any norm on $\mathbb{R}^d$, $B_r(x) := \{z \in \mathbb{R}^d |\ ||z-x|| \leq r\}$ and $\{x_1, \ldots, x_M\} \subset \mathbb{R}^d$ a maximal ϵ-packing of $B_r(0)$. That is, w.r.t. the metric induced by the norm we have $M = M(\epsilon, B_r(0))$. Then the balls $B_{\epsilon/2}(x_i)$ are mutually disjoint and lie inside $B_{r+\epsilon/2}(0)$. If $v := vol(B_1(0))$ is the

[4] A *pseudometric space* lacks only one property to a metric space: distinct points are not required to have distance zero.

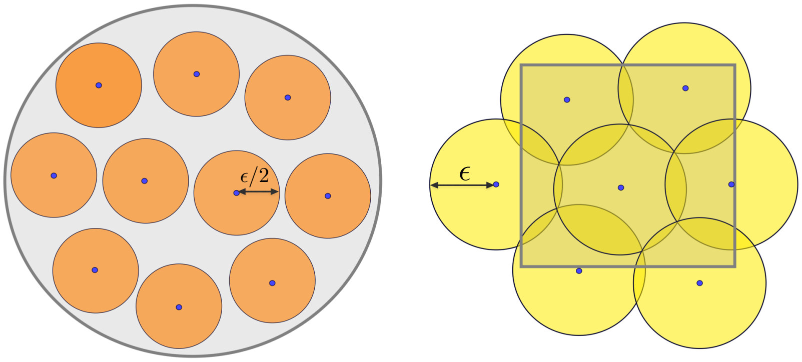


Figure 1.3: Left: the set of blue points forms an ϵ-packing of the gray disk—the $\epsilon/2$-balls around them are non-intersecting. Right: an ϵ-cover of the gray square. The cover is not internal.

volume (i.e., Lebesgue measure) of the unit ball, then $vol(B_{\epsilon/2}(x_i)) = (\epsilon/2)^d v$ and $vol(B_{r+\epsilon/2}(0)) = (r+\epsilon/2)^d v$. So under the assumption that $\epsilon \leq r$ we obtain the bound:

$$M\big(\epsilon, B_r(0)\big) \leq \frac{(r+\epsilon/2)^d v}{(\epsilon/2)^d v} \leq \left(\frac{3r}{\epsilon}\right)^d . \tag{1.59}$$

This examples exhibits a typical behavior of covering and packing numbers: the ϵ-packing number of a bounded object B of algebraic dimension d typically scales as

$$\ln M(\epsilon, B) \sim d \ln \frac{1}{\epsilon} \ .$$

One of the central observations that make covering and packing numbers useful in the context of statistical learning theory is that this relation still holds when the algebraic dimension is replaced by a combinatorial dimension, such as the VC-dimension. One bound of this type is the content of the subsequent Lemma. In order to state it, we first need to introduce the metrics with respect to which the covering and packing numbers will be considered. For any set $\mathcal{Z}$, $z \in \mathcal{Z}^n$ and any function class $\mathcal{G} \subseteq \mathbb{R}^{\mathcal{Z}}$ define the $||\cdot||_{p,z}$-seminorm on the linear span of $\mathcal{G}$ for $p \in [1,\infty)$ as

$$||g||_{p,z} := \left(\frac{1}{n}\sum_{i=1}^{n} |g(z_i)|^p\right)^{1/p} . \tag{1.60}$$

The corresponding pseudometric is then given by $(g_1, g_2) \mapsto ||g_1 - g_2||_{p,z}$. Note that due to $||g||_{p,z} \leq ||g||_{q,z}$ for $p \leq q$ there is a monotone behavior of packing/covering numbers when computed w.r.t. different values of p. For instance,

$$M(\epsilon, \mathcal{G}, ||\cdot||_{p,z}) \leq M(\epsilon, \mathcal{G}, ||\cdot||_{q,z}) \quad \text{if } p \leq q.$$

If the ranges of functions in $\mathcal{G}$ are uniformly bounded, then all these packing/covering numbers are finite, which can be seen by simply discretizing the range space. However, if no further constraint is imposed on the class of functions, the covering and packing numbers will grow exponentially with n. However, if, for instance, the VC-dimension of the considered function class is bounded, then an n-independent bound on the packing number can be given:

Lemma 1.17 (Packing numbers for binary classifiers)**.** *For any $\mathcal{F} \subseteq \{0,1\}^{\mathcal{X}}$, $x \in \mathcal{X}^n$, $\epsilon \in (0,1]$ and $p \in [1,\infty)$ the ϵ-packing number w.r.t. $||\cdot||_{p,x}$ can be bounded in terms of $VCdim(\mathcal{F}) =: d$ via*

$$M(\epsilon, \mathcal{F}) \leq \Big(\frac{9}{\epsilon^p} \ln \frac{2e}{\epsilon^p}\Big)^d. \tag{1.61}$$

Proof. It suffices to prove the statement for $p = 1$. Due to the particular binary target space, the general case follows from $||f_1 - f_2||_{p,x}^p = ||f_1 - f_2||_{1,x}$, which implies $M(\epsilon, \mathcal{F}, ||\cdot||_{p,x}) = M(\epsilon^p, \mathcal{F}, ||\cdot||_{1,x})$.

Let $\{f_1, \ldots, f_M\} \subseteq \mathcal{F}$ be a maximal ϵ-packing of $\mathcal{F}$. That is, for all $k \neq l$: $||f_k - f_l||_{1,x} > \epsilon$ and $M = M(\epsilon, \mathcal{F})$. Note that $||f_k - f_l||_{1,x}$ can be interpreted as the probability that $f_k(x_i) \neq f_l(x_i)$ when x_i is drawn uniformly from x (when the latter is regarded as an n-element set). So if $A \subseteq x$ is a random subset with $|A| = m$, then $\mathbb{P}_A[f_k|_A = f_l|_A] \leq (1-\epsilon)^m \leq e^{-m\epsilon}$. Using the union bound, this leads to

$$\mathbb{P}_A\big[\exists k, l: \; k \neq l \wedge f_k|_A = f_l|_A\big] < M^2 e^{-m\epsilon}.$$

If this probability is smaller than one, as it is the case when $m \geq \frac{2}{\epsilon} \ln M$, then there is an A on which all f_k's differ. This implies that $\mathcal{F}|_A$ contains at least M different functions and therefore $M \leq \Gamma(m)$. Using that $\Gamma(m) \leq (em/d)^d$ (Thm.1.8) and inserting $m = \frac{2}{\epsilon} \ln M$ this can be written as

$$M^{1/d} \leq \frac{2e}{\epsilon} \ln M^{1/d}. \tag{1.62}$$

Exploiting that $a \leq b \ln a \Rightarrow a \leq (1-1/e)^{-1} b \ln b$ and applying it with $a = M^{1/d}$, $b = 2e/\epsilon$ to Eq.(1.62) then leads to $M \leq \Big(\frac{9}{\epsilon} \ln \frac{2e}{\epsilon}\Big)^d$. □

The following theorem exploits covering numbers to bound Rademacher complexities.

Theorem 1.15: Dudley's theorem

For a fixed vector $z \in \mathcal{Z}^n$ let $\mathcal{G}$ be any subset of the pseudometric space $\big(\mathbb{R}^{\mathcal{Z}}, ||\cdot||_{2,z}\big)$ and set $\gamma_0 := \sup_{g \in \mathcal{G}} ||g||_{2,z}$. The empirical Rademacher complexity of $\mathcal{G}$ w.r.t. z can by upper bounded in terms of the covering numbers $N(\epsilon, \mathcal{G})$ via

$$\hat{\mathcal{R}}(\mathcal{G}) \leq \inf_{\epsilon \in [0, \gamma_0/2)} \; 4\epsilon + \frac{12}{\sqrt{n}} \int_{\epsilon}^{\gamma_0} \big(\ln N(\beta, \mathcal{G})\big)^{1/2} d\beta. \tag{1.63}$$

Remark: often Dudley's theorem is stated without the additional infimum, by choosing $\epsilon = 0$. One advantage of the above form is that the expression remains useful for function classes for which, for instance, $\ln N(\beta, \mathcal{G})$ grows faster than $1/\beta^2$ for $\beta \to 0$. In this case, the integral would diverge, when starting at $\epsilon = 0$.

Proof. Define $\gamma_j := 2^{-j}\gamma_0$ for $j \in \mathbb{N}$ and let $G_j \subseteq \mathbb{R}^{\mathcal{Z}}$ be a minimal γ_j-cover of $\mathcal{G}$. That is, $|G_j| = N(\gamma_j, \mathcal{G})$ and for every $g \in \mathcal{G}$ there is a $g_j \in G_j$ such that $||g - g_j||_{2,z} \leq \gamma_j$. This inequality continues to hold for $j = 0$ if we set $g_0 := 0$. For later use, we estimate

$$
\begin{aligned}
\frac{1}{n}\Big(\sum_{i=1}^{n} |g_j(z_i) - g_{j-1}(z_i)|^2\Big)^{1/2} &= \frac{1}{\sqrt{n}}||g_j - g_{j-1}||_{2,z} \\
&\leq \frac{1}{\sqrt{n}}\big(||g_j - g||_{2,z} + ||g - g_{j-1}||_{2,z}\big) \\
&\leq \frac{\gamma_j + \gamma_{j-1}}{\sqrt{n}} = \frac{3\gamma_j}{\sqrt{n}}. \qquad (1.64)
\end{aligned}
$$

For some $m \in \mathbb{N}$ to be chosen later, insert $g = g - g_m + \sum_{j=1}^{m}(g_j - g_{j-1})$ into the definition of the empirical Rademacher complexity. In this way, we obtain

$$
\begin{aligned}
\hat{\mathcal{R}}(\mathcal{G}) &= \frac{1}{n}\mathbb{E}_\sigma\Big[\sup_{g \in \mathcal{G}} \sum_{i=1}^{n} \sigma_i\Big(g(z_i) - g_m(z_i) + \sum_{j=1}^{m} g_j(z_i) - g_{j-1}(z_i)\Big)\Big] \\
&\leq \frac{1}{n}\mathbb{E}_\sigma\Big[\sup_{g \in \mathcal{G}} \sum_{i=1}^{n} \sigma_i\Big(g(z_i) - g_m(z_i)\Big)\Big] + \qquad (1.65) \\
&\quad \frac{1}{n}\sum_{j=1}^{m} \mathbb{E}_\sigma\Big[\sup_{g \in \mathcal{G}} \sum_{i=1}^{n} \sigma_i\big(g_j(z_i) - g_{j-1}(z_i)\big).\Big] \qquad (1.66)
\end{aligned}
$$

We bound the two summands separately. For the term in Eq.(1.65) we can use Cauchy-Schwarz for the inner product related to $|| \cdot ||_{2,z}$ to obtain the upper bound γ_m. For the term in Eq.(1.66) we can exploit Massart's Lemma (Lem.1.13) together with the estimate in Eq.(1.64). Hence, we can continue with

$$
\begin{aligned}
\hat{\mathcal{R}}(\mathcal{G}) &\leq \gamma_m + \frac{3}{\sqrt{n}} \sum_{j=1}^{m} \gamma_j \sqrt{2\ln\Big(|G_j| \cdot |G_{j-1}|\Big)} \\
&\leq \gamma_m + \frac{12}{\sqrt{n}} \sum_{j=1}^{m} (\gamma_j - \gamma_{j+1}) \sqrt{\ln N(\gamma_j, \mathcal{G})} \qquad (1.67) \\
&\leq \gamma_m + \frac{12}{\sqrt{n}} \int_{\gamma_{m+1}}^{\gamma_0} \sqrt{\ln N(\beta, \mathcal{G})}\, d\beta, \qquad (1.68)
\end{aligned}
$$

where we have used $|G_{j-1}| \leq |G_j|$ and $\gamma_j = 2(\gamma_j - \gamma_{j+1})$ in Eq.(1.67) and that the integral in Eq.(1.68) is lower bounded by its lower Riemann sum appearing in Eq.(1.67).

Finally, for any fixed $\epsilon \in [0, \gamma_0/2)$ choose m so that $\epsilon < \gamma_{m+1} \leq 2\epsilon$. Then $\gamma_m \leq 4\epsilon$ and Eq.(1.68) can be bounded from above by the expression in Eq.(1.63). □

Now, let us apply Dudley's theorem to the case of binary classification. Recall that when bounding the estimation error (or Rademacher complexity) in terms of the VC-dimension d directly via the growth function, an extra factor $\ln d$ appeared. Dudely's theorem improves this situation:

Corollary 1.18 (Improved bound for binary classifiers)**.** *Let $\mathcal{F} \in \{0,1\}^{\mathcal{X}}$ have $VCdim(\mathcal{F}) =: d$. Then its empirical Rademacher complexity w.r.t. an arbitrary point in $\mathcal{X}^n$ can be bounded by*

$$\hat{\mathcal{R}}(\mathcal{F}) \leq 31\sqrt{\frac{d}{n}} \ . \tag{1.69}$$

Proof. We use Eq.(1.63) from Dudely's theorem with $\epsilon = 0$ and $\gamma_0 \leq 1$. By Lemma 1.16 we can upper bound the covering number by the corresponding packing number for which we use the bound derived in Eq.(1.61). Using the simple inequality $\ln x \leq x/e$ this leads to $\ln N(\beta, \mathcal{F}) \leq d \ln \frac{18}{\beta^4}$ so that Dudley's theorem and numerical integration lead to

$$\hat{\mathcal{R}}(\mathcal{F}) \ \leq \ 12\sqrt{\frac{d}{n}} \int_0^1 \sqrt{\ln 18 + 4 \ln \frac{1}{\beta}} \, d\beta \ \leq \ 31\sqrt{\frac{d}{n}}.$$

□

Note: Since the r.h.s. of the bound does not depend on the empirical distribution, it holds for the expectation value, i.e., for the Rademacher complexities $\mathcal{R}_n(\mathcal{F})$, as well.

1.10 Fat-shattering dimension

... to be written ...

1.11 Algorithmic stability

So far, the type of the learning algorithm merely played a role through its range—the considered hypotheses class $\mathcal{F}$. In this section, we shift the focus from the range of the learning algorithm to its stability. Here, *stability* refers to the stability of the hypotheses at the output of the algorithm w.r.t small changes of its input. A 'small change of the input' typically means the change or omission of a single data point in the training data set. The most common way to quantify changes in the hypothesis is by means of the loss function. This leads to the following definitions:

Definition 1.19 (Stability). *Consider a loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. A learning algorithm that maps $S \in (\mathcal{X} \times \mathcal{Y})^n$ to a hypothesis h_S is said to be*

- uniformly stable *with rate $\epsilon : \mathbb{N} \to \mathbb{R}$ if for all $n \in \mathbb{N}$, $i \in \{1, \ldots, n\}$, $(x, y) \in \mathcal{X} \times \mathcal{Y}$ the following inequality holds for all $S, S' \in (\mathcal{X} \times \mathcal{Y})^n$ that differ in only one element:*

$$\Big| L\big(y, h_S(x)\big) - L\big(y, h_{S'}(x)\big) \Big| \leq \epsilon(n).$$

- on-average stable *with rate $\epsilon : \mathbb{N} \to \mathbb{R}$ if for all $n \in \mathbb{N}$ and all probability measures P on $\mathcal{X} \times \mathcal{Y}$:*

$$\Big| \mathbb{E}_{S \sim P^n} \mathbb{E}_{(x,y) \sim P} \mathbb{E}_i \Big[L\big(y_i, h_S(x_i)\big) - L\big(y_i, h_{S^i}(x_i)\big) \Big] \Big| \leq \epsilon(n), \qquad (1.70)$$

where $S = ((x_i, y_i))_{i=1}^n$, S^i is obtained from S by replacing the i'th element with (x, y) and $\mathbb{E}_i$ denotes the expectation with respect to a uniform distribution of $i \in \{1, \ldots, n\}$.

Obviously, uniform stability implies on-average stability with the same rate.

The presented definitions are often referred to as *replace-one stability*, as opposed to *leave-one-out stability*, where instead of replacing one data point it is omitted. Although the two ways of defining stability are conceptually very similar, they are formally incomparable. We focus on replace-one stability, as it is defined above. The following simple but crucial observation relates the generalization error to on-average stability:

Theorem 1.16: on-average stability = on-average generalization

With the notation of the foregoing definition the following holds for any learning algorithm $S \mapsto h_S$ and any probability measure P:

$$\mathbb{E}_S \big[R(h_S) - \hat{R}(h_S) \big] = \mathbb{E}_S \mathbb{E}_{(x,y)} \mathbb{E}_i \Big[L\big(y_i, h_{S^i}(x_i)\big) - L\big(y_i, h_S(x_i)\big) \Big] \quad (1.71)$$

Proof. On the one hand, since (x_i, y_i) and (x, y) are i.i.d. we can interchange them and write

$$\mathbb{E}_S \big[R(h_S) \big] \quad = \quad \mathbb{E}_S \mathbb{E}_{(x,y)} \Big[L\big(y, h_S(x)\big) \Big] = \mathbb{E}_S \mathbb{E}_{(x,y)} \Big[L\big(y_i, h_{S^i}(x_i)\big) \Big].$$

Since this holds equally for all i we may, in addition, take the expectation value $\mathbb{E}_i$ on the r.h.s.. On the other hand, we have

$$\mathbb{E}_S \big[\hat{R}(h_S) \big] = \mathbb{E}_S \mathbb{E}_i \Big[L\big(y_i, h_S(x_i)\big) \Big] = \mathbb{E}_S \mathbb{E}_{(x,y)} \mathbb{E}_i \Big[L\big(y_i, h_S(x_i)\big) \Big],$$

so that the difference of the two identities gives Eq.(1.71). □

As a consequence, we obtain for any on-average stable learning algorithm with rate $\epsilon(n)$ that

$$\Big| \mathbb{E}_S \big[R(h_S) - \hat{R}(h_S) \big] \Big| \leq \epsilon(n).$$

That is, the generalization error is, on average, bounded by the stability rate function. This generalization bound is weaker than the PAC-type bounds that we derived previously. In principle, closeness in expectation still leaves room for significant fluctuations, while PAC-type bounds guarantee that the empirical risk is close to the risk with high probability. However, such bounds can be derived from stability as well:

Theorem 1.17: PAC bound from stability

Consider a loss function with range in $[-c, c]$ and any learning algorithm $S \mapsto h_S$ that is uniformly stable with rate $\epsilon_1 : \mathbb{N} \to \mathbb{R}$. Then the following holds w.r.t. repeated sampling of training data sets of size n. For all $\epsilon > 0$ and all probability measures over $\mathcal{X} \times \mathcal{Y}$:

$$\mathbb{P}_S\Big[\big|\hat{R}(h_S) - R(h_S)\big| \geq \epsilon + \epsilon_1(n)\Big] \leq 2\exp\left[-\frac{n\epsilon^2}{2(n\epsilon_1(n)+c)^2}\right]. \tag{1.72}$$

Proof. We consider $\varphi(S) := \hat{R}(h_S) - R(h_S)$ as a function of n i.i.d. random variables to which we want to apply McDiarmid's inequality (Lemma 1.10). To this end, observe that $|\mathbb{E}[\varphi(S)]| \leq \epsilon_1(n)$, which follows from stability and Eq.(1.71), and note that $|\varphi(S)| \geq \epsilon + |\mathbb{E}[\varphi(S)]| \Rightarrow |\varphi(S) - \mathbb{E}[\varphi(S)]| \geq \epsilon$. Hence,

$$\begin{aligned}
\mathbb{P}_S\Big[\big|\hat{R}(h_S) - R(h_S)\big| \geq \epsilon + \epsilon_1(n)\Big] &\leq \mathbb{P}_S\Big[\big|\varphi(S) - \mathbb{E}[\varphi(S)]\big| \geq \epsilon\Big] \\
&\leq 2\exp\left[-\frac{2\epsilon^2}{n\nu^2}\right],
\end{aligned}$$

where the second step is McDiarmid's inequality with ν an upper bound on $|\varphi(S) - \varphi(S^i)|$ that is yet to be determined. This can be done by again applying the assumed stability to the inequality

$$\begin{aligned}
|\varphi(S) - \varphi(S^i)| &\leq \frac{1}{n}\sum_{j\neq i}\Big|L\big(y_j, h_S(x_j)\big) - L\big(y_j, h_{S^i}(x_j)\big)\Big| \\
&\quad + \frac{2c}{n} + \big|R(h_S) - R(h_{S^i})\big|.
\end{aligned}$$

We can bound the sum in the first line of the r.h.s. by $\epsilon_1(n)$ and, similarly,

$$\big|R(h_S) - R(h_{S^i})\big| = \Big|\mathbb{E}_{(X,Y)}\Big[L\big(Y, h_S(X)\big) - L\big(Y, h_{S^i}(X)\big)\Big| \leq \epsilon_1(n).$$

The claim then follows with $\nu := 2(\epsilon_1(n) + c/n)$. □

In the remaining part of this section we analyze the use of Tikhonov regularization as a means to guarantee uniform stability. To this end, we need the following notion from convex optimization:

Definition 1.20 (Strong convexity). *Let $\mathcal{F}$ be a convex subset of a real inner product space and $\alpha > 0$ a real number. A function $\Phi : \mathcal{F} \to \mathbb{R}$ is called* α-strongly convex, *if the map $h \mapsto \Phi(h) - \frac{\alpha}{2}\langle h, h\rangle$ is convex on $\mathcal{F}$.*

Denoting by $||\cdot||$ the norm induced by the inner product, it is straight forward to see that α-strong convexity is equivalent to requiring that

$$\lambda\Phi(h)+(1-\lambda)\Phi(g)\geq\Phi\big(\lambda h+(1-\lambda)g\big)+\frac{\alpha}{2}\lambda(1-\lambda)||h-g||^2 \tag{1.73}$$

holds for all $g,h\in\mathcal{F}$ and $\lambda\in[0,1]$. If the minimum of an α-strongly convex function exists, it is unique and the value of any other point can be bounded in terms of its distance to the minimizer:

Lemma 1.21. *If $\Phi:\mathcal{F}\to\mathbb{R}$ is α-strongly convex and attains its minimum at h, then for all $g\in\mathcal{F}$:*

$$\Phi(g)\geq\Phi(h)+\frac{\alpha}{2}||h-g||^2. \tag{1.74}$$

Proof. When using minimality of h in Eq.(1.73) we obtain $\lambda\Phi(h)+(1-\lambda)\Phi(g)\geq\Phi(h)+\alpha\lambda(1-\lambda)||h-g||^2/2$, which can be simplified to $\Phi(g)\geq\Phi(h)+\alpha\lambda||h-g||^2/2$. Setting $\lambda=1$ then completes the proof. □

With this, we are equipped for the following relation between stability and regularization:

Theorem 1.18: Uniform stability from regularization

Let $\lambda>0$ and $\mathcal{F}\subseteq\mathcal{Y}^{\mathcal{X}}$ be a convex subset of an inner product space. If for all $(x,y)\in\mathcal{X}\times\mathcal{Y}$ the map $h\mapsto L(y,h(x))$ is convex and $l-Lipschitz$ on $\mathcal{F}$, then the learning algorithm that minimizes the functional $f_S(h):=\hat{R}(h)+\lambda\langle h,h\rangle$ is uniformly stable with rate $\frac{2l^2}{\lambda n}$.

Proof. With $h:=h_S$, $h':=h_{S^i}$ and the norm being the one induced by the inner product we can bound

$$\begin{aligned}
f_S(h')-f_S(h) &= \hat{R}_S(h')-\hat{R}_S(h)+\lambda\big(||h'||^2-||h||^2\big)\\
&= \hat{R}_{S^i}(h')-\hat{R}_{S^i}(h)+\lambda\big(||h'||^2-||h||^2\big) \qquad (1.75)\\
&+ \frac{1}{n}\Big[L\big(y_i,h'(x_i)\big)-L\big(y_i,h(x_i)\big)+L\big(y,h(x)\big)-L\big(y,h'(x)\big)\Big]\\
&\leq \frac{2l}{n}\,||h_{S^i}-h_S||,
\end{aligned}$$

where we have used that the term in Eq.(1.75) is negative, since h' minimizes f_{S^i}, together with the Lipschitz assumption. As f_S is 2λ-strongly convex with minimizer h we can, on the other hand, exploit Lemma 1.21 to obtain $\lambda||h'-h||^2\leq f_S(h')-f_S(h)$. Combining these two bounds leads to $||h'-h||\leq 2l/(\lambda n)$. Using the Lipschitz property once again, we finally arrive at uniform stability:

$$\Big|L\big(y,h(x)\big)-L\big(y,h'(x)\big)\Big|\leq l||h-h'||\leq\frac{2l^2}{\lambda n}.$$

□

It is possible to derive a similar implication when replacing the Lipschitz assumption for the loss function by a Lipschitz assumption for its gradient. In either case, there is a trade-off between stability, and thus small generalization error, on the one hand and an effective restriction of the hypotheses class on the other hand: if λ is too small, there is no generalization guarantee. If λ is too large, hypotheses with small norm dominate, whether they describe the data adequately, or not. The following corollary aims at formalizing this trade-off:

Corollary 1.22 (Regularization trade-off)**.** *Let $\lambda > 0$ and $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a convex subset of an inner product space. Assume that for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$ the map $h \mapsto L(y, h(x))$ is convex and $l-$Lipschitz on $\mathcal{F}$ and define $h^* := \operatorname{argmin}_{h \in \mathcal{F}} R(h)$. Then the learning algorithm $S \mapsto h_S$ that minimizes the functional $f_S(h) := \hat{R}(h) + \lambda ||h||^2$ satisfies*

$$\mathbb{E}_S\big[R(h_S)\big] \leq R(h^*) + \lambda ||h^*||^2 + \frac{2l^2}{\lambda n}. \tag{1.76}$$

Proof. Since h_S minimizes f_S, we have

$$\mathbb{E}_S\big[\hat{R}(h_S)\big] \leq \mathbb{E}_S\big[f_S(h_S)\big] \leq \mathbb{E}_S\big[f_S(h^*)\big] = R(h^*) + \lambda ||h^*||^2.$$

As uniform stability implies on-average stability, we can use Thm.1.18 together with Thm.1.16 to obtain $\mathbb{E}_S\big[R(h_S) - \hat{R}(h_S)\big] \leq 2l^2/(\lambda n)$. Combining the two bounds then leads to the claimed result:

$$\begin{aligned} \mathbb{E}_S\big[R(h_S)\big] &= \mathbb{E}_S\big[\hat{R}(h_S)\big] + \mathbb{E}_S\big[R(h_S) - \hat{R}(h_S)\big] \\ &\leq R(h^*) + \lambda ||h^*||^2 + \frac{2l^2}{\lambda n}. \end{aligned}$$

□

The optimal value for λ that minimizes the r.h.s. of Eq.(1.76) is then

$$\lambda_{opt} = \frac{l}{||h^*||}\sqrt{\frac{2}{n}} \quad \Rightarrow \quad \mathbb{E}_S\big[R(h_S)\big] \leq R(h^*) + 2l\ ||h^*|| \sqrt{\frac{2}{n}}.$$

Clearly, in practice the norm $||h^*||$ is not known, but one could try to estimate it, for instance on the basis of a validation data set, and then work with the estimate.

The form of the above statements should, however, not hide the fact that Tikhonov regularization provides more freedom than the choice of λ, namely the choice of the inner product.

1.12 Relative entropy bounds

In this section we will slightly extend the framework and allow for a stochastic component in the learning algorithm. Instead of choosing a hypothesis $h \in \mathcal{F}$ deterministically upon input of a training data set $S \in (\mathcal{X} \in \mathcal{Y})^n$, the learning algorithms we deal with in this section choose a distribution of hypotheses, characterized by a probability measure μ_S on $\mathcal{F}$. A hypothesis $h \in \mathcal{F}$ is then drawn according to μ_S. In this way, the map from S to h becomes stochastic and the learning algorithm specifies the assignment $S \mapsto \mu_S$. In terms of probability theory, we may think of the learning algorithm as a Markov kernel. With slight abuse of notation, we will denote the corresponding expected risk and empirical risk by

$$R(\mu_S) := \mathbb{E}_{h\sim\mu_S}\big[R(h)\big] \qquad \text{and} \qquad \hat{R}_S(\mu_S) := \mathbb{E}_{h\sim\mu_S}\big[\hat{R}_S(h)\big]. \tag{1.77}$$

A useful toolbox for such stochastic schemes is given by so-called *PAC-Bayesian* bounds. These rely on a famous functional, which is ubiquitous in information theory, thermodynamics and statistical inference:

For two probability measures μ and ν defined on the same space, the *Kullback-Leibler divergence* (KL-divergence a.k.a. *relative entropy*) is defined as

$$KL(\mu||\nu) := \int \log\left[\frac{d\mu}{d\nu}\right] d\mu \tag{1.78}$$

if μ is absolutely continuous w.r.t. ν and $KL(\mu||\nu) = \infty$ otherwise.[5] Its main properties are that it is (i) non-negative and zero only if $\mu = \nu$ (almost everywhere w.r.t. μ), (ii) jointly convex and (iii) it satisfies a data-processing inequality, i.e., it is non-increasing if the random variables corresponding to the two arguments undergo the same stochastic map. The KL-divergence is not a metric since it does neither satisfy the triangle inequality nor is it symmetric. Nevertheless, it can be useful to think of it as a distance measure (in the colloquial sense) between probability distributions. In the present context, the KL-divergence enters the discussion via the following inequality:

Lemma 1.23 (Fenchel-Young inequality for KL-divergence)**.** [6] *Let μ, ν be probability measures on $\mathcal{F}$ and $\varphi : \mathcal{F} \to \mathbb{R}$ a measurable function. Then*

$$\log \int_{\mathcal{F}} e^{\varphi} d\nu \geq \int_{\mathcal{F}} \varphi d\mu - KL(\mu||\nu).$$

Here, equality holds if $d\mu/d\nu := e^{\varphi}/\big(\int e^{\varphi} d\nu\big)$.

Proof. We define a probability measure μ_0 via $d\mu_0/d\nu := e^{\varphi}/(\int e^{\varphi} d\nu)$. Assuming that $KL(\mu||\nu) < \infty$ (as otherwise the statement is trivial) we known that μ is absolutely continuous w.r.t. ν and thus also w.r.t. μ_0. Therefore

$$\begin{aligned}
\int_{\mathcal{F}} \varphi d\mu - KL(\mu||\nu) &= \int_{\mathcal{F}} \varphi d\mu - \int_{\mathcal{F}} \log\left[\frac{d\mu}{d\nu}\right] d\mu \\
&= \int_{\mathcal{F}} \varphi d\mu - \int_{\mathcal{F}} \log\left[\frac{d\mu_0}{d\nu}\right] d\mu - \int_{\mathcal{F}} \log\left[\frac{d\mu}{d\mu_0}\right] d\mu \\
&= \log\Big[\int_{\mathcal{F}} e^{\varphi} d\nu\Big] - KL(\mu||\mu_0).
\end{aligned}$$

[5] Here, $d\mu/d\nu$ is the Radon-Nikodym derivative. If the measures are given by probability densities p_μ and p_ν, then $KL(\mu||\nu) = \int p_\mu(x)\big[\log p_\mu(x) - \log p_\nu(x)\big]dx$.

[6] The fact that equality can be attained turns Lemma 1.23 into a useful variational principle that runs under the name *Gibbs variational principle* in thermodynamics.

The Lemma then follows from non-negativity of the Kullback-Leibler divergence, applied to $KL(\mu||\mu_0)$. □

This Lemma can now be used to derive a template for proving PAC-Bayesian bounds:

Proposition 1.24 (Template for PAC-Bayesian bounds)**.** *Let ν be a probability measure on $\mathcal{F}$, and $\phi : \mathcal{F} \times (\mathcal{X} \times \mathcal{Y})^n \to \mathbb{R}$. With probability at least $1-\delta$ w.r.t. repeated sampling of $S \in (\mathcal{X}\times\mathcal{Y})^n$, distributed according to a probability measure P_n, we have that for all probability measures μ on $\mathcal{F}$:*

$$\mathbb{E}_{h\sim\mu}\big[\phi(h,S)\big] \leq \frac{1}{n}\Big(KL(\mu||\nu) + \log(1/\delta) + \log \mathbb{E}_{h\sim\nu}\mathbb{E}_{S'\sim P_n}\big[e^{n\phi(h,S')}\big]\Big).$$

Proof. Applying Lemma 1.23 to $\varphi(h) := n\phi(h,S)$ gives

$$\mathbb{E}_{h\sim\mu}\big[\phi(h,S)\big] \leq \frac{1}{n}\Big(KL(\mu||\nu) + \log \mathbb{E}_{h\sim\nu}\big[e^{n\phi(h,S)}\big]\Big). \tag{1.79}$$

From Markov's inequality we know that with probability at least $1-\delta$ w.r.t. repeated sampling of S according to P_n, we have

$$\mathbb{E}_{h\sim\nu}\big[e^{n\phi(h,S)}\big] \leq \frac{1}{\delta}\mathbb{E}_{S'\sim P_n}\mathbb{E}_{h\sim\nu}\big[e^{n\phi(h,S')}\big].$$

Using that ν does not depend on the sample S' we can interchange the expectation values and insert the inequality into Eq.(1.79) to complete the proof. □

It is essential that ν is independent of S, whereas μ is allowed to depend on S. In fact, we will consider $\mu = \mu_S$, the distribution that characterizes the stochastic learning algorithm. Also note that at this point P_n is not required to be a product measure.

There are various reasonable choices for the function ϕ. The most popular one leads to the bounds in the following theorem in which $\mathrm{kl} : [0,1] \times [0,1] \to [0,\infty]$ denotes the binary relative entropy, i.e.,

$$\mathrm{kl}(p\|q) := p\log\frac{p}{q} + (1-p)\log\frac{1-p}{1-q},$$

which is nothing but the relative entropy between the discrete probability distributions $(p, 1-p)$ and $(q, 1-q)$.

Theorem 1.19: PAC-Bayesian bounds

Let ν be a probability measure on $\mathcal{F}$, $\delta \in (0,1)$, and R a risk function with values in $[0,1]$. With probability at least $1-\delta$ w.r.t. repeated sampling of $S \in (\mathcal{X} \times \mathcal{Y})^n$, distributed according to a product probability measure

P^n, the following inequalities hold for all probability measures μ on $\mathcal{F}$:

$$\mathrm{kl}\big(\hat{R}_S(\mu)\|R(\mu)\big) \leq \frac{\mathcal{K}}{n}, \tag{1.80}$$

$$\big|\hat{R}_S(\mu) - R(\mu)\big| \leq \sqrt{\frac{\mathcal{K}}{2n}}, \tag{1.81}$$

$$\big|\hat{R}_S(\mu) - R(\mu)\big| \leq \sqrt{\frac{2\hat{R}_S(\mu)\big(1-\hat{R}_S(\mu)\big)\mathcal{K}}{n}} + \frac{\mathcal{K}}{n}, \tag{1.82}$$

where $\mathcal{K} := KL(\mu||\nu) + \log(2\sqrt{n}/\delta)$.

Proof. (sketch) In order to derive Eq.(1.80), we use Prop.1.24 with $\phi(h,S) := \mathrm{kl}\big(\hat{R}_S(h)\|R(h)\big)$. The l.h.s. of the resulting inequality can be bounded by $\mathbb{E}_{h\sim\mu}\big[\mathrm{kl}\big(\hat{R}_S(h)\|R(h)\big)\big] \geq \mathrm{kl}\big(\hat{R}_S(\mu)\|R(\mu)\big)$ using joint convexity of the relative entropy. On the r.h.s. we have to estimate $E_{S'\sim P^n}\big[\exp n\ \mathrm{kl}\big(\hat{R}_{S'}\|R(h)\big)\big]$ for which the sharp[7] upper bound $2\sqrt{n}$ has been shown by Maurer. Eq.(1.81) follows from Eq.(1.80) via Pinsker's inequality, which states that $\mathrm{kl}(p,q) \geq 2|p-q|^2$. Similarly, Eq.(1.82) follows from Eq.(1.80) via the estimate

$$|p-q| \leq \sqrt{2p(1-p)\mathrm{kl}(p\|q)} + \mathrm{kl}(p\|q).$$

□

Since the theorem holds uniformly over all probability measures μ, we may as well allow the measure to depend on S and set $\mu = \mu_S$ with μ_S characterizing a stochastic learning algorithm.

As it is clear from the proof, Eq.(1.80) is the strongest among the three stated inequalities. The main purpose of Eq.(1.81) is to display a familiar form and the main purpose of Eq.(1.82) is to show that the original inequality has a better n-dependence in the regime of extremal empirical error. The additional logarithmic dependence on n turns out to be unnecessary since it can be shown, using generic chaining techniques, that for some constant C

$$R(\mu) - \hat{R}_S(\mu) \leq C\sqrt{\frac{KL(\mu||\nu) + \log(2/\delta)}{n}}.$$

The role of ν is the one of a 'free parameter' in the bound, which has to be chosen independent of S. It is often called a *prior distribution* although it need not reflect our knowledge or believe about the distribution—any ν will lead to a valid bound. Consider the simple case of a countable set $\mathcal{F}$ and a deterministic algorithm leading to a hypothesis h. If ν assigns a probability $p(h)$ to the hypotheses, then $KL(\mu||\nu) = \log\big(1/p(h)\big)$. Hence, we essentially recover Thm.1.5.

So which prior ν should we choose? A smaller relative entropy term means a better bound. We cannot make $KL(\mu_S||\nu)$ vanish though since ν must not

[7] Maurer also showed that $\sqrt{n}$ is a lower bound.

depend on S precluding $\nu = \mu_S$. However, ν is allowed to depend on n as well as on the distribution P from which the training data is drawn. Although P is unknown, we may obtain insightful bounds and/or be able to bound the resulting term in a P-independent way. In the light of this, it seems reasonable to choose ν so that $KL(\mu_S||\nu)$ is minimized in expectation:

Lemma 1.25 (Optimal prior distribution)**.** *For any (not necessarily product) distribution of S we obtain*

$$\text{argmin}_\nu \mathbb{E}_S\big[KL(\mu_S||\nu)\big] = \mathbb{E}_S[\mu_S], \tag{1.83}$$

if the minimum is taken over all probability measures ν.

Proof. This follows from considering the difference

$$\mathbb{E}_S\big[KL(\mu_S||\nu)\big] - KL\big(\mathbb{E}_S[\mu_S]||\nu\big) = H\big(\mathbb{E}_S[\mu_S]\big) - \mathbb{E}_S\big[H(\mu_S)\big].$$

Whatever the r.h.s. is[8], the crucial point is that this difference is independent of ν. Hence, the measure ν that minimizes $KL\big(\mathbb{E}_S[\mu_S]||\nu\big)$ must also minimize $\mathbb{E}_S\big[KL(\mu_S||\nu)\big]$. From the properties of the relative entropy we know, however, that the former is minimized (to zero) for $\nu = \mathbb{E}_S[\mu_S]$. □

If we insert the optimal prior into the expected relative entropy, we obtain the *mutual information* between the samples S and the hypotheses h. In general, if two random variables X and Y are governed by a joint distribution P_{XY} whose marginals are P_X and P_Y, then their mutual information is defined as

$$I(X:Y) := KL\big(P_{XY}||P_X \times P_Y\big), \tag{1.84}$$

i.e., the KL-divergence of P_{XY} from the product of its marginals. The mutual information quantifies correlations between random variables, it is symmetric in its arguments, non-negative and zero iff the random variables are independent.

Lemma 1.26. $\mathbb{E}_S\big[KL(\mu_S||\mathbb{E}_{S'}[\mu_{S'}])\big] = I(h:S)$*, where S' denotes an independent copy of S.*

Proof. The learning algorithm, described by μ_S, together with the distribution of S defines a probability distribution on the product space $\mathcal{F} \times (\mathcal{X} \times \mathcal{Y})^n$. Let us denote by $p(h,S)$ a corresponding probability density w.r.t. a suitable reference measure, by $p(h)$ and $p(S)$ the marginals of $p(h,S)$ and by $p(h|S) := p(h,S)/p(S)$ the conditional probability density. Then

$$\begin{aligned}\mathbb{E}_S\big[KL(\mu_S||\mathbb{E}_{S'}[\mu_{S'}])\big] &= \int_{(\mathcal{X}\times\mathcal{Y})^n} p(S)\left(\int_{\mathcal{F}} p(h|S)\log\left[\frac{p(h|S)}{p(h)}\right] dh\right) dS \\ &= \int\int p(h,S)\log\left[\frac{p(h,S)}{p(h)p(S)}\right] dh\, dS \;=\; I(h:S).\end{aligned}$$

□

[8] It is the difference of two differential entropies that are defined relative to a suitable reference measure.

Inserting this observation back into the PAC-Bayesian bound, we obtain the following:

Theorem 1.20: Mutual Information PAC-bound

Let μ_S describe a stochastic learning algorithm, $\delta \in (0,1)$ and let R be a risk function with values in $[0,1]$. With probability at least $1-\delta$ w.r.t. repeated sampling of $S \in (\mathcal{X} \times \mathcal{Y})^n$, distributed according to a product probability measure P^n, the following holds

$$\mathrm{kl}\big(\hat{R}_S(\mu_S)\|R(\mu_S)\big) \leq \frac{1}{n}\left(\frac{2I(h:S)}{\delta} + \log\left[\frac{4\sqrt{n}}{\delta}\right]\right). \tag{1.85}$$

Proof. We have to modify the proofs of Prop.1.24 and Thm.1.19 essentially only at the step following Eq.(1.79). Using $\nu = \mathbb{E}_S[\mu_S]$ as prior, we can exploit Lem.1.26 together with Markov's inequality and upper bound $KL(\mu_S||\nu)$ in Eq.(1.79) by $I(h:S)/\delta$. In order to take into account that this holds again only with probability at least $1-\delta$, we have to divide δ by two, invoking the union bound. In this way, Eq.(1.85) is obtain analogous to Eq.(1.80). □

In the same way as in the proof of Thm.1.19, we could derive variants of this bound, which we omit, though. Although Thm.1.20 is not directly applicable, as the mutual information depends on the unknown distribution of S, it has a remarkable interpretation: it guarantees good generalization if the correlation between hypotheses and training data is small. In other words, if the learning algorithm manages to achieve small empirical risk without having learned too much about the actual sample S, it will also perform well on unseen data. A similar mutual information bound can also be derived for the expected generalization error:

Theorem 1.21: Mutual Information vs. expected generalization

Consider a stochastic learning algorithm, described by μ_S, a risk function with values in $[0,1]$, and training data sets S drawn from a distribution over n independent elements. Then

$$\Big|\mathbb{E}_S\big[\hat{R}_S(\mu_S) - R(\mu_S)\big]\Big| \leq \sqrt{\frac{I(h:S)}{2n}}.$$

Proof. By definition, we can express the mutual information $I(h:S)$ in terms of the KL-divergence of the joint distribution from the product of its marginals. The KL-divergence, in turn, can be lower bounded using the Fenchel-Young inequality from Lem.1.23. With $\varphi(h,S) := \lambda\hat{R}_S(h)$ for $\lambda \in \mathbb{R}$ we obtain

$$\begin{aligned} I(h:S) &\geq \lambda\mathbb{E}\big[\hat{R}_S(h)\big] - \log\bar{\mathbb{E}}\Big[e^{\lambda\hat{R}_S(h)}\Big] \\ &\geq \lambda\mathbb{E}\big[\hat{R}_S(h)\big] - \frac{\lambda^2}{8n} - \lambda\bar{\mathbb{E}}\big[R(h)\big], \end{aligned} \tag{1.86}$$

where $\mathbb{E} = \mathbb{E}_S \mathbb{E}_{\mu_S}$ denotes the expectation w.r.t. the joint distribution of h and S, while $\bar{\mathbb{E}}$ denotes the expectation w.r.t. the product of their marginals. The second step in Eq.(1.86) follows from an application of Hoeffding's Lemma, which together with the independence of the n elements in S implies

$$\bar{\mathbb{E}}\Big[e^{\lambda\big(\hat{R}_S(h) - \bar{\mathbb{E}}[R(h)]\big)}\Big] \leq \exp\frac{\lambda^2}{8n}.$$

The statement in the Lemma is then obtained by taking the maximum over λ in Eq.(1.86) and noting that $\bar{\mathbb{E}}[R(h)] = \mathbb{E}[R(h)]$. □

An important property of the mutual information, which it inherits from the relative entropy, is the data processing inequality: in general, if A, B, C are random variables that form a Markov chain $A - B - C$ (i.e., A depends on C only via B), then $I(A : C) \leq I(A : B)$. Applied to the present context, if $h - B - S$ is a Markov chain that describes for instance preprocessing of the data or postprocessing of the hypotheses, then $I(h : S) \leq \min\{I(h : B), I(B : S)\}$.

Let us return to the PAC-Bayesian framework. Lem.1.25 derives the optimal prior ν for a fixed stochastic learning algorithm ('posterior') μ_S. What about the converse? What is the optimal choice for μ_S if ν is given? One possibility to interpret and address this question is to consider the expected weighted sum of empirical risk and relative entropy

$$\mathbb{E}_S\Big[\hat{R}_S(\mu_S) + \frac{1}{\beta} KL(\mu_S||\nu)\Big] \tag{1.87}$$

and ask for a μ_S that minimizes this expression. Here, $\beta > 0$ is a parameter that balances empirical risk minimization and generalization (where the latter is estimated by the relative entropy with prior ν, motivated by the PAC-Bayesian bounds). The resulting μ_S is often called the *Gibbs posterior* and the approach of sampling hypothesis according to the Gibbs posterior, the *Gibbs algorithm.*

Proposition 1.27 (Optimal posterior)**.** *For given β and ν, Eq.(1.87) is minimized for all (not necessarily product) distributions of S if μ_S is chosen as*

$$\frac{d\mu_S}{d\nu}(h) = \frac{e^{-\beta \hat{R}_S(h)}}{\mathbb{E}_{h'\sim\nu}\big[e^{-\beta\hat{R}_S(h')}\big]}.$$

Proof. This is an immediate consequence of the Fenchel-Young inequality for the relative entropy. Applying Lem.1.23 and setting $\varphi(h) = -\beta\hat{R}_S(h)$ we obtain

$$\begin{aligned}\mathbb{E}_S\Big[\hat{R}_S(\mu_S) + \frac{1}{\beta}KL(\mu_S||\nu)\Big] \;&\geq\; \mathbb{E}_S\Big[\mathbb{E}_{h\sim\mu_S}\Big[\hat{R}_S(h) + \frac{1}{\beta}\varphi(h)\Big] - \frac{1}{\beta}\log\int e^{\varphi}d\nu\Big] \\ &=\; -\frac{1}{\beta}\,\mathbb{E}_S\Big[\log\int e^{-\beta\hat{R}_S}d\nu\Big].\end{aligned}$$

Since this lower bound does not longer depend on μ_S, we can exploit the conditions for equality in Lem.1.23 and arrive at the stated result. □

Note that the Gibbs algorithm can be regarded as a stochastic version of empirical risk minimization, to which it converges in the limit $\beta \to \infty$.

Now it is tempting to combine the optimal posterior in Prop.1.27 with the optimal prior from Lem.1.25 or even to see-saw these optimizations. Unfortunately, the optimal prior turns out to be difficult to combine with the Gibbs posterior. The analysis becomes feasible, however, for the following pair of posterior and prior distribution, which is motivated by the above discussion. We define μ_S and ν in terms of probability densities p_S and q w.r.t. the same suitable reference measure[9] as

$$p_S(h) := \frac{e^{-\beta \hat{R}_S(h)}}{Z_p}, \qquad q(h) := \frac{e^{-\beta R(h)}}{Z_q}, \tag{1.88}$$

where Z_p and Z_q are normalization factors, defined by their purpose to let p_S and q become probability densities.

Theorem 1.22: Generalization bounds for Gibbs algorithms

Let μ_S and ν be given by Eq.(1.88) for some $\beta > 0$, and $\delta \in (0,1)$. With probability at least $1-\delta$ w.r.t. repeated sampling of $S \in (\mathcal{X} \times \mathcal{Y})^n$ from an i.i.d. distribution, the following two inequalities hold simultaneously

$$\mathrm{kl}\big(\hat{R}_S(\mu_S)\big)\|R(\mu_S) \leq \frac{1}{n}\Big(KL(\mu_S\|\nu) + \log\big[2\sqrt{n}/\delta\big]\Big), \tag{1.89}$$

$$KL\big(\mu_S\|\nu\big) \leq \frac{\beta^2}{2n} + \beta\sqrt{\frac{2}{n}\log\frac{2\sqrt{n}}{\delta}}. \tag{1.90}$$

Moreover, $I(h:S) \leq \min\big\{\beta, \beta^2/(2n)\big\}$, and

$$0 \leq \mathbb{E}_S\big[R(\mu_S) - \hat{R}_S(\mu_S)\big] \leq \frac{\beta}{2n}. \tag{1.91}$$

Proof. The first inequality is just Eq.(1.80). In order to arrive at the second inequality, observe that

$$\begin{aligned} KL\big(\mu_S||\nu\big) &= \mathbb{E}_{h\sim\mu_S} \log \frac{Z_q\, e^{-\beta \hat{R}_S(h)}}{Z_p\, e^{-\beta R(h)}} \\ &= \beta\, \mathbb{E}_{h\sim\mu_S}\big[R(h) - \hat{R}_S(h)\big] - \log\frac{Z_p}{Z_q} \\ &= \beta\, \mathbb{E}_{h\sim\mu_S}\big[R(h) - \hat{R}_S(h)\big] - \log \int q(h)\, e^{\beta\left(R(h)-\hat{R}_S(h)\right)} dh \\ &\leq \beta\, \big(\mathbb{E}_{h\sim\mu_S} - \mathbb{E}_{h\sim\nu}\big)\big[R(h) - \hat{R}_S(h)\big], \end{aligned} \tag{1.92}$$

where we used $Z_q^{-1} = q(h)e^{\beta R(h)}$ in the third line and concavity of the log in the last line. Eq.(1.92) implies that an upper bound on $KL\big(\mu_S||\nu\big)$ can be derived

[9] Since we are interested in their relative entropy, the reference measure will drop out in the end.

by upper and lower bounding the expectation of $R(h) - \hat{R}_S(h)$ w.r.t. μ_S and ν, respectively. Both bounds can be obtained from Eq.(1.81) when applied to $\mu = \mu_S$ and $\mu = \nu$, respectively. Inserting these bounds into Eq.(1.92) we get

$$KL\big(\mu_S||\nu\big) \leq \frac{\beta}{\sqrt{2n}}\left(\sqrt{KL\big(\mu_S||\nu\big) + \log\left(2\sqrt{n}/\delta\right)} + \sqrt{\log\left(2\sqrt{n}/\delta\right)}\right).$$

The largest value of $KL\big(\mu_S||\nu\big)$ that is consistent with this inequality is obtained by assuming equality and solving the resulting quadratic equation. This leads to Eq.(1.90).

In order to obtain the claimed bound on the mutual information, we use that combining Lem.1.25 with Lem.1.26 leads to $I(h:S) \leq \mathbb{E}_S\big[KL(\mu_S||\nu)\big]$. This holds for an arbitrary probability measure ν and if we choose ν as assumed in the statement of the theorem, we can further bound $KL(\mu_S||\nu)$ via Eq.(1.92). In this way, we obtain

$$I(h:S) \leq \beta\, \mathbb{E}_S\big[R(\mu_S) - \hat{R}_S(\mu_S)\big]. \tag{1.93}$$

Note that the expectation values w.r.t. ν disappeared, due to $\mathbb{E}_S$. From Eq.(1.93) we can obtain the stated bound on the mutual information by, on the one hand, upper bounding the r.h.s. by β, and on the other hand, upper bounding it using Thm.1.21 and solving the resulting quadratic equation. The latter then leads to $I(h:S) \leq \beta^2/(2n)$. Reinserting this into Thm.1.21 finally leads to Eq.(1.91). □

1.13 Differential privacy

... to be written ...

1.14 AdaBoost

Ensemble methods are meta-algorithms that combine several machine learning algorithms or predictors to form a more powerful one. A famous example for the success of ensemble methods is the winner of the $1M Netflix prize in 2009, which substantially improved Netflix' recommender system. All the top submissions in that competition were combinations of combinations of ... hundreds of predictors.

We will describe one of the most common ensemble methods for binary classification, the *AdaBoost* (short for *Ada*ptive *Boost*ing). The starting point of this method is a hypotheses class $\mathcal{F} \subseteq \{-1,1\}^{\mathcal{X}}$ whose elements are called *base hypotheses*. AdaBoost is an iterative method that when stopped after the T'th iteration returns a hypothesis of the form

$$f := \operatorname{sgn}\left(\sum_{t=1}^{T} w_t h_t\right), \quad w_t \in \mathbb{R}, \quad h_t \in \mathcal{F}. \tag{1.94}$$

That is, f is constructed so that its prediction is a weighted majority vote of the predictions of T base hypotheses. Note that $f \notin \mathcal{F}$, unless $\mathcal{F}$ is incidentally closed under such operations. In every iteration of AdaBoost an ERM algorithm for $\mathcal{F}$ is called as a subroutine, which returns one of the h_t's. The key idea is that the empirical risk that is minimized within this subroutine assigns different weights to the training data instances. The algorithm puts more weight on those instances that appear to be hard in the sense that they were misclassified by the previous h_t's. Suppose the training data S consists of n pairs $(x_i, y_i) \in \mathcal{X} \times \{-1, 1\}$. Let $p^{(t)}$ be a yet to be constructed probability distribution over S that is used in the t'th iteration. Define by

$$\epsilon_t := \sum_{i=1}^{n} p_i^{(t)} \mathbb{1}_{h_t(x_i) \neq y_i} \tag{1.95}$$

the $p^{(t)}$-weighted empirical risk of h_t, i.e., the error probability of h_t on S when the entries in S are weighted according to $p^{(t)}$. Given $p^{(t)}$, the hypothesis h_t is ideally chosen so that it minimizes this weighted empirical risk. We will, however, treat the selection of h_t as a black box and do not require that h_t really minimizes the weighted risk. ϵ_t is simply defined as in Eq.(1.95), whether this is optimal or not. From here define

$$w_t := \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right). \tag{1.96}$$

The details of this choice will become clear later. For now, observe that w_t increases with decreasing ϵ_t and that $w_t \geq 0$ whenever $\epsilon_t \leq \frac{1}{2}$, i.e., whenever the hypothesis h_t performs at least as good as random guessing. The update rule for the probability distribution then reads

$$\begin{aligned} p_i^{(t+1)} &:= \frac{p_i^{(t)}}{Z_t} \times \begin{cases} e^{-w_t} & \text{if } h_t(x_i) = y_i \\ e^{w_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= p_i^{(t)} e^{-w_t y_i h_t(x_i)} / Z_t, \end{aligned}$$

where Z_t is a normalization factor chosen so that $\sum_{i=1}^{n} p_i^{(t+1)} = 1$. Note that the step from $p^{(t)}$ to $p^{(t+1)}$ aims at increasing the weight that corresponds to (x_i, y_i) if x_i has been misclassified by h_t (in case h_t performs better than random guessing).

Upon input of the training data S, AdaBoost starts with a uniform distribution $p^{(1)}$ and iterates the above procedure, where in each iteration ϵ_t, w_t, h_t and $p^{(t+1)}$ are computed. The number T of iterations is a free parameter which essentially allows to balance between the estimation error and the approximation error. If the class $\mathcal{F}$ of base hypotheses is simple, then small T may lead to large approximation error, whereas choosing T very large makes it more likely that overly complex hypotheses are returned.

The following theorem shows that the empirical risk can decrease rapidly with increasing T:

Theorem 1.23: Empirical risk bound for AdaBoost

Let f be the hypothesis that is returned after T iterations of AdaBoost that led to intermediate weighted empirical risks $\epsilon \in [0,1]^T$. Then the error probability of f on the training data set is bounded by

$$\hat{R}(f) \leq \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)}. \tag{1.97}$$

With $\gamma := \min\{|\epsilon_t - 1/2|\}_{t=1}^T$ this implies in particular $\hat{R}(f) \leq \exp[-2\gamma^2 T]$.

Proof. Define $F := \sum_{t=1}^T w_t h_t$ and observe that with $p_i^{(1)} = 1/n$ we can write

$$\begin{aligned} p_i^{(T+1)} &= p_i^{(1)} \times \frac{e^{-w_1 y_i h_1(x_i)}}{Z_1} \times \cdots \times \frac{e^{-w_T y_i h_T(x_i)}}{Z_T} \\ &= \frac{e^{-y_i F(x_i)}}{n \prod_{t=1}^T Z_t}. \end{aligned} \tag{1.98}$$

If $f(x_i) \neq y_i$, then $y_i F(x_i) \leq 0$, which implies $e^{-y_i F(x_i)} \geq 1$. Therefore,

$$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{f(x_i) \neq y_i} \leq \frac{1}{n} \sum_{i=1}^n e^{-y_i F(x_i)} = \prod_{t=1}^T Z_t, \tag{1.99}$$

where the last step uses Eq.(1.98) together with the fact that the $p_i^{(T+1)}$'s sum up to 1. Next, we write the normalization factors Z_t in a more suitable form:

$$\begin{aligned} Z_t &= \sum_{i=1}^n p_i^{(t)} e^{-w_t y_i h_t(x_i)} = \sum_{i:h_t(x_i) \neq y_i} p_i^{(t)} e^{w_t} + \sum_{i:h_t(x_i) = y_i} p_i^{(t)} e^{-w_t} \\ &= \epsilon_t e^{w_t} + (1-\epsilon_t) e^{-w_t} = 2\sqrt{\epsilon_t(1-\epsilon_t)}, \end{aligned} \tag{1.100}$$

where we have inserted w_t from Eq.(1.96). This completes the proof of Eq.(1.97). In order to arrive at the second claim of the theorem, we use that $1 - x \leq e^{-x}$ holds for all $x \in \mathbb{R}$, which allows us to bound

$$2\sqrt{\epsilon_t(1-\epsilon_t)} = \sqrt{1 - 4(\epsilon_t - 1/2)^2} \leq \exp\big[-2(\epsilon_t - 1/2)^2\big].$$

□

The proof reveals two more things about AdaBoost. First, we can understand the particular choice of the w_t's. Looking at Eq.(1.100) one is tempted to choose them so that they minimize the expression $\epsilon_t e^{w_t} + (1-\epsilon_t)e^{-w_t}$ and, indeed, this is exactly what the choice in Eq.(1.96) does. Second, notice that after inserting all expressions we obtain

$$\sum_{i:h_t(x_i)=y_i} p_i^{(t+1)} = \sum_{i:h_t(x_i)=y_i} \frac{p_i^{(t)}}{Z_t} e^{-w_t} = (1-\epsilon_t)\frac{e^{-w_t}}{Z_t} = \frac{1}{2}.$$

This means that in every iteration the new probability distribution $p^{(t+1)}$ is chosen so that the correctly classified instances all together get total weight one half (and so do the misclassified ones). Hence, $p^{(t+1)}$ can be computed from $p^{(t)}$ by a simple rescaling of the probabilities of these two sets.

Thm.1.23 shows that if we manage to keep the error probabilities ϵ_t a constant γ away from $1/2$ (the performance of flipping a coin), the empirical risk will decrease exponentially in the number T of iterations. More precisely, it is upper bounded by a decreasing exponential—it does in fact not have to decrease monotonically itself.

In order to get a theoretical bound on the risk, i.e., on the performance beyond the training data, we look at the VC-dimension:

Theorem 1.24: VC-dimension of linearly combined classifiers

Let d be the VC-dimension of $\mathcal{F} \subseteq \{-1,1\}^{\mathcal{X}}$ and for $T \in \mathbb{N}$ define $\mathcal{F}_T := \{f = \operatorname{sgn} \sum_{t=1}^T w_t h_t \mid w_t \in \mathbb{R},\ h_t \in \mathcal{F}\}$. Then the growth function Γ of $\mathcal{F}_T$ satisfies

$$\begin{aligned} \Gamma(n) &\leq \left(\frac{en}{T}\right)^T \left(\frac{en}{d}\right)^{Td} \quad \text{and} && (1.101)\\ \text{VCdim}(\mathcal{F}_T) &\leq 2T(d+1)\log_2\big(2eT(d+1)\big). && (1.102) \end{aligned}$$

Proof. In order to bound the growth function we regard $\mathcal{F}_T = \mathcal{G} \circ \mathcal{H}$ as a composition of two function classes

$$\begin{aligned} \mathcal{G} &:= \Big\{g : \mathbb{R}^T \to \{-1,1\} \mid g(z) = \operatorname{sgn} \sum_{i=1}^T w_i z_i,\ w_i \in \mathbb{R}\Big\},\\ \mathcal{H} &:= \Big\{h : \mathcal{X} \to \mathbb{R}^T \mid h(x) = \big(h_1(x), \ldots, h_T(x)\big),\ h_t \in \mathcal{F}\Big\}. \end{aligned}$$

Following Lemma 1.4 we have $\Gamma(n) \leq \Gamma_{\mathcal{G}}(n)\Gamma_{\mathcal{H}}(n)$ where $\Gamma_{\mathcal{G}}$ and $\Gamma_{\mathcal{H}}$ denote the growth functions of $\mathcal{G}$ and $\mathcal{H}$, respectively. Since the VC-dimension of $\mathcal{G}$ is equal to T by Thm.1.9, we can apply Thm.1.8 and obtain $\Gamma_{\mathcal{G}}(n) \leq (en/T)^T$. The product structure of $\mathcal{H}$ implies that $\Gamma_{\mathcal{H}}(n) = \Gamma_{\mathcal{F}}(n)^T$ where $\Gamma_{\mathcal{F}}$ denotes the growth function of $\mathcal{F}$. The latter can by Thm.1.8 be bounded in terms of the VC-dimension so that $\Gamma_{\mathcal{F}}(n) \leq (en/d)^d$. Collecting the terms this finally leads to Eq.(1.101).

In order to arrive at a bound for the VC-dimension, note that $D \geq \text{VCdim}(\mathcal{F}_T)$ if $2^D > \Gamma(D)$. Inserting the upper bound on the growth function from Eq.(1.101) this is implied by

$$D > T(d+1)\log_2(eD) - T\log_2 T - dT\log_2 d.$$

Straight forward calculation shows that this is satisfied, if we choose D equal to the r.h.s. of Eq.(1.102). □

Comparing the scaling of this bound for the VC-dimension with the one of the empirical risk in Thm.1.23 is already promising: while the VC-dimension grows not much faster than linearly with T, the empirical risk ideally decreases exponentially. In practice, AdaBoost has been observed to be remarkably resistant against overfitting as long as the data is not too noisy.

From a purely theoretical perspective, AdaBoost shows that a priori different notions of learnability coincide. Consider a weak and a strong notion of learnability, where the strong notion is the one we discussed in the context of PAC learnability. This requires something for all $\epsilon \in (0, 1]$ where the weak notion would only ask for $\epsilon \in (1/2 - \gamma, 1]$ for some fixed, possibly small $\gamma > 0$. Then AdaBoost can be used to 'boost' learnability from weak to strong and to show that these two notions actually coincide.

Chapter 2

Neural networks

2.1 Information processing in the brain

The human brain contains about 10^{11} *neurons*, which can be regarded as its basic information processing units. A typical neuron consist of a *cell body*, *dendrites*, which receive incoming signals from other neurons and an *axon*, which transmits signals to other neurons. While there are typically several dendrites originating from the cell body and then branching out in the neighborhood of the neuron, there is only one axon, which may have a local branching in the neighborhood of the cell body and a second branching at a distance. This can mean everything from 0.1mm to 2m.

On the macroscopic scale, if we regard the human brain as a whole, we see it covered with a folded outer layer, which is about 3mm thick (when unfolded), and called the *cerebral cortex.* The largest part of the cerebral cortex is also its evolutionary youngest part and for this reason called *neocortex.* The neocortex plays a crucial role in many higher brain functions.

If we look at slices of the brain, we see the cerebral cortex as *gray matter* clearly separated from the *white matter*, which it surrounds. White matter almost exclusively consists of axons that connect more distant parts of the brain. The axons originate from neurons (mainly so-called pyramidal neurons, named after their shape), which are part of the gray matter, then leave the gray matter, traverse parts of the brain in the white matter, which is formed by them, and then reenter the gray matter and connect to other neurons. In this sense, white matter is related to (long distance) communication, whereas information storage and processing happens in the gray matter. The difference in color stems from the myelin, a fatty white substance which covers the axons in the white matter. The main purpose of the myelin sheaths is to increase the speed at which signals travel down the axons. Therefore, only the long distance connections are covered with myelin.

The gray matter exhibits horizontal as well as vertical substructures. In most regions, six horizontal layers can be identified that distinguished depending on the occurring neuronal cell types and/or the types of connections with other regions.[1] There are also vertical units, called *cortical (mini)collumns*, which are sometimes regarded as the basic functional units (or elementary pattern recognizers) in the brain. However, depending on the region of the brain, this viewpoint is subject of debate.

A typical pyramidal neuron in the neocortex forms a highly connected local network in the gray matter where it is connected to about 10^4 of its neighbors that are less then 1mm apart. In addition, via the axon traversing the white matter, the neuron is connected to a similar number of distant neurons. There is evidence that connections between different regions of the neocortex are typically two-way connections in the sense of region-to-region, but usually not point-to-point.

The neocortex is very homogeneous throughout the brain so that different functions that are assigned to different areas are not obviously reflected physiologically. The assignment of special functions to specific areas clearly depends on which parts or sensory inputs the area is connected to.

The signals between neurons are electrical pulses that originate in a change of the electrical potential of in total about 100mV—the *action potential*. Such a pulse takes about 1ms and travels down the axon where it reaches so-called *synapses* at the axon's branches. A synapse connects the axon of one neuron with the dendrite of another neuron. The signal transmission within most synapses is of chemical nature. The arriving electrical pulse induces a chemical process inside the synapse, which in turn leads to a change of electrical potential in the postsynaptic neuron. The time it takes for a signal to pass a chemical synapse is around 1ms.

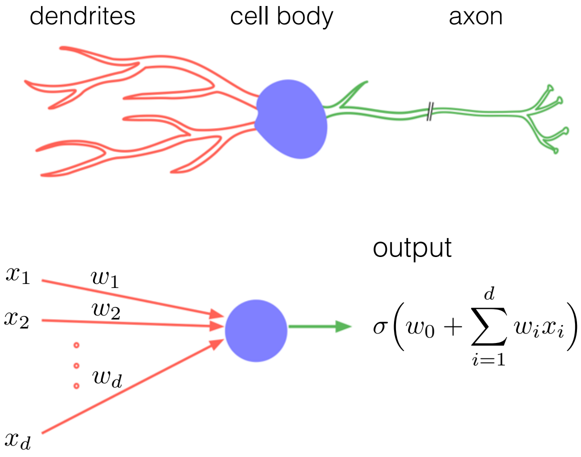
In the dendritic part of the postsynaptic neuron all the incoming signals are integrated. If they lead to a joint stimulus above a certain threshold, they will cause an action potential and the neuron will fire. This is an all-or-nothing process and all stimuli above threshold lead to the same pulse with standardized amplitude and duration. While the outgoing signal in this way can be considered digital, the integration/accumulation of the incoming signals appears to be more of analog nature.

The effect of an incoming pulse on the postsynaptical neuron can vary significantly—over time, in duration and in strength. It may change the potential from milliseconds to minutes and during this period have an excitatory or an inhibitory effect.[2] The variability of the strength of this effect is considered crucial for purposes of learning and memorization.

A neuron can fire several hundred times per second (while the average firing rate is closer to 1Hz). A limiting factor to higher rates is the duration of each pulse and a corresponding *refractory period* of about 1ms after initiation of an action potential during which no stimulus can lead to firing. Within about 4ms after this strict refractory period stimuli still have to be stronger than usual to

[1] These layers, however, are very different from the layers we will encounter later on.

[2] However, connections between pyramidal neurons, which are believed to correspond to the majority of synapses in the cerebral cortex, are exclusively excitatory.



lead to an action potential. This period is called *relative refractory period.*

Everything said in this section merely describes (in a nutshell) the basic structure and physiology that is thought to be relevant for information processing in the brain. How memory and learning actually work on the basis of the described pieces, is much less understood and has to be left aside here.

Let us finally make a rough comparison between the human brain and present day computers in terms of the basic numbers. The power consumption of the brain is around 20 Watts and thus about the same as the one of a present day laptop. Also the estimated number of neurons (10^{11}) is not too far from the number of transistors, which is $10^9 - 10^{10}$ on a state-of-the-art chip. Significant differences lie in the connectivity, the frequency and the related logical depth. A transistor is only directly connected to a few others, it runs with a clock rate of several GHz and can be involved in computations of enormous logical depth. A neuron in comparison is connected to 10^4 or more others, but operates at frequencies of only a few hundred Hz, which is a factor of 10^7 below the computers clock rates. Since most 'computations' in the brain are nevertheless done within a fraction of a second, they cannot have logical depth significantly beyond 100.

Other noticeable differences between computers and brains are that while the former work deterministically, use universal clock cycles and are still essentially 2D structures, the latter appear to be of stochastic nature, work without universal clock cycles and make significant better use of the third dimension.

2.2 From Perceptrons to networks

Artifical neurons A simple artificial neuron model that incorporates some of the properties of biological neurons described in the last section is the *Perceptron*, introduced by Rosenblatt in 1958. More specifically, the Perceptron incorporates (i) several inputs whose effects are determined by variable weights, (ii) a single output (which may, however, be copied/fanned out an arbitrary number of times), (iii) integration of input signals and (iv) an all-or-nothing process with adjustable threshold.

Mathematically, each input is characterized by a real number x_i where $i = 1, \ldots, d$ runs over the number of inputs. Each of the input lines gets assigned a weight $w_i \in \mathbb{R}$. The mapping from the inputs to the output is then modeled by

$$x \mapsto \sigma\left(w_0 + \sum_{i=1}^{d} w_i x_i\right), \tag{2.1}$$

where $w_0 \in \mathbb{R}$ plays the role of a threshold value and the *activation function* $\sigma : \mathbb{R} \to \mathbb{R}$ is, in the case of the Perceptron, given by the step-function $\sigma(z) = \mathbb{1}_{z \geq 0}$. It is convenient to regard w_0 as the weight of a constant input $x_0 = 1$. Nowadays, one usually considers generalizations of this model that differ from the original Perceptron in the choice of the activation function. The main reason for choosing different activation functions is that they enable gradient descent techniques for learning algorithms. Common choices are:

- *Logistic sigmoid*[3] $\sigma(z) = \frac{1}{1+e^{-z}}$,
- *Hyperbolic tangent* $\sigma(z) = \tanh(z)$ (which is an affinely transformed logistic sigmoid),
- *Rectified linear* $\sigma(z) = \max\{0, z\}$. This is sometimes modified to $\max\{\alpha x, x\}$ with some $\alpha \in (0, 1)$.

Note that the logistic sigmoid function and the hyperbolic tangent are smooth versions of the step functions $\mathbb{1}_{z \geq 0}$ and $\operatorname{sgn}(z)$, respectively. In practice, the logistic sigmoid and tanh, which were used over many years, are nowadays more and more replaced by rectified linear activation functions.

A multivariate function of the form $\mathbb{R}^d \ni x \mapsto \sigma(w \cdot x)$ with $\sigma : \mathbb{R} \to \mathbb{R}$ is often called a *ridge function* - especially in the context of approximation theory. Note that every ridge function is constant on hyperplanes characterized by $w \cdot x = c$. In the case of the Perceptron with $\sigma(z) = \mathbb{1}_{z \geq 0}$ we have that the set of points in $\mathbb{R}^d$ that are mapped to 1 forms a closed half space. This relation between Perceptrons and half spaces is obviously bijective and often provides a useful geometric depiction.

Neural networks Composing several artificial neurons of the type just introduced by making use of the possibility to fan out their outputs we obtain a *neural network*. This is then described by a weighted directed graph $G = (V, E)$ where vertices correspond to single neurons or input/output nodes, directions mark the flow of signals and the weights are the w_i's assigned to every individual neuron's inputs. For a weighted directed graph to represent a neural

[3] "Sigmoidal" just means S-shaped.

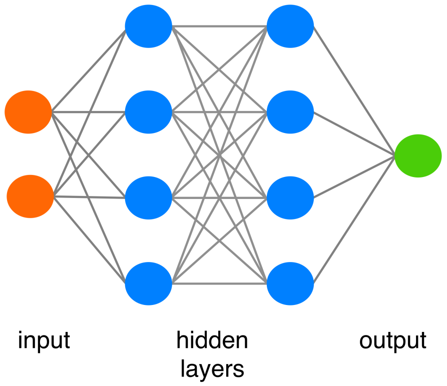


Figure 2.1: Structure of a feedforward neural network with two inputs, two hidden layers with four neurons each and a single output neuron.

network in the usual sense, there need to be input and output nodes, which are usually vertices with only outgoing and only incoming edges, respectively. In addition, we have to assign an individual threshold value w_0 to every neuron and choose an activation function σ. The latter is often chosen equally for all hidden neurons. The activation function of the output neuron(s) is sometimes chosen differently or simply omitted. The simple reason for a different choice at the output is to match the desired range, e.g. $\mathbb{R}$ in case of regression or a particular discrete set in case of classification.

The graph underlying a neural network is called the network's *architecture*, which then neglects the values of weights and thresholds. If the graph is an *acyclic* directed graph, meaning it does not contain directed cycles, then the network is called a *feedforward* network. Otherwise, it is called a *recurrent* network. A particular class of feedforward neural networks are *layered* (a.k.a. multilayered) feedforward neural networks. In this case the vertices $V = \bigcup_{l=0}^m V_l$ are arranged into disjoint *layers* $\{V_l\}_{l=0}^m$ so that connections only exist between neighboring layers, i.e., $E \subseteq \bigcup_{l=0}^{m-1} \{(u,v) | u \in V_l, v \in V_{l+1}\}$. m is then called the *depth* of the network and $m-1$ is the number of *hidden layers* ("hidden" in the sense of in between input and output). In the following we will focus on multilayered feedforward networks. If there is no further specification, we will always assume that neighboring layers are *fully connected*, i.e., every element of V_l is connected to every element in V_{l+1}.

Other models ... to be written ...

2.3 Representation and approximation

A neural network with d inputs and d' outputs represents a function $f : \mathbb{R}^d \to \mathbb{R}^{d'}$. In this section we address the question which functions can be represented (exactly or approximately) by a multilayered feedforward neural network, depending on the architecture and on the chosen activation function. We will start with the discrete case.

Representation of Boolean functions As a warm-up, consider a single Perceptron with two Boolean inputs. Can it represent basic Boolean functions like AND, OR, NAND or XOR? The simple but crucial observation for answering this question is that the function $f_w : \mathbb{R}^d \to \mathbb{R}$ in Eq.(2.1) that describes a Perceptron is, for every choice of the weights $w \in \mathbb{R}^{d+1}$, constant on hyperplanes that are orthogonal to the vector $(w_1, \ldots, w_d)$. Moreover, due to the special choice of σ as a step function f_w separates half-spaces and by choosing suitable weights any half-space can be separated from its complement.

If we regard the inputs of AND, OR or NAND as points in $\mathbb{R}^2$, then in all three cases the subsets that are mapped to 0 or 1 can be separated from each other by a line. Consequently, AND, OR and NAND can be represented by a single Perceptron. This is already somewhat promising since we know that every Boolean function can be obtained as a composition of many of such building blocks. XOR, on the other hand, cannot be represented by a single Perceptron since in this case the inputs that are mapped to 0 cannot be linearly separated from the ones mapped to 1. This implies that representing an arbitrary Boolean function by a feedforward neural network requires at least one hidden layer. The following theorem shows that a single hidden layer is already sufficient.

Theorem 2.1: Representation of Boolean functions

Every Boolean function $f : \{0,1\}^d \to \{0,1\}$ can be represented exactly by a feedforward neural network with a single hidden layer containing at most 2^d neurons, if $\sigma(z) = \mathbb{1}_{z \geq 0}$ is used as activation function.

Proof. If $a, b \in \{0,1\}$ are Boolean variables, then $2ab - a - b \leq 0$ with equality iff $a = b$. With this observation we can write $\mathbb{1}_{x=u} = \sigma\big(\sum_{i=1}^d 2x_iu_i - x_i - u_i\big)$ for $x, u \in \{0,1\}^d$. Denoting by $A := f^{-1}\big(\{1\}\big)$ the set of all vectors u for which $f(u) = 1$, we can then represent f as

$$f(x) = \sigma\Big(-1 + \sum_{u \in A} \mathbb{1}_{x=u}\Big) = \sigma\Big(-1 + \sum_{u \in A} \sigma\big(\sum_{i=1}^d 2x_iu_i - x_i - u_i\big)\Big), \quad (2.2)$$

which is the sought representation using a single hidden layer with $|A| \leq 2^d$ neurons. □

We will see in Sec.2.4 that the exponential increase of the number of neurons cannot be avoided.

Binary classification in $\mathbb{R}^d$:

Theorem 2.2: Binary classification of finite sets in $\mathbb{R}^d$

Let $A = \{x_1, \ldots, x_N\}$ be a finite subset of $\mathbb{R}^d$ and $f : A \to \{-1, 1\}$ arbitrary. There is a feedforward neural network that implements a function $F : \mathbb{R}^d \to \{-1, 1\}$ with a single hidden layer containing $m \leq N$ neurons and using $\sigma = \operatorname{sgn}$ as activation function so that $F|_A = f$. If the points in A are in general position (i.e., no hyperplane in $\mathbb{R}^d$ contains more than d of them), then $m \leq 2\lceil N/(2d) \rceil$ neurons suffice.

Proof. Denote by A_+ and A_- the subsets of A that are mapped to 1 and -1, respectively. W.l.o.g. assume that $|A_+| \leq |A_-|$ so that $|A_+| \leq N/2$. For every $x \in A_+$ we can find a hyperplane $H := \{z \in \mathbb{R}^d | a \cdot z + b = 0\}$ characterized by $a \in \mathbb{R}^d$ and $b \in \mathbb{R}$ so that $A \cap H = \{x\}$. Due to finiteness of A we can now find two hyperplanes that are parallel to H, contain H and thus x in between them, but none of the other points from A. In other words, we can choose $\epsilon \neq 0$ appropriately, so that the map $z \mapsto \sigma(\epsilon + a \cdot z + b) + \sigma(\epsilon - a \cdot z - b)$ takes on the value 2 for $z = x$ but is zero on $A \setminus \{x\}$. Repeating this for all points in A_+ we can finally construct

$$F(z) := \sigma\left(-1 + \sum_{x \in A_+} \sigma\big(\epsilon_x + a_x \cdot z + b_x\big) + \sigma\big(\epsilon_x - a_x \cdot z - b_x\big)\right), \qquad (2.3)$$

so that $F|_A = f$. Then F has the form of a neural network with a single hidden layer that contains $m = 2|A_+| \leq N$ neurons.

Now assume the points in A are in general position. Then we can in every (but the last) step of the construction choose the hyperplane H so that it contains d points from A_+ and no other point from A. In this way, we reduce the number of terms essentially by a factor d and we get $m \leq 2\lceil N/(2d) \rceil$. □

Let us consider binary classification of subsets of $\mathbb{R}^d$ via neural networks from a more geometric point of view. Consider a network with a single hidden layer with m neurons and $\sigma(z) = \mathbb{1}_{z \geq 0}$ as activation function. As mentioned before, every individual Perceptron can be characterized by a half space, say H_j for the j'th hidden neuron, in such a way that the output of the Perceptron upon input x is given by the value of the indicator function $\mathbb{1}_{x \in H_j}$. In this way we can write the function $f : \mathbb{R}^d \to \{0, 1\}$ that is implemented by the network as

$$f(x) = \sigma\Big(w_0 + \sum_{j=1}^{m} w_j \mathbb{1}_{x \in H_j}\Big).$$

Defining by $\mathcal{A} := \{A \subseteq \{1, \ldots, m\} \mid \sum_{j \in A} w_j \geq -w_0\}$ the set of all subsets of hidden neurons that are capable of activating the output neuron by firing together, we can write

$$f^{-1}\big(\{1\}\big) = \bigcup_{A \in \mathcal{A}} \bigcap_{j \in A} H_j. \qquad (2.4)$$

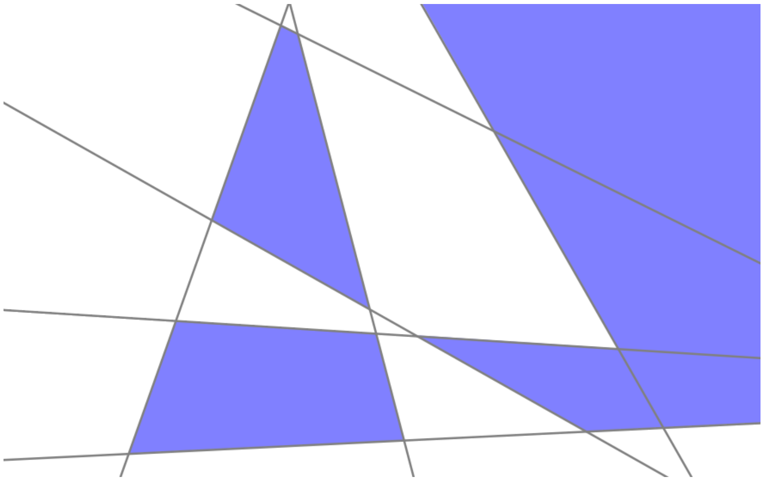


Figure 2.2: The subsets of $\mathbb{R}^d$ that are classified by a neural network with a single hidden layer containing m hidden neurons with $\sigma(z) = \mathbb{1}_{z\geq 0}$ are unions of convex polyhedra that are obtained as intersections of subsets of m half spaces. Here $m = 7$, the hyperplanes that delimit the half spaces are gray lines and the blue regions exemplify $f^{-1}(\{1\})$.

Note that $\bigcap_{j\in A} H_j$ is, as an intersection of at most m half spaces, a convex polyhedron with at most m facets. Hence, the set of points that are mapped to 1 by the network is given by a union of convex polyhedra that are obtained as intersections of some of m half spaces (see Fig.2.2).

It should be mentioned that this picture can change considerably if the activation function is changed. As we have just seen, the geometry of binary classification with m neuron's with a step activation function in a single hidden layer is determined by m hyperplanes. However, if we replace the step function with a rectified linear function $\sigma(z) := \max\{0, z\}$, a much larger number of hyperplanes can occur. As a consequence, some functions that have a simple representation when using rectified linear units are difficult to represent using step activation functions:

Proposition 2.1. *A representation of the function* $f : \mathbb{R}^d \mapsto \{-1, 1\}$, $f(x) := \operatorname{sgn}\big[-1 + \sum_{i=1}^d \max\{0, x_i\}\big]$ *in terms of a neural network with a single hidden layer that contains* m *neurons and uses* $\sigma = \operatorname{sgn}$ *exists only if* $m \geq 2^{(d-1)}$.

Remark: clearly, the function can be represented using rectified linear activation functions in a single hidden layer of only $m = d$ neurons.

Proof. The region $f^{-1}(\{1\})$ is the intersection of all half spaces of the form $H_A := \{x \in \mathbb{R}^d | \sum_{i\in A} x_i \geq 1\}$ where A is any non-empty subset of $\{1, \ldots, d\}$. Since there are $2^{(d-1)}$ such sets, $f^{-1}(\{1\})$ could have this number of facets, which would indeed imply that $m \geq 2^{(d-1)}$ as for $\sigma = \operatorname{sgn}$ the number of hyperplanes is at most the number of neurons in the hidden layer. So, it remains to show that none of these half spaces is redundant in the characterization of $f^{-1}(\{1\})$. This is done by constructing a point x for every nonempty $A \subseteq \{1,\ldots,d\}$ so that $x \in H_{A'} \Rightarrow A' = A$. A possible choice for such a point is

$$x_i = \begin{cases} \frac{1}{|A|} & , \; i \in A \\ -1 & , \; i \notin A \end{cases}$$

□

Let us conclude the geometric discussion with a classical theorem that provides a tight bound on the number of regions into which $\mathbb{R}^d$ is cut by n hyperplanes. For its proof we again use the relation between VC-dimension and 'cell counting' that has led to Thm.1.10, but now in the opposite direction.

Theorem 2.3: Zaslavsky's theorem

Let $h_1,\ldots,h_n \subseteq \mathbb{R}^d$ be hyperplanes. The number N of connected components of $\mathbb{R}^d \setminus \bigcup_{i=1}^n h_i$ then satisfies

$$N \leq \sum_{i=0}^{d} \binom{n}{i} \leq \left(\frac{en}{d}\right)^d . \tag{2.5}$$

Remark: the first inequality can be shown to be tight. A generic collection of n hyperplanes turns out to cut $\mathbb{R}^d$ into exactly $\sum_{i=0}^{d}\binom{n}{i}$ regions.

Proof. (sketch) By translation we can always ensure that none of the considered hyperplanes goes through the origin. Under this assumption, every hyperplane can be characterized by a vector $w \in \mathbb{R}^d$ via $h := \{x \in \mathbb{R}^d | \langle x, w\rangle = 1\}$. Denote by $\mathcal{X}$ the set of all these hyperplanes in $\mathbb{R}^d$, and define a set of functions $\mathcal{F} \subseteq \{-1,1\}^{\mathcal{X}}$ via $\mathcal{F} := \{h \mapsto \text{sgn}[g(h)] \mid g \in \mathcal{G}\}$, where $\mathcal{G} := \{h \mapsto \langle x, w\rangle - 1 \mid x \in \mathbb{R}^d\}$ with w being the vector characterizing h. Every function in $\mathcal{F}$ then corresponds to a single point in $\mathbb{R}^d$ and assigns a value ±1 to a hyperplane, depending on whether the point lies on one side of the plane or on the other. As $\mathcal{G} \simeq \mathbb{R}^d$ forms a d-dimensional affine space (a small modification of) Thm.1.9 implies that $VCdim(\mathcal{F}) = d$.

Assume now that a collection $A \subseteq \mathcal{X}$ of n hyperplanes separate N regions in $\mathbb{R}^d$. That is, there are N points in $\mathbb{R}^d$ so that for any pair of them, there is one separating hyperplane among the n considered ones. In other words, $\mathcal{F}|_A$ contains at least N different functions and since $|A| = n$ this implies that $N \leq \Gamma(n)$, where Γ denotes the growth function of $\mathcal{F}$. Using Thm.1.8 we can bound the growth function in terms of the VC-dimension, and thus d, in precisely the way stated in Eq.(2.5). □

Representing real-valued functions on finite sets Now we move to real-valued functions and first consider the case of finite domains. The following theorem shows that the number of parameters required in a feedforward neural network in order to exactly represent an arbitrary assignment of real numbers to N points in $\mathbb{R}^d$ scales linearly with N. The network that implements the function involves *weight sharing* in the sense that the weights of the edges between the d inputs and the neurons of the hidden layer do not depend on the neuron.

Theorem 2.4: Representation with few parameters

Let $A = \{x_1, \ldots, x_N\}$ be a finite subset of $\mathbb{R}^d$ and $f : A \to \mathbb{R}$ arbitrary. There is a feedforward neural network that implements a function $F : \mathbb{R}^d \to \mathbb{R}$ with a single hidden layer containing N neurons and $2N + d$ parameters so that $F|_A = f$. The network can be chosen such that it uses the ReLU activation function $\sigma(x) = \max\{0, x\}$ in the hidden layer and the identity as activation function at the output.

Note: The additional '$+d$' parameters depend only weakly on A and f. As will be seen in the proof, a random choice will do the job with probability one, so that there are really only $2N$ essential parameters. These correspond to the weights of the output neuron and the biases/threshold values of the neurons in the hidden layer.

Proof. The function $F : \mathbb{R}^d \to \mathbb{R}$,

$$F(x) := \sum_{j=1}^{N} a_j \max\{0, w \cdot x - v_j\}$$

has $2N + d$ parameters given by $w \in \mathbb{R}^d, a, v \in \mathbb{R}^N$ and it can be implemented using a neural network of the specified type. We have to show that there is always a choice of parameters that solves the set of equations $F(x_i) = f(x_i)$, $i = 1, \ldots, N$. Such a solution exists, if the $N \times N$ matrix M with entries $M_{i,j} := \max\{0, w \cdot x_i - v_j\}$ is invertible, since we can then simply choose $a_j := \sum_i M^{-1}_{j,i} f(x_i)$. W.l.o.g. we can assume that A contains N distinct elements. Then there exists a $w \in \mathbb{R}^d$ so that all $z_i := w \cdot x_i$ are distinct, as well. Assuming in addition that the z_i's are in increasing order (and otherwise relabeling them accordingly) we can choose v so that $v_1 < z_1 < v_2 < z_2 < \ldots$. In this way M becomes a triangular matrix with non-vanishing diagonal entries and is thus invertible. □

Approximating real-valued functions We will begin with the one-dimensional case $f : \mathbb{R} \to \mathbb{R}$ and later lift the obtained results to the case of higher dimensional input and output spaces. Let us denote by $\mathcal{F}_{\sigma,m}$ the class of functions representable by a feedforward network with a single hidden layer with m neurons in the hidden layer. That is,

$$\mathcal{F}_{\sigma,m} := \Big\{ f : \mathbb{R} \to \mathbb{R} \;\Big|\; f(x) = \sum_{v=1}^{m} a_v \sigma(w_v x + b_v),\ a_v, b_v, w_v \in \mathbb{R} \Big\} \tag{2.6}$$

and define $\mathcal{F}_\sigma := \bigcup_{m\in\mathbb{N}} \mathcal{F}_{\sigma,m}$. Note that there is no activation function (or, equivalently, just the identity) at the output.

The following approximation theorem is formulated in terms of the *modulus of continuity* of the function f to be approximated. It is defined as

$$\omega(f,\delta) := \sup_{x,y:|x-y|\leq\delta} \big|f(x)-f(y)\big|. \tag{2.7}$$

Note that if f is continuous, then $\omega(f,\delta) \to 0$ for $\delta \to 0$. Moreover, if f is $L-$Lipschitz, then $\omega(f,\delta) \leq \delta L$.

Theorem 2.5: Approximations using bounded sigmoids

Let $\sigma : \mathbb{R} \to \mathbb{R}$ be any bounded function that satisfies $\lim_{z\to-\infty} \sigma(z) = 0$ and $\lim_{z\to\infty} \sigma(z) = 1$. There is a constant c so that for every $f \in C\big([0,1]\big)$ and every $m \in \mathbb{N}$ we have

$$\inf_{f_m\in\mathcal{F}_{\sigma,m}} ||f - f_m||_\infty \leq c\,\omega(f, 1/m). \tag{2.8}$$

Note: From the proof we get that $c = 2 + 2||\sigma||_\infty$ is a valid constant (where $||\sigma||_\infty = \sup_{z\in\mathbb{R}} |\sigma(z)|$). A more careful analysis shows that $c = ||\sigma||_\infty$ suffices. The assumption that σ is asymptotically either 0 or 1 is convenient for the proof but not really crucial. The same argument works if only the limits $\lim_{z\to\pm\infty} \sigma(z)$ exist in $\mathbb{R}$ and differ from each other. Similarly, the domain $[0,1]$ can be replaced by any compact subset of $\mathbb{R}$.

Proof. The idea is to first approximate f by a piecewise constant function h_m and then h_m by an appropriate f_m. Define $x_i := i/m$ and $h_m(x)$ so that it takes the value $f(x_i)$ in the interval $x \in \big[x_{i-1}, x_i\big)$ where $i = 1,\ldots,m$. By construction $||f - h_m||_\infty \leq \omega(f, 1/m)$. With $j := \lfloor mx \rfloor$ write

$$\begin{aligned} h_m(x) &= f(x_1) + \sum_{i=1}^{j} \big(f(x_{i+1}) - f(x_i)\big) \quad \text{and define} \\ f_m(x) &:= f(x_1)\sigma(\alpha) + \sum_{i=1}^{m-1} \big(f(x_{i+1}) - f(x_i)\big)\sigma\big(\alpha(mx-i)\big), \end{aligned} \tag{2.9}$$

for some $\alpha \in \mathbb{R}$ to be chosen shortly. Note that f_m is of the desired form. The claim is that f_m approximates h_m well for large α. To bound the distance between the two functions, fix any $\epsilon > 0$ and choose α such that $|\sigma(z) - \mathbb{1}_{z\geq 0}| \leq \epsilon/m$ whenever $|z| \geq \alpha$. This is possible since σ is assumed to be a sigmoidal function. Note that, by the choice of α, we get that if $i \notin \{j, j+1\}$ the term $\sigma\big(\alpha(mx-i)\big)$ is ϵ/m-close to the step function $\mathbb{1}_{i\leq\lfloor mx\rfloor}$. Consequently, we can bound

$$\begin{aligned} |f_m(x) - h_m(x)| &\leq \frac{\epsilon}{m}\Big[|f(x_1)| + (m-2)\omega(f,1/m)\Big] \\ &\quad + \big|f(x_{j+1}) - f(x_j)\big| \, \Big|1 - \sigma\big(\alpha(mx-j)\big)\Big| \\ &\quad + \big|f(x_{j+2}) - f(x_{j+1})\big| \, \Big|\sigma\big(\alpha(mx-j-1)\big)\Big|, \end{aligned}$$

where the r.h.s. of the first line can be made arbitrary small by the choice of ϵ and the sum of the last two lines can be bounded by $\omega(f,1/m)(1+2||\sigma||_\infty)$. □

As a consequence, an arbitrary L-Lipschitz function can be approximated uniformly by a feedforward network with a single hidden layer of m neurons so that the approximation error is $\mathcal{O}(L/m)$. The following proposition characterizes the class of continuous activation functions for which similar approximation results can be obtained.

Proposition 2.2 (Universality of all non-polynomial activation functions)**.** *Let $\sigma \in C(\mathbb{R})$. The set of functions $\mathcal{F}_\sigma$ representable by a neural network with a single hidden layer and activation function σ is dense in $C(\mathbb{R})$ w.r.t. the topology of uniform convergence on compacta iff σ is not a polynomial.*

Proof. (sketch) Suppose σ is a polynomial of degree k. Since these form a closed set under linear combinations, $\mathcal{F}_\sigma$ will still only contain polynomials of degree at most k and thus cannot be dense in $C(\mathbb{R})$.

For the converse direction we will restrict ourselves to the case $\sigma \in C^\infty(\mathbb{R})$. The extension of the argument from $C^\infty(\mathbb{R})$ to $C(\mathbb{R})$ can be found in [20]. It is known (for instance as a non-trivial consequence of Baire's category theorem, cf. [8]) that for any non-polynomial $\sigma \in C^\infty(\mathbb{R})$ there is a point z such that $\sigma^{(k)}(z) \neq 0$ for all $k \in \mathbb{N}$. Since $\big[\sigma((\lambda+\delta)x+z) - \sigma(\lambda x + z)\big]/\delta$ represents a function in $\mathcal{F}_\sigma$ for all $\delta \neq 0$, we get that

$$\frac{d}{d\lambda}\sigma(\lambda x + z)\Big|_{\lambda=0} = x\,\sigma^{(1)}(z), \tag{2.10}$$

as a function of x, is contained in the closure of $\mathcal{F}_\sigma$. Similarly, by taking higher derivatives, we can argue that $x \mapsto x^k \sigma^{(k)}(z)$ is in the closure of $\mathcal{F}_\sigma$. Since all derivatives of σ are non-zero at z, all monomials and therefore all polynomials are contained in the closure of $\mathcal{F}_\sigma$. As these are dense in $C(\mathbb{R})$, by Weierstrass' theorem, so is $\mathcal{F}_\sigma$. □

Note that if σ is polynomial, then an increasing number of layers in the network will lead to polynomials of increasing degree and thus enable better and better approximations of an arbitrary continuous function. The foregoing discussion shows that if σ is non-polynomial, then depth of the network can be traded for width.

Lemma 2.3 (Approximation by exponentials)**.** *Let $K \subseteq \mathbb{R}^d$ be compact. Then $\mathcal{E} := \text{span}\{f : K \to \mathbb{R} \mid f(x) = \exp\sum_{i=1}^d w_i x_i,\ w_i \in \mathbb{R}\}$ is dense in $\big(C(K), ||\cdot||_\infty\big)$.*

Proof. This is an immediate consequence of the Stone-Weierstrass theorem, which says that $\mathcal{E}$ is dense if (i) $\mathcal{E}$ forms an algebra (i.e., it is closed under multiplication and linear combination), (ii) $\mathcal{E}$ contains a non-zero constant function and (iii) for every distinct pair $x, y \in K$ there is an $f \in \mathcal{E}$ so that $f(x) \neq f(y)$. Here (i) holds by the property of the exponential and the construction of $\mathcal{E}$ as linear span, (ii) holds since $1 \in \mathcal{E}$ and (iii) holds since for $w := (x - y)$ we get $e^{w \cdot x} \neq e^{w \cdot y}$. □

Theorem 2.6: Approximation of multivariate functions

Let $d, d' \in \mathbb{N}$, $K \subseteq \mathbb{R}^d$ be compact and $\sigma : \mathbb{R} \to \mathbb{R}$ any activation function that is (i) continuous and non-polynomial or (ii) bounded and so that the limits $\lim_{z \to \pm\infty} \sigma(z)$ exist in $\mathbb{R}$ and differ from each other. Then the set of functions representable by a feedforward neural network with a single hidden layer of neurons with activation function σ is dense in the space of continuous functions $f : K \to \mathbb{R}^{d'}$ in the topology of uniform convergence.

Note: A norm inducing the topology of uniform convergence would be $||f|| := ||\big(||f_i||_\infty\big)_{i=1}^{d'}||'$ where $|| \cdot ||'$ is an arbitrary norm on $\mathbb{R}^{d'}$.

Proof. First note that it suffices to show the result for $d' = 1$ by considering the d' components in $\big(f_1(x), \ldots, f_{d'}(x)\big) = f(x)$ separately. If each of the f_i's can be approximated up to ϵ using m neurons in the hidden layer, then the same order of approximation is obtained for f with md' neurons just by stacking the d' hidden layers on top of each other.

In order to prove the statement for the case $f : K \to \mathbb{R}$ we first approximate f by exponentials and then the exponentials by linear combinations of activation functions. According to Lemma 2.3 for every $\epsilon > 0$ there is a $k \in \mathbb{N}$, a set of vectors $v_1, \ldots, v_k \in \mathbb{R}^d$ and signs $s \in \{-1, 1\}^k$ so that $g : \mathbb{R}^d \to \mathbb{R}$, $g(x) := \sum_{i=1}^k s_i e^{v_i \cdot x}$ satisfies $||f - g||_\infty \leq \epsilon/2$.

Define $K_1 := \bigcup_{i=1}^k \{v_i \cdot x \mid x \in K\}$ and note that $K_1 \subseteq \mathbb{R}$ is compact. Following Thm.2.5 and Prop.2.2 there is an $l \in \mathbb{N}$ and a set of real numbers a_j, w_j, b_j so that $\sup_{y \in K_1} \left| e^y - \sum_{j=1}^l a_j \sigma(w_j y - b_j) \right| \leq \epsilon/(2k)$. Combining the two approximations we obtain

$$\begin{aligned} & \Big|\Big| f - \sum_{j=1}^l \sum_{i=1}^k s_i a_j \sigma(w_j \; v_i \cdot x - b_j) \Big|\Big|_\infty \\ \leq \quad & \Big|\Big| f - \sum_{i=1}^k s_i e^{v_i \cdot x} \Big|\Big|_\infty + \sum_{i=1}^k \Big|\Big| e^y - \sum_{j=1}^l a_j \sigma(w_j y - b_j) \Big|\Big|_\infty \leq \epsilon, \end{aligned}$$

where the sup-norms are understood as $\sup_{x \in K}$ and $\sup_{y \in K_1}$, respectively. □

Let us finally make a remark, primarily of historical interest, concerning the relation of the above discussion to Kolmogorov's solution of Hilbert's 13th problem. Hilbert conjectured that a solution of the general equation of degree seven cannot be expressed as a finite superposition of continuous functions of two variables. In 1957 Kolmogorov and his student Arnold disproved this conjecture by showing that every continuous multivariate function can even be represented as a finite superposition of continuous functions of only one variable. This eventually led to the following:

Proposition 2.4 (Kolmogorov's superposition theorem)**.** *For every* $n \in \mathbb{N}$ *there exist functions* $\varphi_j \in C\big([0,1]\big)$*,* $j = 0, \ldots, 2n$ *and constants* $\lambda_k \in \mathbb{R}_+$*,* $k = 1, \ldots, n$ *such that for every continuous function* $f : [0,1]^n \to \mathbb{R}$ *there exists* $\phi \in C\big([0,1]\big)$ *so that*

$$f(x_1, \ldots, x_n) = \sum_{j=0}^{2n} \phi\left(\sum_{k=1}^{n} \lambda_k \varphi_j(x_k)\right). \tag{2.11}$$

This theorem can be extended in various directions. First, one can restrict to increasing continuous functions φ_j and even show that the set of $2n+1$ tuples of such functions that fulfill the proposition is 'fat', in the sense of being of second Baire category. Moreover, one can show that there is a single continuous function φ in terms of which the φ_j's can be expressed as $\varphi_j(x) = c\varphi(aj+x)+bj$ with constants a, b, c.

From the point view of neural networks, Eq.(2.11) can be interpreted as a feedforward network with two hidden layers where the first hidden layer contains $n(2n+1)$ neurons, which use the φ_j's as activation functions, the second hidden layer contains $2n+1$ neurons with activation function ϕ and the output neuron uses a linear activation function $\sigma(z) = z$. Hence, Eq.(2.11) provides an exact representation using only finitely many neurons, but at the cost of having an activation function, namely ϕ, that depends on f.

2.4 VC dimension of neural networks

We will now look into bounds on the VC-dimension of feedforward neural networks. These will depend, among other things, on the chosen activation function. We will start with the simplest case where σ is a step function. For a single Perceptron Thm.1.9 and Cor.1.7 tell us that its VC-dimension is equal to the number of its parameters. The following theorem shows that this relation still almost holds for layered feedforward networks of Perceptrons:

Theorem 2.7: VC-dimension—step-activation functions

For arbitrary $n_0, \omega \in \mathbb{N}$ fix an architecture of a multilayered feedforward neural network with n_0 inputs, a single output and ω parameters (i.e., weights and threshold values). Denote by $\mathcal{F}$ the set of all functions $f : \mathbb{R}^{n_0} \to \{-1, 1\}$ that can be implemented by any feedforward network with this architecture when using $\sigma(z) = \operatorname{sgn}(z)$ as activation function. Then

$$\operatorname{VCdim}(\mathcal{F}) < 2\omega \log_2(e\omega). \tag{2.12}$$

Note: ω equals the number of edges in the graph that represents the network if we add to every neuron an additional edge that corresponds to the constant input related to the threshold value. Not surprisingly, the bound in Eq.(2.12) also holds (via the same argument) if $\sigma(z) = \mathbb{1}_{z \geq 0}$ and functions into $\{0, 1\}$ are considered.

Proof. Suppose the considered network has depth m and let n_i be the number of nodes (i.e., neurons or inputs) in the i'th layer with $i = 0, \ldots, m$. Then n_0 is the number of inputs and $n_m = 1$. We can decompose every function f that the considered architecture implements into functions $f_i : \mathbb{R}^{n_{i-1}} \to \{-1, 1\}^{n_i}$ that represent the mappings corresponding to the individual layers. Then $f = f_m \circ \cdots \circ f_1$. Furthermore, we can breakdown every $f_i(x) = \big(f_{i,1}(x), \ldots, f_{i,n_i}(x)\big)$ into its n_i components, each of which describes the action of a single neuron. Denote by $\omega_{i,j}$ the number of free parameters in $f_{i,j}$, i.e., the number of weights (including the threshold value) of the corresponding neuron. Then $\omega = \sum_{i=1}^m \sum_{j=1}^{n_i} \omega_{i,j}$ and we can bound the growth function of $\mathcal{F}$ via

$$\begin{aligned} \Gamma(n) \;&\leq\; \prod_{i=1}^m \Gamma_{f_i}(n) \;\leq\; \prod_{i=1}^m \prod_{j=1}^{n_i} \Gamma_{f_{i,j}}(n) \\ &\leq\; \prod_{i=1}^m \prod_{j=1}^{n_i} \left(\frac{en}{\omega_{i,j}}\right)^{\omega_{i,j}} \;\leq\; (en)^{\omega}. \end{aligned} \tag{2.13}$$

Here, the first inequality is an application of the composition property of the growth function shown in Lemma 1.4. Γ_{f_i} and $\Gamma_{f_{i,j}}$ denote the growth functions of the function classes corresponding to the i'th layer and the j'th neuron in the i'th layer, respectively. The step from the first to the second line in Eq.(2.13) exploits that by Thm.1.9 the VC-dimension of a single neuron is equal to the number of weights $\omega_{i,j}$, which then leads to an upper bound to the growth function following Thm.1.8. Finally, the last inequality simply uses $1/\omega_{i,j} \leq 1$.

From Eq.(2.13) we obtain that $\mathrm{VCdim}(\mathcal{F}) < D$, if $2^D > (eD)^\omega$. This is satisfied by $D = 2\omega \log_2(e\omega)$ for all $\omega > 1$. For $\omega = 1$, however, Eq.(2.12) holds as well since the VC-dimension in this case is at most 1. □

The bound on the VC-dimension given in Eq.(2.12) turns out to be asymptotically tight. That is, there are neural networks of the considered type for which a lower bound of order $\Omega(\omega \log \omega)$ can be proven. This implies that the VC-dimension of those networks is strictly larger than the sum of their components VC-dimension.

In Thm.2.1 we saw that an arbitrary Boolean function on d inputs can be represented by a neural network with 2^d neurons. As an implication of the above theorem on the VC-dimension of feedforward neural networks we can now show that an exponential number of neurons is indeed necessary.

Corollary 2.5. *For any $d \in \mathbb{N}$ consider feedforward neural networks with d inputs, a single output and activation function $\sigma(z) = \mathbb{1}_{z \geq 0}$. Within this setting the number of neurons $\nu(d)$ of the smallest architecture that is capable of representing any Boolean function $f : \{0,1\}^d \to \{0,1\}$ satisfies $\nu(d) + d \geq 2^{(d-2)/3}$.*

Proof. The VC-dimension corresponding to the smallest such architecture has to be at least the VC-dimension of the class of Boolean functions, which is 2^d. On the other hand, it is at most $2\omega \log_2(e\omega)$ by Thm.2.7. If we let $G = (V, E)$ be the underlying graph and use that $|E| \leq |V|^2/2$ and thus $\omega \leq |E| + |V| \leq |V|^2$, we can estimate

$$2^d \leq 2\omega \log_2(e\omega) \leq 2|V|^2 \log_2\left(e|V|^2\right) \leq 4|V|^3.$$

With $\nu(d) + d = |V|$ we arrive at the desired result. □

The same type of argument also applies to many other classes of activation functions once a (non-exponential) upper bound on the VC-dimension is established. Before we come to these bounds, we will have a look a small example that helps understanding the necessity of rather specific assumptions on the activation function for obtaining finite VC-dimension.

Consider the following family of activation functions:

$$\sigma_c(z) := \frac{1}{1 + e^{-z}} + c z^3 e^{-z^2} \sin(z), \quad c \geq 0. \tag{2.14}$$

The members of this family have many of the properties of the standard logistic sigmoid function, which is given by σ_0: the functions are analytic, satisfy $\lim_{z\to\infty} \sigma_c(z) = 1$, $\lim_{z\to-\infty} \sigma_c(z) = 0$ and for sufficiently small but not necessarily vanishing c we have that the second derivative $\sigma''(z)$ is strictly positive for $z < 0$ and strictly negative for $z > 0$. That is, the function is convex/concave in the respective regions.

Activation function	VCdim	Reference
sgn	$\mathcal{O}(\omega \log \omega)$	[4]
	$\Omega(\omega \log \omega)$	[18]
piecewise polynomial	$\mathcal{O}(\omega N)$	[21, 13]
	$\mathcal{O}(\omega\delta^2 + \omega\delta \log \omega)$	[2]
	$\mathcal{O}(\omega t)$	[12]
piecewise linear	$\mathcal{O}(\omega\delta \log \omega)$	[13]
	$\Omega\big(\omega\delta \log(\omega/\delta)\big)$	[13]
Pfaffian, incl. $1/(1+e^{-x})$	$\mathcal{O}(\omega^2 N^2)$	[16]
$\lim\limits_{x\to\infty} \sigma(x) \neq \lim\limits_{x\to-\infty} \sigma(x) \wedge \exists z : \sigma'(z) \neq 0$	$\Omega(\omega\delta)$	[17, 2]

Figure 2.3: VC-dimension bounds for various classes of feedforward neural networks depending on the activation function. N, ω and t denote the numbers of neurons, parameters and computational steps, respectively. δ is the depth, often assuming a layered architecture. Lower bounds of the form $\Omega(\ldots)$ mean that there exist networks with the stated asymptotic scaling.

Proposition 2.6 (Neural network with infinite VC-dimension)**.** *Consider feedforward networks with one input, a single output neuron whose activation function is $z \mapsto \operatorname{sgn}(z)$ and a single hidden layer with two neurons using σ_c with $c > 0$ as activation function. The class of functions $f : \mathbb{R} \to \{-1, 1\}$ representable by such networks has infinite VC-dimension.*

Proof. From Exp.1.6 we know that $\mathcal{F} := \{f : \mathbb{R}_+ \to \mathbb{R} \mid \exists \alpha \in \mathbb{R} : f(x) = \operatorname{sgn} \sin(\alpha x)\}$ has infinite VC-dimension. So the proposition follows by showing that $\mathcal{F}$ is contained in the function class representable by the considered networks. To this end, we choose the weights and threshold such that

$$\begin{aligned} f(x) &= \operatorname{sgn}\left[\sigma_c(\alpha x) + \sigma_c(-\alpha x) - 1\right] \\ &= \operatorname{sgn}\Big[2c(\alpha x)^3 e^{-\alpha^2 x^2} \sin(\alpha x)\Big] \;=\; \operatorname{sgn} \sin(\alpha x). \end{aligned}$$

□

Of course, this example is only of academic interest and, not surprisingly, the VC-dimensions for practically used activation functions is finite. In fact, in all known cases, it is bounded by a low degree polynomial in the size-parameters (such as number of neurons, parameters or layers, see Fig.2.3). The usual approach for obtaining upper bounds on the VC-dimension for networks with non-trivial activation function is to exploit the 'cell counting' method that gave rise to Thm.1.10 and Thm.1.11, either via those theorems or more directly. Here is one example:

Theorem 2.8: VC-dimension—piecewise polynomial units

Consider an arbitrary architecture of a feedforward neural network with N neurons, m inputs, a single output and ω real parameters (i.e., weights and threshold values). Let $\mathcal{F}$ be the set of functions from $\mathbb{R}^m$ to $\{0,1\}$ that is representable within this architecture when every hidden neuron uses a piecewise polynomial activation function of degree at most d and with at most p pieces and the output neuron uses $\sigma(z) = \mathbb{1}_{z\geq 0}$. Then

$$VCdim(\mathcal{F}) \leq 2\omega\Big[N\log_2 p + \log_2\Big(8e\max\big\{\delta+1, 2d^{\delta}\big\}\Big)\Big], \tag{2.15}$$

where $\delta \in \{0,\ldots,N-1\}$ denotes the depth of the network.

Remark: Here, *depth* means the largest number of hidden neurons along any path through the network. Hence, in a layered network δ is the number of hidden layers. The number ω only counts distinct parameters, i.e., if *weight sharing* is used each weight is only counted once.

Proof. Considering the ω parameters as additional variables, the network can be regarded as a function $\Psi : \mathbb{R}^m \times \mathbb{R}^\omega \to \{0,1\}$. We interpret Ψ as a predicate and aim at expressing it as a Boolean combination of polynomial (in-)equalities, so that Thm.1.10 applies. To this end, we have to bound the number of polynomial predicates and their degree.

We use $i \in \{1,\ldots,N\}$ to label the neurons, ordered so that the network graph contains no path from i to $j < i$. Denote by $\delta(i) \in \{0,\ldots,\delta\}$ the maximal depth of the i'th neuron in the network graph and by $a_i \in \mathbb{R}$ the value of the i'th neuron's output before applying the activation function.

Define $I : \mathbb{R} \to \{1,\ldots,p\}$ so that $I^{-1}(\{j\})$ is the interval corresponding to the j'th piece in the domain of the piecewise polynomial activation function. Clearly, $I(a)$ can be obtained from the truth values of $p-1$ inequalities that are linear in a. Since a_1 is a quadratic function on $\mathbb{R}^m \times \mathbb{R}^\omega$ the value of $I(a_1)$ can be obtained from $p-1$ polynomial predicates over $\mathbb{R}^m \times \mathbb{R}^\omega$ of degree 2. Now consider $I(a_i)$. Conditioned on $I(a_1),\ldots,I(a_{i-1})$, a_i is a polynomial of degree at most $d^{\delta(i)} + \sum_{j=0}^{\delta(i)} d^j =: deg(i)$ over $\mathbb{R}^m \times \mathbb{R}^\omega$. Consequently, $I(a_i)$ can be determined from the truth values of $p-1$ polynomial predicates over $\mathbb{R}^m \times \mathbb{R}^\omega$ of degree $deg(i)$. However, there are p^{i-1} different possibilities for $I(a_1),\ldots,I(a_{i-1})$ leading to up to $p^{i-1}(p-1)$ different polynomial predicates that, together with all previously obtained ones, determine $I(a_i)$.

So for determining $I(a_{N-1})$ we need at most $\sum_{j=1}^{N-1} p^{j-1}(p-1) = p^{N-1}-1$ polynomial predicates of degree at most $deg(N-1)$. Finally, determining $\mathbb{1}_{a_N\geq 0}$ requires, for each value of $I(a_1),\ldots,I(a_{N-1})$, one predicate that is linear in a_N and thus polynomial of degree $deg(N-1) \leq \max\{\delta+1, 2d^\delta\}$. This is also the maximal degree of all the, in total less than p^N, polynomial predicates. With these numbers, we can now enter Thm.1.10, which completes the proof. □

To summarize, if d and p are fixed, then

$$VCdim(\mathcal{F}) = \mathcal{O}(\omega N).$$

The proof technique is quite flexible in some directions: *weight sharing* or *max pooling* as used in *convolutional neural networks*, as well as *product units* or small variations of the architecture are easily incorporated without changing the picture. Under additional assumptions, such as a layered architecture or piecewise linear activation functions, the above bound can even be improved further, in the latter case down to $\mathcal{O}(\omega\delta \log \omega)$. However, if the basic units are not piecewise algebraic, then 'cell counting' can become considerably more difficult. For the logistic sigmoid $\sigma(x) = (1 + e^{-x})^{-1}$ this can, however, still be done leading to a bound based on Eq.(1.38). Fig.2.3 summarizes some of the known results and points to the corresponding references.

2.5 Rademacher complexity of neural networks

Rademacher complexities of feedforward neural networks are most easily estimated, if the network obeys a layered structure. Then, with the help of the properties of the (empirical) Rademacher complexities, which were summarized in Thm.1.14, we can express the Rademacher complexities at the outputs of one layer in terms of the ones corresponding to the foregoing layer:

Theorem 2.9: Rademacher complexity progression

Let $a, b \in \mathbb{R}$, $\tilde{\sigma} : \mathbb{R} \to \mathbb{R}$ l-Lipschitz and assume $\mathcal{F}_0 \subseteq \mathbb{R}^{\mathcal{X}}$ is a set of functions that includes the zero function. The empirical Rademacher complexity of $\mathcal{F} := \big\{x \mapsto \tilde{\sigma}\big(v + \sum_{j=1}^m w_j f_j(x)\big)\big|\ |v| \leq a, ||w||_1 \leq b, f_j \in \mathcal{F}_0\big)\big\} \subseteq \mathbb{R}^{\mathcal{X}}$ w.r.t. any point $x \in \mathcal{X}^n$ can be bounded in terms of the one of $\mathcal{F}_0$ (w.r.t. the same point) by

$$\hat{\mathcal{R}}(\mathcal{F}) \leq l \left(\frac{a}{\sqrt{n}} + 2b\, \hat{\mathcal{R}}(\mathcal{F}_0) \right). \tag{2.16}$$

The factor 2 can be dropped if $\mathcal{F}_0 = -\mathcal{F}_0$.

Note: the number m of neurons in the layer enters the bound only indirectly via the $|| \cdot ||_1$-constraint on the weights.

Proof. First, we exploit the Lipschitz-property of $\tilde{\sigma}$ together with the corresponding property of the Rademacher complexity (point 5. in Thm.1.14) and obtain

$$\hat{\mathcal{R}}(\mathcal{F}) \leq \frac{l}{n}\, \mathbb{E}_\sigma \left[\sup_{v,w,f_j} \sum_{i=1}^n \sigma_i \left(v + \sum_{j=1}^m w_j f_j(x_i) \right) \right]. \tag{2.17}$$

Next, note that $\sum_j w_j f_j \in b\,\mathrm{conv}\{\mathcal{F}_0 - \mathcal{F}_0\} =: \mathcal{G}_1$. With $\mathcal{G}_2 := \{x \mapsto v \mid |v| \leq a\}$ we can regard the function class that appears in Eq.(2.17) as a sum of function classes so that (following property 3. in Thm.1.14) we can write

$$\begin{aligned} \hat{\mathcal{R}}(\mathcal{F}) &\leq l\big(\hat{\mathcal{R}}(\mathcal{G}_1 + \mathcal{G}_2)\big) = l\big(\hat{\mathcal{R}}(\mathcal{G}_1) + \hat{\mathcal{R}}(\mathcal{G}_2)\big) \\ &\leq l\left(\frac{a}{\sqrt{n}} + 2b\,\hat{\mathcal{R}}(\mathcal{F}_0)\right). \end{aligned} \tag{2.18}$$

In the second line, we used separate bounds for the two appearing empirical Rademacher complexities. On the hand, we used that (again by Thm.1.14)

$$\hat{\mathcal{R}}(\mathcal{G}_1) = b\hat{\mathcal{R}}(\mathrm{conv}\{\mathcal{F}_0 - \mathcal{F}_0\}) = b\hat{\mathcal{R}}(\mathcal{F}_0 - \mathcal{F}_0) = b\big(\hat{\mathcal{R}}(\mathcal{F}_0) + \hat{\mathcal{R}}(-\mathcal{F}_0)\big) = 2b\hat{\mathcal{R}}(\mathcal{F}_0),$$

where the factor 2 is unnecessary if $\mathcal{F}_0 = -\mathcal{F}_0$. On the other hand, we used that $\hat{\mathcal{R}}(\mathcal{G}_2) \leq a\mathbb{E}(|Z|)/n$ with $Z := \sum_{i=1}^n \sigma_i$, which in turn can be bounded via Jensen's inequality leading to

$$\mathbb{E}[|Z|] \leq \mathbb{E}[Z^2]^{1/2} = \sqrt{n},$$

when exploiting the independence of the Rademacher variables. □

In order to arrive at an upper bound for the empirical Rademacher complexity of en entire network, we can now apply the previous theorem recursively. Only the first layer requires a different treatment. One possibility is to use the following ingredient:

Lemma 2.7. *For $b, c > 0$, consider $\mathcal{X} := \{x \in \mathbb{R}^d \mid ||x||_\infty \leq c\}$ and $\mathcal{G} := \{\mathcal{X} \ni x \mapsto \langle x, w\rangle \mid ||w||_1 \leq b\}$. The empirical Rademacher complexity of $\mathcal{G}$ w.r.t. any $z \in \mathcal{X}^n$ can be bounded by*

$$\hat{\mathcal{R}}(\mathcal{G}) \leq \frac{bc}{\sqrt{n}}\sqrt{2\ln(2d)}. \tag{2.19}$$

Proof. The proof is an application of Hölder's and Massart's inequality:

$$\begin{aligned} n\hat{\mathcal{R}}(\mathcal{G}) = \mathbb{E}_\sigma\left[\sup_w \sum_{j=1}^d \sum_{i=1}^n \sigma_i w_j z_{i,j}\right] &\leq b\,\mathbb{E}_\sigma\left[\max_j \left|\sum_{i=1}^n \sigma_i z_{i,j}\right|\right] \\ &= b\,\mathbb{E}_\sigma\left[\max_{a\in A} \sum_{i=1}^n \sigma_i a_i\right] \\ &\leq bc\,\sqrt{2n\ln(2d)}, \end{aligned}$$

where we used Hölder's inequality in the first line, we set $A := \{\pm x_1, \ldots, \pm x_d\} \subseteq \mathbb{R}^n$ with $(x_j)_i := z_{i,j}$ in the second line and we exploited Massart's inequality from Eq.(1.54) with $|A| \leq 2d$ and $||x_j||_2 \leq \sqrt{n}||x_j||_\infty \leq \sqrt{n}c$ in the last line. □

Combining Lemma 2.7 and Thm.2.9 then leads to the following:

Corollary 2.8 (Rademacher complexity of layered network)**.** *Let* $a, b > 0$ *and* $\mathcal{X} := \{x \in \mathbb{R}^d | \; ||x||_\infty \leq 1\}$*. Fix a neural network architecture with* δ *hidden layers that implements* $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{X}}$ *s.t.*

1. *the activation function* $\sigma : \mathbb{R} \to \mathbb{R}$ *is* $1-$*Lipschitz and anti-symmetric, i.e.* $\sigma(-x) = -\sigma(x)$*,*

2. *within every layer the vector* w *of weights satisfies* $||w||_1 \leq b$*,*

3. *the moduli of the threshold values are bounded by* a*.*

Then the empirical Rademacher complexity of $\mathcal{F}$ *w.r.t. any* $z \in \mathcal{X}^n$ *then satisfies*

$$\hat{\mathcal{R}}(\mathcal{F}) \leq \frac{1}{\sqrt{n}} \left(b^\delta \sqrt{2 \ln(2d)} + a \sum_{i=0}^{\delta-1} b^i \right). \tag{2.20}$$

Note: $\sigma = \tanh$ for instance satisfies the requirements.

Proof. The result follows by replacing the function class $\mathcal{G}_1$ in the proof of Thm.2.9 with the class $\mathcal{G}$ of Lemma 2.7. Moreover, we use $l = 1$ as well as $\mathcal{F}_0 = -\mathcal{F}_0$ for every single layer. □

The appearance of b^δ, which bounds the product $\prod_{i=1}^{\delta} ||w_i||_1$ of norms of the weight vectors of all layers, motivates the use of expressions of this type for regularization (as opposed to $\sum_{i=1}^{\delta} ||w_i||_2^2$ in Tikhonov regularization).

2.6 Training neural networks

In this section, we will sketch the framework for training a neural network. Particular aspects and issues will then be discussed in greater detail in the following sections. Training means to optimize the parameters of the model so that that the model describes the training data well. Hence, we will have to choose a loss function and an optimization algorithm. The architecture of the network is supposed to be fixed.

Loss function. In case of regression, the most common choices for the loss function are the quadratic loss and the l_1-distance. The latter is less sensitive to ‘outliers’ than the former.

In case of classification, a common practice is the following: If $\mathcal{Y}$ is the set of possible classes, then instead of choosing one output node with values ranging in $\mathcal{Y}$, one uses $|\mathcal{Y}|$ output nodes with real values, say $z \in \mathbb{R}^{\mathcal{Y}}$. To those the so-called *softmax* activation

$$\sigma_{\max}(z)_y := \frac{e^{\beta z_y}}{\sum_{y \in \mathcal{Y}} e^{\beta z_y}}, \quad \beta \in [0, \infty) \tag{2.21}$$

is applied. Note that the softmax function is, in contrast to other activation functions, not a scalar function that can be applied to each coordinate separately, but maps $\mathbb{R}^{\mathcal{Y}}$ into $(0,\infty)^{\mathcal{Y}}$. More precisely, it turns a vector with real entries into a probability distribution that indicates the maximum entry in the limit $\beta \to \infty$ and is a 'soft' version thereof otherwise. Usually, $\beta = 1$ is chosen.

Using such an output layer, the network maps every input $x \in \mathcal{X}$ to a probability distribution, which can be interpreted as a quantification of the levels of confidence. Let us denote the components of the distribution by $p(y|x,w)$, where w is the collection of parameters the network depends on. The training data set $\big((x_1,y_1),\ldots,(x_n,y_n)\big)$, on the other hand, defines an empirical distribution $\hat{p}(y|x_i) := \delta_{y,y_i}$. For every data point x_i the deviation between the two distributions can be quantified using the Kullback-Leibler divergence

$$\begin{aligned} D_{KL}\big(\hat{p}(x_i)||p(x_i,w)\big) &:= \sum_{y\in\mathcal{Y}} \hat{p}(y|x_i) \log \frac{\hat{p}(y|x_i)}{p(y|x_i,w)} \\ &= -\log p(y_i|x_i,w). \end{aligned} \tag{2.22}$$

To cast this into a loss function of the form defined in Sec.1.1 we can use the modified space of labels $\tilde{\mathcal{Y}} := \mathbb{R}^{\mathcal{Y}}$ with $\tilde{y}_i := (\delta_{y,y_i})_{y\in\mathcal{Y}}$. The loss function $L : \tilde{\mathcal{Y}} \times \tilde{\mathcal{Y}} \to [0,\infty]$ giving rise to Eq.(2.22) then takes on the form

$$L\big(\tilde{y}, h(x)\big) = -\langle \tilde{y}, \log h(x)\rangle,$$

which is sometimes simply called the *log-loss*. It is also known as *cross entropy* (when viewed as arising from the Kullback-Leibler divergence) or the *negative log-likelihood* (when $w \mapsto p(y|x,w)$ is viewed as likelihood function for the parameters w). Irrespective of these motivations the main reason for choosing this type of loss-function is, however, that it appears to work well, on heuristic grounds.

Algorithm The risk-function that has to be optimized as a function of the parameters, which we denote by $w \in \mathbb{R}^N$, is always an average of the loss-function over the n training data points. That is, formally we have to minimize a function of the form

$$f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w).$$

Gradient descent is a greedy (and probably the conceptually simplest) algorithm for dealing with such an optimization problem: choose an initial point $w_0 \in \mathbb{R}^N$ and a (possibly time-dependent) step size $\alpha > 0$ and then move along the sequence given by

$$w_{t+1} := w_t - \alpha \nabla f(w_t),$$

where $\nabla f(w_t)$ denotes the gradient of f at w_t. In other words, follow the negative gradient. Leaving aside issues of convergence, non-global minima and saddle-points for the moment, there are (at least) two obstacles to overcome if both n and N are very large. Consider a naive numerical way of computing the gradient: due to linearity, the gradient is the sum of n gradients $\nabla f_i(w_t)$ each of which requires of the order of N function evaluations to be estimated numerically since N is the dimension of the underlying space. Hence, before even a single step can be made downhill, $n \cdot N$ function evaluations are required. In our case, this means $n \cdot N$ evaluations (a.k.a. 'forward passes') of the neural network, which can easily be around 10^{14} — not very encouraging.

Two crucial ingredients reduce these nN evaluations, loosely speaking, to 2:

1. *Backpropagation:* For each f_i the gradient can be computed analytically. Using the chain rule combined with some bookkeeping this requires only one forward and one 'backward' pass (and in this sense 2 evaluations). This routine is called 'backpropagation'.

2. *Stochastic gradient descent:* Instead of computing the gradient of f exactly, one uses the gradients of the f_i's as stochastic approximation. After all, on average the gradients of the latter are equal to the gradient of the former.

With these two ingredients, which will be discussed in detail in Sec. 2.7 and Sec. 2.8, one possible way of proceeding is then as follows: start at a random initial point w_0, choose a data point (x_i, y_i) at random (in practice, usually without replacement), compute the gradient of the corresponding f_i using backpropagation, update w by moving opposite to that gradient, and then iterate this procedure.

One complete run over the set of n training data points is called an *epoch*. Instead of making n steps in parameter space during an epoch, it is often advantageous to form disjoint groups, so-called *mini-batches*, of k data points each and to average the corresponding k gradients before making one step. In this way, the stochastic gradient becomes less 'noisy', the step size can be increased and, since the gradients within one mini-batch are computed using the same parameters but different data points, it opens a door for parallelization.

2.7 Backpropagation

... exact computation of the gradient...

... to be completed ...

Consider a layered feedforward neural network whose layers will be labeled by an upper or lower index $l \in \{0, \dots, m\}$. Here, the 0'th layer corresponds to the input and the m'th to the output. N_l will be the number of neurons in the l'th layer. By w^l_{jk} we will denote the weight that corresponds to the connection from the k'th neuron in layer $l-1$ to the j'th neuron in layer l. Similarly, b^l_j will be the threshold value of the j'th neuron in layer l. The vector x^l, whose components x^l_j are the outputs of the neurons of the l'th layer, can then be expressed in matrix/vector notation as $x^l = \sigma(w^l x^{l-1} + b^l)$, where the activation function σ is applied component-wise. We introduce a separate variable $z^l := w^l x^{l-1} + b^l$ to denote the output before application of the activation function.

Consider a function $f : \mathbb{R}^{N_m} \to \mathbb{R}$ that maps the output x^m to a real number—such as the loss function, which acts as $L(y, x^m) =: f(x^m)$ for a fixed pair (x^0, y) of the training data. By expanding x^m in terms of previous layers and the corresponding weights and threshold values, we may interpret f as a function of different kinds of variables. In particular, we will consider the mappings $(w, b) \mapsto f(x^m)$, $x^l \mapsto f(x^m)$ and $z^m \mapsto f(x^m)$. Abusing notation all these mappings will be denoted by f.

Our aim is to compute the partial derivatives of f w.r.t. all weights and threshold values. To this end, we introduce intermediate quantities $\delta^l_j := \frac{\partial f}{\partial z^l_j}$ in terms of which all the sought derivatives will be expressed by use of the chain rule. The latter is also central in computing the δ^l_j's themselves. Beginning with the output layer, we obtain

$$\delta^m_j = \sum_k \frac{\partial f}{\partial x^m_k} \frac{\partial x^m_k}{\partial z^m_j} = \sigma'(z^m_j)\, \frac{\partial f}{\partial x^m_j}, \tag{2.23}$$

where the summation runs over all neurons in the considered layer. Next, we show that δ^l can be expressed in terms of δ^{l+1}, so that all δ's can be computed by going layerwise backwards from the output layer:

$$\begin{aligned} \delta^l_j = \frac{\partial f}{\partial z^l_j} &= \sum_k \frac{\partial f}{\partial z^{l+1}_k} \frac{\partial z^{l+1}_k}{\partial z^l_j} \\ &= \sum_k \delta^{l+1}_k \, w^{l+1}_{kj} \, \sigma'(z^l_j). \end{aligned} \tag{2.24}$$

Finally, we can express the sought derivatives in terms of the δ's:

$$\frac{\partial f}{\partial b^l_j} = \sum_k \frac{\partial f}{\partial z^l_k} \frac{\partial z^l_k}{\partial b^l_j} = \delta^l_j, \tag{2.25}$$

$$\frac{\partial f}{\partial w^l_{jk}} = \sum_i \frac{\partial f}{\partial z^l_i} \frac{\partial z^l_i}{\partial w^l_{jk}} = \delta^l_j \, x^{l-1}_k. \tag{2.26}$$

As a result we obtain that in order to compute the partial derivatives of f w.r.t. all weights and threshold values it suffices to run the network once forward (to obtain all x's and z's) and once 'backwards' (to obtain the δ's). This has to be contrasted with the naive approach, where for every individual partial derivative the network had to be evaluated twice already.

... add: approach via Lagrange duality ...

2.8 Gradient descent and descendants

This section is a first excursion into optimization theory. Motivated by but not restricted to the training of neural networks, we will have a look at iterative optimization methods that can be regarded as descendants of the *gradient descent* method. The common strategy of these methods for minimization of a function $f:\mathbb{R}^d\rightarrow\mathbb{R}$ is as follows. Start at a randomly/cleverly chosen point $x_0\in\mathbb{R}^d$ and then step-by-step move along a path given by $x_{t+1}=x_t+\Delta_t$ that is iteratively constructed and ideally 'descending' regarding the value of the function. Here, the increment Δ_t depends on (stochastic approximations of) local properties of the function at x_t and, in some cases, on the history of the path up to that point. The central 'local property' is the gradient of the function at x_t. Using higher order derivatives is in principle beneficial, but the computational costs per step often exceed the feasibility limit if the problem at hand is very high-dimensional (especially, if $d\gg 10^5$)[4].

Gradient descent and its descendants are ideally suited for the realm of high dimensions. The main reason for this is that its *oracle complexity* is essentially dimension-free. That is, the number of times the gradient (or function value) has to be computed before convergence is achieved up to some accuracy ϵ, is essentially independent of the dimension. In fact, all bounds that are derived in this section are independent of d. They do involve constants characterizing continuity or convexity properties of the function and, of course, those may implicitly depend on the dimension. Also the cost of each evaluation of the function or of its gradient depends on d. In contrast to other methods (such as interior point or ellipsoid methods, which when applicable would have much faster convergence) gradient descent techniques, however, do not add additional dimension factors.

Steepest descent Before going into details, let us consider different choices for the increment Δ_t from a more distant perspective. Assuming differentiability, we can approximate the function in a neighborhood around a point x as

$$f(x-\Delta)\simeq f(x)+\langle\Delta,\nabla f(x)\rangle.$$

Aiming at a descending path, a reasonable choice for the increment Δ is thus one that minimizes the inner product with the gradient. This will determine the direction of *steepest descent*. Bounding the step size by $\alpha>0$, which is ideally chosen so that the linear approximation is still reasonably good, this means

$$\Delta=\operatorname{argmin}\big\{\langle\Delta,\nabla f(x)\rangle\big|\;||\Delta||\leq\alpha\big\}. \tag{2.27}$$

At this point, we have to choose the norm (or even a more general normalizing functional) that constrains Δ. Suppose we choose $||x||:=\langle x,Px\rangle^{1/2}$ for some positive definite matrix P. Then

$$\Delta=-\frac{\alpha P^{-1}\nabla f(x)}{||P^{-1}\nabla f(x)||} \tag{2.28}$$

solves the minimization problem. If $P=\mathbb{1}$, which means that $||\cdot||=||\cdot||_2$ is the Euclidean norm, then Δ equals the step size times the negative gradient. This is the choice made in the gradient descent method. However, the Euclidean norm may not be the most natural or most relevant choice. For instance, if

[4] Neural networks are trained with currently up to $d\sim 10^{11}$ parameters.

we regard the constraint due to α as a guarantee for the quality of the linear approximation, then the norm where P equals the *Hessian* of f at x seems to be more appropriate. In fact, this is the choice made in *Newton's method.* Other choices can be well-motivated, as well. Having in mind generalization, regularization or sparsity, one may for instance want to have preferred directions, which are then reflected in the chosen normalizing functional. We will, however, now close this door again and have a closer look at the relatives of gradient descent, where the Euclidean norm lies beneath the update rule $x_{t+1} = x_t - \alpha\nabla f(x_t)$.

Gradient descent For gradient descent to become a meaningful algorithm, the gradient should not be too wild. One way to formalize this is to demand the gradient to be Lipschitz continuous. The following Lemma summarizes two central implications of this assumption.

Lemma 2.9. *Let $x, y \in \mathbb{R}^d$ and $f \in C^1(\mathbb{R}^d)$ be such that ∇f is L-Lipschitz. Then the following holds with the norm induced by the inner product:*

1. $\left|f(x) - f(y) - \langle\nabla f(x), x-y\rangle\right| \leq \frac{L}{2}||x-y||^2$.
2. *If f is convex:* $f(x) - f(y) - \langle\nabla f(x), x-y\rangle \leq -\frac{1}{2L}||\nabla f(y) - \nabla f(x)||^2$.

Note: The first inequality shows how the function can be bounded by a quadratic function. The second inequality can be regarded as a strengthening of the convexity condition. In fact, if we set the r.h.s. of the second inequality to zero ($L = \infty$), then validity of the inequality for all x, y is equivalent to convexity of f. Geometrically, this is the tangent plane lying below the graph.

Proof. 1. By the fundamental theorem of calculus, we can write $f(x) - f(y) = \int_0^1 \langle\nabla f\big(y + t(x-y)\big), x-y\rangle dt$. With the help of Cauchy-Schwarz and the Lipschitz-property of the gradient, this leads to

$$\begin{aligned}\left|f(x) - f(y) - \langle\nabla f(x), x-y\rangle\right| &\leq \int_0^1 \Big|\langle\nabla f\big(y+t(x-y)\big) - \nabla f(x), x-y\rangle\Big| dt \\ &\leq \int_0^1 \|\nabla f\big(y+t(x-y)\big) - \nabla f(x)\| \; \|x-y\| dt \\ &\leq \int_0^1 tL\|x-y\|^2 dt \;=\; \frac{L}{2}\|x-y\|^2.\end{aligned}$$

2. To prove the second inequality, we introduce the auxiliary variable $z := y + (\nabla f(x) - \nabla f(y))/L$. Then

$$\begin{aligned}&f(x) - f(y) = f(x) - f(z) + f(z) - f(y) \\ &\leq \langle\nabla f(x), x-z\rangle + \langle\nabla f(y), z-y\rangle + \frac{L}{2}||z-y||^2 \\ &= \langle\nabla f(x), x-y\rangle - \frac{1}{2L}||\nabla f(x) - \nabla f(y)||^2, \qquad (2.29)\end{aligned}$$

where the step to the second line used convexity of f for the first two terms and exploited the just proven inequality 1. to bound $f(z) - f(y)$. □

If we insert $x = x_t$ and $y = x_{t+1} = x_t - \alpha\nabla f(x_t)$ into inequality 1. of Lemma 2.9, we obtain, after collecting terms

$$f(x_t) - f(x_{t+1}) \geq \alpha\left(1 - \frac{\alpha L}{2}\right) ||\nabla f(x_t)||^2. \tag{2.30}$$

Assuming that the gradient is not vanishing, the r.h.s. of Eq.(2.30) is positive, which means that gradient descent is indeed descending, if $\alpha \in (0, 2/L)$. Furthermore, it is maximal when $\alpha = 1/L$. If the update rule would be $x_{t+1} = x_t - P\nabla f(x_t)$ for some positive matrix P, then the operator norm $||P||_\infty$ would play the role of α and $||P||_\infty \in (0, 2/L)$ would imply monotonicity. We will, however, not pursue this direction further and stick to the standard update. The following theorem collects the implications of Eq.(2.30). It shows, in particular, that under reasonable assumptions gradient descent converges to a stationary point.

Theorem 2.10: Gradient descent - convergence to stationarity

Let $f \in C^1(\mathbb{R}^d)$ have L-Lipschitz gradient and consider the sequence $x_{t+1} = x_t - \alpha\nabla f(x_t)$ for some $\alpha \in (0, 2/L)$ and $x_0 \in \mathbb{R}^d$. Then

1. $f(x_{t+1}) < f(x_t)$ unless $\nabla f(x_t) = 0$.
2. If f is bounded from below, then $\nabla f(x_t) \to 0$ for $t \to \infty$.
3. If f attains a minimum at x^* and we choose $\alpha = 1/L$, then for all $T \in \mathbb{N}$:
$$\min_{t<T} ||\nabla f(x_t)||^2 \leq \frac{2L\big(f(x_0) - f(x^*)\big)}{T}. \tag{2.31}$$

Proof. 1. follows immediately from Eq.(2.30). In order to arrive at 2. and 3. we take the sum $\sum_{t=0}^{T-1}$ over Eq.(2.30). Then

$$\begin{aligned} f(x_0) - f(x_T) &\geq \alpha\left(1 - \frac{\alpha L}{2}\right) \sum_{t=0}^{T-1} ||\nabla f(x_t)||^2 \\ &\geq \alpha\left(1 - \frac{\alpha L}{2}\right) T \min_{t<T} ||\nabla f(x_t)||^2. \end{aligned}$$

By assumption, the l.h.s. in the first line is uniformly bounded for all T. So we can take the limit $T \to \infty$ and observe that the r.h.s. can only remain bounded if $\nabla f(x_t) \to 0$. 3. follows from $f(x_T) \geq f(x^*)$ when inserting $\alpha = 1/L$. □

In order to obtain results that are stronger than mere (and rather slow) convergence towards a stationary point, we need stronger assumptions. An often made assumption is strong convexity (see Def.1.20). An alternative and slightly more general condition is the *Polyak-Łojasiewicz inequality* for some $\mu > 0$:

$$\forall x \in \mathbb{R}^d : \quad \frac{1}{2}||\nabla f(x)||^2 \geq \mu\big(f(x) - f(x^*)\big), \tag{2.32}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is supposed to attain a global minimum at x^*. Note that the validity of this inequality for $\mu > 0$ implies that every stationary point is a global minimum. The condition is independent of convexity[5], but it is implied by μ-strong convexity:

Lemma 2.10 (Polyak-Łojasiewicz from strong convexity). *If there is a $\mu > 0$ for which $f : \mathbb{R}^d \to \mathbb{R}$ is $\mu-$strongly convex, then Eq.(2.32) holds for every subgradient.*

Note: recall that $v \in \mathbb{R}^d$ is a *subgradient* of f at x iff

$$\forall y \in \mathbb{R}^d : \quad f(y) \geq f(x) + \langle v, y - x\rangle. \tag{2.33}$$

If f is convex and continuous, then at every point x the set of subgradients is non-empty. If it is in addition differentiable at x, then the subgradient at x is unique and given by the gradient $\nabla f(x)$. Where convenient, we will use the notation $\nabla f(x)$ also for subgradients in the non-differentiable case.

Proof. By definition f is μ-strongly convex iff the map $x \mapsto f(x) - \mu||x^2||/2$ is convex. Applied to this map, Eq.(2.33) reads

$$f(y) \geq f(x) + \frac{\mu}{2}||x - y||^2 + \langle \nabla f(x), y - x\rangle. \tag{2.34}$$

Minimizing both sides w.r.t. y then gives $f(x^*) \geq f(x) - ||\nabla f(x)||^2/(2\mu)$, which is the Polyak-Łojasiewicz inequality. □

Theorem 2.11: Gradient descent - exponential convergence

Let $f \in C^1(\mathbb{R}^d)$ satisfy the Polyak-Łojasiewicz inequality in Eq.(2.32) for some $\mu > 0$, have $L-$Lipschitz gradient and a global minimum attained at x^*. For $\alpha \in [0, 2/L]$ and $x_0 \in \mathbb{R}^d$ the sequence $x_{t+1} := x_t - \alpha\nabla f(x_t)$ satisfies for all $T \in \mathbb{N}$:

$$\begin{aligned} f(x_T) - f(x^*) \quad &\leq \quad \Big(1 + \alpha\mu(\alpha L - 2)\Big)^T \big(f(x_0) - f(x^*)\big) \\ &= \quad \Big(1 - \frac{\mu}{L}\Big)^T \big(f(x_0) - f(x^*)\big), \quad \text{for} \quad \alpha = 1/L. \end{aligned} \tag{2.35}$$

Note: Depending on the community this type of convergence is called *linear*, *exponential* or *geometric convergence*.

[5] For instance, $x \mapsto x^2 + 3(\sin x)^2$ is not convex but satisfies Eq.(2.32) with $\mu = 1/32$ and, on the other side, $x \mapsto |x|$ is convex but does not satisfy Eq.(2.32) for any $\mu > 0$.

Proof. We begin with applying inequality 1. from Lemma 2.9 and inserting the update rule. Then

$$\begin{aligned} f(x_{t+1}) &\leq f(x_t) + \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2}||x_{t+1} - x_t||^2 \\ &= f(x_t) - \alpha(1 - \alpha L/2)\ ||\nabla f(x_t)||^2 \\ &\leq f(x_t) + \alpha\mu(\alpha L - 2)\big(f(x_t) - f(x^*)\big), \end{aligned} \tag{2.36}$$

where the last step used the Polyak-Łojasiewicz condition. Subtracting $f(x^*)$ from both sides of the inequality and applying it recursively then leads to the claimed result. □

Note that the speed of convergence in this bound is governed by L/μ. If $f \in C^2$, then locally L/μ corresponds to the condition number of the Hessian. So, loosely speaking, the better the Hessian is conditioned, the faster the convergence. Note that also Newtons methods appears well-motivated in this light, since it aims at minimizing the condition number of the Hessian by locally transforming it to the identity matrix. In fact, in this way, Newton's method achieves super-exponential convergence.

Stochastic gradient descent We will now consider variants of the *stochastic gradient descent* method, where the gradient is replaced by a stochastic approximation. This is no longer a strict 'descent', since the direction of the increment now becomes a random variable, which is only proportional to the gradient on average.

Theorem 2.12: Stochastic gradient descent - fixed step size

Let $f \in C^1(\mathbb{R}^d)$ satisfy the Polyak-Łojasiewicz inequality in Eq.(2.32) for some $\mu > 0$, have $L-$Lipschitz gradient and a global minimum attained at x^*. For any $T \in \mathbb{N}$, $x \in \mathbb{R}^d$ let $g_1(x), \ldots, g_T(x)$ be i.i.d. random variables with values in $\mathbb{R}^d$ such that $\mathbb{E}[g_t(x)] = \nabla f(x)$. With $x_0 \in \mathbb{R}^d$ consider the sequence $x_{t+1} := x_t - \alpha g_t(x_t)$.

1. If $\forall x, t : \mathbb{E}\big[||g_t(x)||^2\big] \leq \gamma^2$ and $\alpha \in [0, 1/(2\mu)]$, then

$$\mathbb{E}\big[f(x_T)\big] - f(x^*) \leq (1 - 2\mu\alpha)^T \big(f(x_0) - f(x^*)\big) + \frac{L\alpha\gamma^2}{4\mu}. \tag{2.37}$$

2. If $\forall x, t : \mathbb{E}\big[||g_t(x)||^2\big] \leq \beta^2 ||\nabla f(x)||^2$ and $\alpha = 1/(L\beta^2)$, then

$$\mathbb{E}\big[f(x_T)\big] - f(x^*) \leq \left(1 - \frac{\mu}{L\beta^2}\right)^T \big(f(x_0) - f(x^*)\big). \tag{2.38}$$

Proof. We begin as in the proof of Thm.2.11 and insert the update rule into inequality 1. from Lemma 2.9. In this way, we obtain

$$f(x_{t+1}) \leq f(x_t) - \alpha\langle \nabla f(x_t), g_t(x_t) \rangle + \alpha^2 L ||g_t(x_t)||^2/2.$$

Next, we take the expectation value w.r.t. g_t conditioned on fixed $g_1, \ldots, g_{t-1}$:

$$\mathbb{E}_{g_t}\big[f(x_{t+1})\big] \leq f(x_t) - \alpha||\nabla f(x_t)||^2 + \alpha^2 L \mathbb{E}_{g_t}\big[||g_t(x_t)||^2\big]/2. \tag{2.39}$$

In order to prove Eq.(2.37) we proceed with bounding the last term in Eq.(2.39) in terms of γ^2 and the second term using the Polyak-Łojasiewicz inequality. Subtracting $f(x^*)$ from both sides of the inequality and taking the expectation value also w.r.t. $g_1, \ldots, g_{t-1}$ then leads to

$$\mathbb{E}\big[f(x_{t+1}) - f(x^*)\big] \leq \mathbb{E}\big[f(x_t) - f(x^*)\big](1 - 2\mu\alpha) + \alpha^2 L\gamma^2/2.$$

Now we can apply the resulting inequality recursively, so that with $T = t + 1$:

$$\mathbb{E}\big[f(x_T) - f(x^*)\big] \leq \big(f(x_0) - f(x^*)\big)(1 - 2\mu\alpha)^T + \frac{\alpha^2 L\gamma^2}{2}\sum_{k=0}^{T-1}(1 - 2\mu)^k,$$

which, after upper bounding the sum by the geometric series, becomes Eq.(2.37).

To obtain Eq.(2.38) we proceed similarly from Eq.(2.39), but now bound the last term in terms of $\beta^2||\nabla f(x_t)||^2$ and then apply the Polyak-Łojasiewicz inequality. Again, we subtract $f(x^*)$ from both sides of the resulting inequality and take the expectation value w.r.t. the remaining random variables. This leads to

$$\mathbb{E}\big[f(x_{t+1}) - f(x^*)\big] \leq \mathbb{E}\big[f(x_t) - f(x^*)\big]\big(1 - \mu\alpha(2 - L\alpha\beta^2)\big),$$

which can be iterated and then leads to Eq.(2.38) after inserting $\alpha = 1/(L\beta^2)$. □

Eq.(2.37) exhibits a central aspect of stochastic gradient descent: a fragile relation between the speed of convergence on large scales and the prevention of convergence by stochastic noise. One the one hand, the first term on the r.h.s. in Eq.(2.37) motivates a large steps size that guarantees fast convergence. The second term, on the other hand, shows that beyond a certain value, which grows with the step size, there is no convergence anymore. This is where stochastic noise dominates (cf. Fig.2.4). In the second statement of the theorem, Eq.(2.38), by assumption, the stochastic noise gets suppressed more and more when coming closer to the minimum. This implies that all stochastic gradients have to vanish simultaneously at the minimum. As this is an extremely strong (and typically unjustified) assumption, we will not pursue it further after having mentioning that the two cases can easily be combined into one assuming that $\forall x, t : \mathbb{E}\big[||g_t(x)||^2\big] \leq \beta^2||\nabla f(x)||^2 + \gamma^2$.

Eq.(2.37) is consistent with a heuristic strategy that is often used in practice: use constant step size for a long time (until stochastic noise prevents progress), then halve the step size and iterate this procedure.

The next theorem shows that appropriately decreasing the step size can indeed guarantee convergence when the function is convex. For this, neither differentiability nor the Polyak-Łojasiewicz condition are necessary. The statement is proven under the additional constraint, that the path remains inside a given compact convex set.

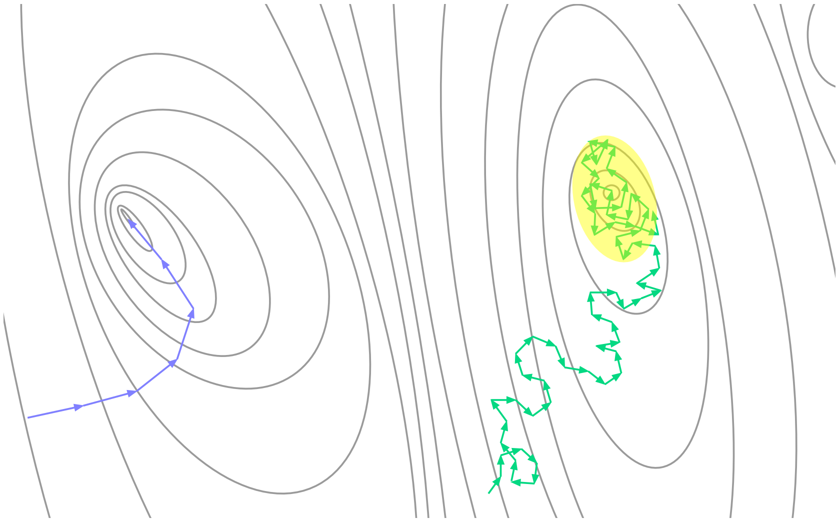


Figure 2.4: Whereas gradient descent (left) with step size $1/L$ always converges to a stationary point (see Thm.2.10), stochastic gradient descent (right) with constant step size can only be expected to converge towards a region, depicted by the yellow ellipsoid, around a critical point (see Eq.(2.37)).

Theorem 2.13: Stochastic subgradient descent

Let $P_C : \mathbb{R}^d \to C$ be the projection onto a compact convex set $C \subset \mathbb{R}^d$ with diameter δ (i.e., $x, y \in C \Rightarrow ||x-y||_2 \leq \delta$) and let $f : C \to \mathbb{R}$ be convex with global minimum at $x^* \in C$. For any $T \in \mathbb{N}$ and $x \in \mathbb{R}^d$ let $g_1(x), \ldots, g_T(x)$ be i.i.d. random variables so that $\mathbb{E}[g_t(x)]$ is any subgradient $\nabla f(x)$ of f at x and $\mathbb{E}[||g_t(x)||^2] \leq \gamma^2$. Let $\alpha \in \mathbb{R}^T$ be s.t. $0 \leq \alpha_t \leq \alpha_{t-1}$ and consider a sequence starting at $x_0 \in \mathbb{R}^d$ and defined by $x_t := P_C\big(x_{t-1} - \alpha_t g_t(x_{t-1})\big)$. Then $\overline{x} := \frac{1}{T}\sum_{t=0}^{T-1} x_t$ satisfies

$$\begin{aligned}\mathbb{E}\big[f(\overline{x})\big] \quad &\leq \quad f(x^*) + \frac{1}{2T}\left(\frac{\delta^2}{\alpha_T} + \gamma^2 \sum_{t=1}^{T} \alpha_t\right) \quad \text{and} \\ &\leq \quad f(x^*) + \sqrt{\frac{2}{T}}\,\delta\gamma \quad \text{for} \quad \alpha_t := \frac{\delta}{\sqrt{2t}\gamma}\,. \end{aligned} \tag{2.40}$$

Proof. We first use that P_C, being a projection, is norm non increasing, i.e., in particular $||x_t - x^*||^2 \leq ||x_{t-1} - x^* - \alpha_t g_t(x_{t-1})||^2$. Taking the expectation w.r.t. g_t we obtain for fixed $g_1, \ldots, g_{t-1}$:

$$\begin{aligned}\mathbb{E}_{g_t}\big[||x_t - x^*||^2\big] \quad &\leq \quad ||x_{t-1} - x^*||^2 - 2\alpha_t \langle \nabla f(x_{t-1}), x_{t-1} - x^* \rangle + \alpha_t^2\gamma^2 \\ &\leq \quad ||x_{t-1} - x^*||^2 - 2\alpha_t\big(f(x_{t-1}) - f(x^*)\big) + \alpha_t^2\gamma^2,\end{aligned}$$

where we used $\mathbb{E}[||g_t(x)||^2] \leq \gamma^2$ and the last inequality exploited the subgradient inequality, Eq.(2.33). Taking the expectation also w.r.t. to $g_1, \ldots, g_{t-1}$ we can rewrite the resulting inequality as

$$\mathbb{E}\big[f(x_{t-1})\big] - f(x^*) \leq \frac{\gamma^2}{2}\alpha_t + \frac{1}{2\alpha_t}\mathbb{E}\big[||x_{t-1} - x^*||^2 - ||x_t - x^*||^2\big]. \tag{2.41}$$

As an intermediate step, consider the sum

$$\begin{aligned} &\sum_{t=1}^T \frac{1}{\alpha_t}\big(||x_{t-1} - x^*||^2 - ||x_t - x^*||^2\big) \\ &= \frac{1}{\alpha_1}||x_0 - x^*||^2 - \frac{1}{\alpha_T}||x_T - x^*||^2 + \sum_{t=2}^T \left(\frac{1}{\alpha_t} - \frac{1}{\alpha_{t-1}}\right)||x_{t-1} - x^*||^2 \\ &\leq \frac{\delta^2}{\alpha_1} + \delta^2 \sum_{t=2}^T \left(\frac{1}{\alpha_t} - \frac{1}{\alpha_{t-1}}\right) = \frac{\delta^2}{\alpha_T}, \end{aligned} \tag{2.42}$$

where the inequality uses that the α_t's are positive and non-increasing together with the finite diameter of the set C that contains all considered points. Using convexity of f in combination with Eq.(2.41) and Eq.(2.42) we obtain

$$\begin{aligned} \mathbb{E}\big[f(\overline{x})\big] &\leq \frac{1}{T}\sum_{t=1}^T \mathbb{E}[f(x_{t-1})] \\ &\leq f(x^*) + \frac{1}{2T}\left(\frac{\delta^2}{\alpha_T} + \gamma^2 \sum_{t=1}^T \alpha_t\right). \end{aligned}$$

Finally, after inserting $\alpha_t = \delta/(\sqrt{2t}\gamma)$, we can simplify the expression by using that $\sum_{t=1}^T t^{-1/2} \leq \int_0^T t^{-1/2} dt = 2\sqrt{T}$, which then leads to Eq.(2.40). □

2.9 (Un)reasonable effectiveness—optimization

We begin with two examples that show how hard problems can arise—even in the absence of a complex architecture and without a large number of non-global minima or saddle points.

NP-hardness of empirical risk minimization Consider an arbitrary graph $G = (V, E)$ whose vertices are numbered so that $V = \{1, \ldots, d\}$. Assign a set $S_G \in \big\{\{0,1\}^{|V|} \times \{0,1\}\big\}^n$ with $n := |V| + |E| + 1$ to the graph in the following way: denoting by $e_i \in \{0,1\}^{|V|}$ the unit vector whose i'th component is equal to one, we set $S_G = \{(e_i, 0), (e_i + e_j, 1), (0, 1)\}_{i \in V, (i,j) \in E}$.

Recall that G is called 3-*colorable* iff there exists a map $\chi : V \to \{1,2,3\}$ with the property that $(i,j) \in E \Rightarrow \chi(i) \neq \chi(j)$. That is, there is an assignment of 'colors' to vertices such that no pair connected by an edge has the same color.

Proposition 2.11. *Consider feedforward neural networks with d inputs, a single hidden layer with three neurons and a single output neuron. Assume all activation functions are $\sigma(z) = \mathbb{1}_{z\geq 0}$ and that the output neuron has all weights and the threshold fixed so that it acts as $x \mapsto \sigma(\sum_{i=1}^3(x_i-1))$. Let $\mathcal{F}_d \subseteq \{0,1\}^{\mathbb{R}^d}$ be the function class that can be represented by such networks. Then for any graph G with d vertices there is an $f \in \mathcal{F}_d$ that correctly classifies S_G iff G is 3-colorable.*

Proof. Note first that the output neuron is set up so that it 'fires' iff all three hidden neurons do so. Assume G is 3-colorable via $\chi : V \to \{1,2,3\}$. Choose the weight $w_{l,i}$ that connects the i'th input and the l'th hidden neuron so that $w_{l,i} = -1$ if $\chi(i) = l$ and $w_{l,i} = 1$ otherwise. The threshold values of the three hidden neurons are all set to $1/2$, which leads to an $f \in \mathcal{F}_d$ that can be characterized by

$$f(x) = 1 \;\Leftrightarrow\; \forall l \in \{1,2,3\} : \sum_k w_{l,k}x_k \geq -\frac{1}{2}.$$

Now we have to verify that f correctly classifies S_G. Clearly, $f(0) = 1$. It also holds that $f(e_i) = 0$ since if $\chi(i) = l$, then $w_{l,i} = -1$ so that $\sum_k w_{l,k}(e_i)_k = w_{l,i} \not\geq -1/2$. In order to verify $f(e_i + e_j) = 1$ for all $(i,j) \in E$, note that for any $l \in \{1,2,3\}$ we have $\chi(i) \neq l \vee \chi(j) \neq l$ since χ is a coloring. Therefore $\sum_k w_{l,k}(e_i + e_j)_k = w_{l,i} + w_{l,j}$ is non-negative for all l.

Let us now show the converse implication and assume that there is an $f \in \mathcal{F}_d$ that correctly classifies S_G. Associating a half space H_l to each of the hidden Perceptrons we can express this assumption as $f^{-1}(\{1\}) = H_1 \cap H_2 \cap H_3 =: H$ where $0 \in H$, $\forall (i,j) \in E : \; e_i + e_j \in H$ and $\forall i \in V : e_i \notin H$. We define $\chi(i) := \min\{l | e_i \notin H_l\}$ and claim that this is a 3-coloring. First note that due to convexity of H and the fact that H contains the origin, we have $(e_i + e_j)/2 \in H$ for every edge $(i,j) \in E$. Suppose, aiming at a contradiction, that there would be an edge for which $\chi(i) = \chi(j) = l$. Then since $e_i, e_j \notin H_l$ this would, again by convexity, imply that $(e_i + e_j)/2 \notin H_l$ – a contradiction. □

Via reduction from 3-SAT, the 3-coloring problem is known to be NP-complete. Hence, the above Proposition shows that (NP-)hard problems can already be found in instances of empirical risk minimization for neural networks with very simple architecture. However, admittedly, the example is of combinatorial nature and uses an activation function that has been practically abandoned—essentially for this reason. So let us consider a 'smoother' problem ...

NP-hardness of classifying stationary points An instructive example for understanding, when and which problems can be hard, is given by homogeneous quartic polynomials. For a symmetric matrix $Q \in \mathbb{Z}^d \times \mathbb{Z}^d$, define $f : \mathbb{R}^d \to \mathbb{R}$ as

$$f(x) := \sum_{i,j=1}^d Q_{i,j} x_i^2 x_j^2. \tag{2.43}$$

At $x = 0$ both the gradient and the Hessian of f are zero. Hence, $x = 0$ is a stationary point, but the Hessian does not provide any information about whether it is a minimum, maximum or merely a saddle point. We do know, however, that there is a global minimum at $x = 0$ if there is a local minimum: suppose it not global, i.e., there is an x with $f(x) < 0$, then $\mathbb{R} \ni \lambda \mapsto f(\lambda x) = \lambda^4 f(x)$ shows that it cannot be a local minimum, either. Moreover, it shows that f is unbounded from below iff 0 is not a local minimum. Consequently, the two following problems are equivalent:

P1 Does f have a saddle point at 0 that is not a local minimum?

P2 Is f unbounded from below?

A negative answer to both these questions is a property of the matrix Q that is called *copositivity.* That is, Q is copositive by definition if $\langle z, Qz \rangle \geq 0$ for all entrywise non-negative z. By relating the problem to one (e.g. the subset-sum problem) that is known to be NP complete, one can prove that showing that Q is not copositive is NP complete, as well [**?**]. Hence, despite the apparent simplicity of f, both P1 and P2 are NP-complete problems. Note that the hardness in this case does not come from a large number of non-global minima or saddle points. It is simply the growing dimension that makes the problem hard.

Saddle points

Lemma 2.12 (Center-stable manifold theorem [**?**, **?**])**.** *Let $g : \mathbb{R}^d \to \mathbb{R}^d$ be a local C^1-diffeomorphism with fixed point $z = g(z)$. Let the Jacobian $Dg(z)$ have k eigenvalues (counting algebraic multiplicities) of modulus less than or equal to one. Then there is a $k-$dimensional manifold $W_z \subseteq \mathbb{R}^d$ and an open ball B_z around z s.t. $g(W_z) \cap B_z \subseteq W_z$ and if $g^t(x) \in B_z$ holds for all $t \in \mathbb{N}_0$, then $x \in W_z$.*

W_z is called *center-stable manifold.*

Lemma 2.13 (Gradient descent update is a diffeomorphism)**.** *Let $f \in C^2(\mathbb{R}^d)$ be such that ∇f is L-Lipschitz w.r.t. the Euclidean norm. If $\alpha \in (0, 1/L)$, then $g(x) := x - \alpha \nabla f(x)$ defines a C^1-diffeomorphism on $\mathbb{R}^d$.*

Proof. We first prove injectivity. Suppose $g(x) = g(y)$, which is equivalent to $(x - y)/\alpha = \nabla f(x) - \nabla f(y)$. Taking norms and using the Lipschitz property of the gradient, this implies $||x - y||/\alpha \leq L||x - y||$. Since $1/\alpha > L$ this can only hold if $x = y$. So g is injective.

It remains to show that g, which is C^1 by construction, is a local C^1-diffeomorphism. To this end, note first that ∇f being L-Lipschitz is equivalent to $\forall x : \nabla^2 f(x) \leq L\mathbb{1}$, i.e., the eigenvalues of the Hessian being not larger than L. This implies that the Jacobian of g, which is $Dg(x) = \mathbb{1} - \alpha \nabla^2 f(x)$ is invertible. By the inverse function theorem, g is thus a local C^1-diffeomorphism. □

Theorem 2.14: Almost no convergence to strict saddle points

Let $f \in C^2(\mathbb{R}^d)$ be such that ∇f is L-Lipschitz w.r.t. the Euclidean norm and define $g(x) := x - \alpha \nabla f(x)$ for some $\alpha \in (0, 1/L)$. Let $\mathcal{S} \subseteq \mathbb{R}^d$ be the set of stationary points of f for which the Hessian has at least one negative eigenvalue. Then $\mathcal{S}_\infty := \{x \in \mathbb{R}^d | \exists z \in \mathcal{S} : \lim_{t\to\infty} g^t(x) = z\}$ has Lebesgue measure zero.

Proof. First note that if $z \in \mathcal{S}$ is such a stationary point, then the Jacobian $Dg(z)$ has strictly less than d eigenvalues of modulus at most one. Consequently, the manifold W_z that corresponds to z in the center-stable manifold theorem (Lemma 2.12) has reduced dimension. Define $B := \bigcup_{z\in\mathcal{S}} B_z$ with the balls from Lemma 2.12. By the Lindelöf covering theorem, there is a countable subcover. That is, there exist $z_i \in \mathcal{S}$ so that $\mathcal{S} \subseteq B = \bigcup_{i\in\mathbb{N}} B_{z_i}$. If $g^t(x)$ converges to $z \in \mathcal{S}$, then $z \in B_{z_i}$ for some $z_i \in \mathcal{S}$ and there is a $\tau \in \mathbb{N}$ so that for all $t \geq \tau$ we have that $g^t(x) \in B_{z_i}$. Therefore, by Lemma 2.12, $g^\tau(x) \in W_{z_i}$, which means that $x \in g^{-\tau}(W_{z_i})$. Arguing like this for all $x \in \mathcal{S}_\infty$ we obtain $\mathcal{S}_\infty \subseteq \bigcup_{i\in\mathbb{N}} \bigcup_{\tau\in\mathbb{N}} g^{-\tau}(W_{z_i})$. This is a countable union of sets of measure zero (as the differentiable map $g^{-\tau}$ maps nullsets to nullsets), so it has measure zero, as well. □

Non-global minima ... to be written ...

2.10 Deep neural nets

... to be completed. Missing: history, vanishing gradient problem, convolutional and residual nets ...

Representation benefits of deep networks Define $\mathcal{F}(m, l) \subseteq \mathbb{R}^\mathbb{R}$ as the set of functions that can be represented by a feedforward neural network with l layers, m neurons within every hidden layer, a single neuron at the output and the rectified linear unit $\sigma_R(z) := \max\{0, z\}$ as activation function. In order to make an $f \in \mathcal{F}(m, l)$ into a classifier, define $\tilde{f}(x) := \mathbb{1}_{f(x)\geq 1/2}$ and let $\hat{R}(f) := \frac{1}{|S|} \sum_{(x,y)\in S} \mathbb{1}_{\tilde{f}(x)\neq y}$ be the corresponding empirical risk w.r.t. a training data set S.

Theorem 2.15: Exponential benefit of deep networks

Let $k \in \mathbb{N}$, $n = 2^k$ and $S := \big((x_i, y_i)\big)_{i=0}^{n-1}$ with $x_i := i/n$ and $y_i := i \bmod 2$.

1. There is an $h \in \mathcal{F}(2, 2k)$ for which $\hat{R}(h) = 0$.
2. If $m, l \in \mathbb{N}$ and $m \leq 2^{\frac{k-2}{l}-1}$, then $\hat{R}(f) \geq \frac{1}{6}$ holds for all $f \in \mathcal{F}(m, l)$.

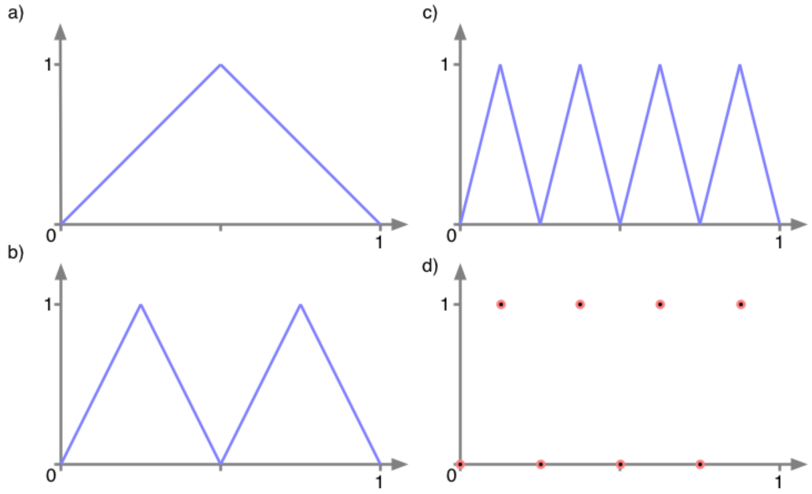


Figure 2.5: Fig. a) - c) show the graphs of g, g^2 and g^3. d) shows the set S for $k = 3$. By construction, it is part of the graph of g^3.

Proof. 1. Define a function $g : \mathbb{R} \to \mathbb{R}$ as

$$g(x) := \begin{cases} 2x, & 0 \leq x \leq 1/2, \\ 2(1-x), & 1/2 < x \leq 1, \\ 0, & \text{otherwise.} \end{cases} \tag{2.44}$$

Since this can be written as $g(x) = \sigma_R\big(2\sigma_R(x) - 4\sigma_R(x-1/2)\big)$ we have $g \in \mathcal{F}(2,2)$. If we compose the function k-times with itself, the graph of $h(x) := g^k(x)$ is a saw-tooth with 2^{k-1} 'teeth' (see Fig.2.5). By construction, $h \in \mathcal{F}(2,2k)$ and $h(x_i) = y_i$ for all $i = 1, \ldots n$. Hence, $\hat{R}(h) = 0$.

2. Every $f \in \mathcal{F}(m,l)$ is piecewise affine with at most $(2m)^l$ pieces. This is a consequence of the following simple fact: suppose f_1 and f_2 are piecewise affine with t_1 and t_2 pieces, respectively. Then $f_1 + f_2$ and $f_1 \circ f_2$ are again piecewise affine with at most $t_1 + t_2$ and $t_1 t_2$ pieces.

With at most $t = (2m)^l$ affine and thus monotone pieces, the graph of a function $f \in \mathcal{F}(m,l)$ crosses $1/2$ at most $2t - 1$ times: not more than once inside every interval and possibly once from one interval to the next. Therefore, $\tilde{f}$ is piecewise constant with at most $2t$ intervals with values 0 or 1. Let us now consider how the n points, whose values alternate between zero and one, can be distributed over these $2t$ intervals. Clearly, at most $2t$ points can be in intervals that contain no more than one point. The other $n - 2t$ points have to be in intervals that contain more than one point. At least one third of these points are thus misclassified so that the empirical risk can be bounded from below as

$$\hat{R}(f) \geq \frac{n-2t}{3n} \geq \frac{1}{3} - \frac{2}{3n}(2m)^l,$$

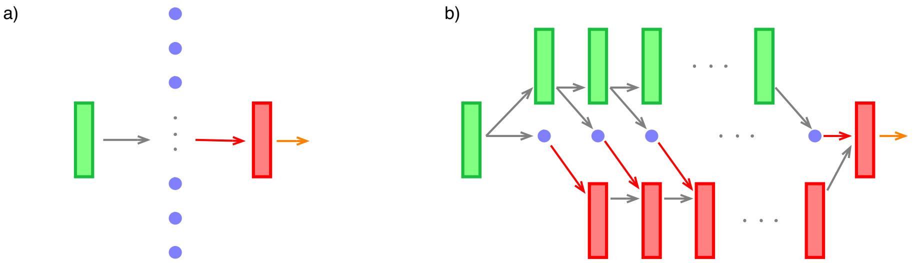


Figure 2.6: a) Schematic representation of a neural network with a single hidden layer containing m neurons. Green an red box depict the input and output layer, respectively. The links/arrows are only drawn pars pro toto. b) The same functions can be represented using a layered network of constant width and depth $m+1$. The construction 'rotates' the hidden layer by 90° and adds an 'input bus' and an 'output bus' whose purposes are to keep a copy of the input and to sequentially collect the output, respectively.

which is at least $1/6$ if $m \leq 2^{(k-2)/l-1}$. □

This rises the question whether deep networks are more expressive than shallow ones. Thm. 2.15 shows that at least some functions can be represented by deep networks using exponentially fewer parameters, but could the opposite happen, as well? the following answers this in the negative, at least for the extreme case where the shallower network under consideration has just a single hidden layer.

Proposition 2.14 (Conversion from wide & shallow to narrow & deep)**.** *Consider the class of layered feed-forward neural networks with d input and l output nodes and activation functions that are taken from a fixed set that includes the identity $\sigma(z) = z$. Within this class, every function representable with a single hidden layer of m neurons admits a representation by a network with m hidden layers each of which contains at most $d + l + 1$ neurons.*

This is easily seen by rotating the hidden layer and adding two parallel busses whose sizes equal the ones of the input and output layer, as depicted in Fig.2.6. There, matching colors of the arrows indicate matching types of activation functions, where gray arrows mean that the identity function is used, so that the information is passed along from one layer to the next. For neural nets using, for instance, the ReLU activation function we obtain the following:

Theorem 2.16: Representation via deep, narrow networks

Let $K \subseteq \mathbb{R}^d$ be compact. Every continuous function $f : K \rightarrow \mathbb{R}^l$ can be approximated arbitrary well in $||\cdot||_\infty$ by a layered feed-forward neural network of width at most $d + l + 1$ using a continuous activation function σ, if the latter is not affine but contains a non-constant affine piece. Here, the output layer is assumed to use the identity as activation function.

Proof. From Thm.2.6 we know that such functions can be arbitrary well approximated by a shallow network using such activation functions and a single hidden layer. Using Prop.2.14 we can convert this into a deep network of width at most $d + l + 1$ when allowing for identity activation functions. The latter can, however, be effectively implemented using the given activation function σ and resorting to compactness. This is seen by observing that for any bounded domain there are affine maps A, B so that $A \circ \sigma \circ B$ becomes the identity on that domain. The required affine maps can be implemented by adjusting weights (for the linear part) and biases (for the offset) in the network. □

2.11 (Un)reasonable effectiveness—generalization

... to be written ...

Chapter 3

Support Vector Machines

3.1 Linear maximal margin separators

Separable case. Consider a real Hilbert space $\mathcal{H}$ and a training data set $S = \big((x_i, y_i)_{i=1}^n\big) \in \big(\mathcal{H} \times \{-1, 1\}\big)^n$. Suppose the two subsets of points corresponding to the labels ±1 can be separated by a hyperplane H. That is, there are $w \in \mathcal{H}$ and $b \in \mathbb{R}$ that characterize the hyperplane via $H = \{x \in \mathcal{H} \mid \langle w, x\rangle + b = 0\}$ so that $\forall i : \operatorname{sgn}\big(\langle w, x_i\rangle + b\big) = y_i$. If there is no point exactly on the hyperplane this is equivalent to

$$y_i\big(\langle w, x_i\rangle + b\big) > 0 \quad \forall i. \tag{3.1}$$

The separating hyperplane is not unique and the question arises, which separating hyperplane to choose. The standard approach in the SVM framework is to choose the one that maximizes the distance to the closest points on both sides. In order to formalize this, we need the following Lemma.

Lemma 3.1 (Distance to a hyperplane)**.** *Let $\mathcal{H}$ be a Hilbert space and $H := \{z \in \mathcal{H} \mid \langle z, w\rangle + b = 0\}$ a hyperplane defined by $w \in \mathcal{H}$ and $b \in \mathbb{R}$. The distance of a point $x \in \mathcal{H}$ to H is given by*

$$d(x, H) := \inf_{z \in H} ||x - z|| = \frac{|\langle x, w\rangle + b|}{||w||}. \tag{3.2}$$

Proof. Let us first determine the distance of an arbitrary hyperplane to the origin: since $\inf_{z\in H} ||z||$ is attained for $z = -bw/||w||^2$ we get that $d(0, H) = |b|/||w||$. Using that translations are isometries, we can rewrite $d(x, H) = d(0, H - x)$ and apply the previous observation to the hyperplane $H - x = \{z | \langle z, w\rangle + b' = 0\}$ with $b' := \langle x, w\rangle + b$. □

Using Lemma 3.1 and Eq.(3.1) we can write the distance between a separating hyperplane and the closest point in S as

$$\rho := \min_i d(x_i, H) = \frac{\min_i y_i\big(\langle w, x_i\rangle + b\big)}{||w||}. \tag{3.3}$$

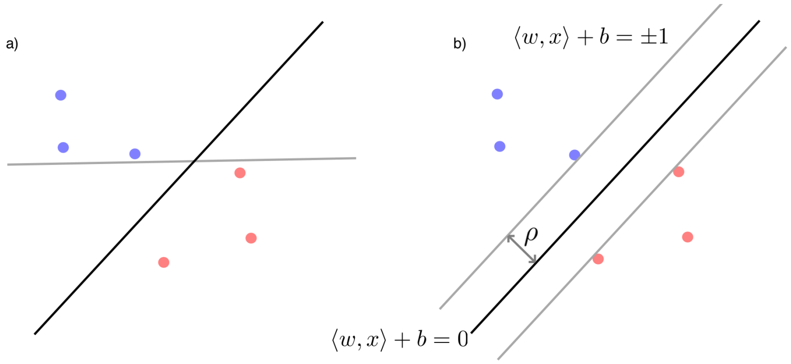


Figure 3.1: a) Red and blue points are separated by both hyperplanes. b) The black hyperplane is the one that maximizes the *margin* ρ. If the two margin hyperplanes are characterized by $\langle w,x\rangle+b=\pm1$, then $\rho=1/||w||$.

ρ is called the *margin* of the hyperplane w.r.t. S and the aim is now to determine the hyperplane that maximizes the margin. To this end, note that there is a scalar freedom in the characterization of the hyperplane: if we multiply both w and b by a positive scalar, then the hyperplane is still the same and also the margin does not change. We can now use this freedom to fix either the denominator in Eq.(3.3) or the enumerator and in this way obtain two different albeit equivalent constrained optimization problems. Constraining the denominator for instance leads to

$$\max_{(b,w)} \rho \;=\; \max_{(b,w):||w||\leq 1} \; \min_i \; y_i\big(\langle w,x_i\rangle+b\big).$$

Assuming that the sets of points that correspond to the two labels are not empty, a maximum is attained since the closed unit ball in a Hilbert space is weakly compact. So writing max instead of sup is indeed justified.

Alternatively, in order to obtain the hyperplane that maximizes the margin, we may use the mentioned scalar freedom to impose a constraint on the enumerator in Eq.(3.3) and minimize the denominator $||w||$ or, for later convenience, $||w||^2/2$, which leads to the same minimizer. That is, the maximal margin hyperplane is the one that achieves the minimum in

$$\min_{(b,w)} \frac{1}{2}||w||^2 \quad \text{s.t.} \quad \forall i: \; y_i\big(\langle w,x_i\rangle+b\big)\geq 1. \tag{3.4}$$

This is an optimization problem with strictly convex target function and affine inequality constraints. Due to strict convexity the minimum is unique. We further apply a standard tool from convex optimization:

Proposition 3.2 (Convex KKT)**.** *Let $\mathcal{H}$ be a real Hilbert space, $\{f_i:\mathcal{H}\to\mathbb{R}\}_{i=0}^n$ a set of continuously differentiable convex functions and assume that there is a $z\in\mathcal{H}$ for which $f_i(z)<0$ holds for all $i=1,\dots,n$. Then for every $\tilde z$ that satisfies $f_i(\tilde z)\leq 0$ for all $i=1,\dots,n$ the following are equivalent:*

1. $f_0(\tilde{z}) = \min_{z\in\mathcal{H}}\{f_0(z) \mid f_i(z) \leq 0 \;\forall i = 1, \ldots, n\}$.

2. *There exist* $\lambda_i \leq 0$ *so that*

$$\begin{aligned} \nabla f_0(\tilde{z}) &= \sum_{i=1}^n \lambda_i \nabla f_i(\tilde{z}) \;\textit{and} && (3.5)\\ \lambda_i f_i(\tilde{z}) &= 0 \quad \forall i = 1, \ldots, n. && (3.6) \end{aligned}$$

Applying this to the optimization problem in Eq.(3.4) leads to the following crucial insight: if $\tilde{w}$ corresponds to the maximal margin hyperplane, then Eq.(3.5) implies $\tilde{w} = \sum_{i=1}^n y_i\lambda_i x_i$. That is, the minimizing $\tilde{w}$ is a linear combination of the training data points x_i. In addition, Eq.(3.6), which in our case reads $\lambda_i\big[1 - y_i\big(\langle\tilde{w}, x_i\rangle + b\big)\big] = 0$, implies that only those x_i's contribute for which the i'th constraint is *active*. This means $y_i\big(\langle\tilde{w}, x_i\rangle + b\big) = 1$ so that the corresponding x_i is sitting on one of the two margin hyperplanes. These x_i's are called *support vectors*.

Non-separable case Now we drop the assumption that the data is exactly linearly separable. However, we still seek a predictor that is given in terms of a hyperplane and that in some sense still has maximal margin. The difference to the foregoing discussion is that we now allow for outliers that may either be on the wrong side of the hyperplane or inside the margin. In order to formalize this, one introduces *slack variables* $\xi_i \geq 0$ that measure the extent to which the i'th constraint is violated. In addition, one penalizes these violations in the object function. This leads to the optimization problem

$$\begin{aligned} &\min_{(b,w,\xi)} \frac{\lambda}{2}||w||^2 + \frac{1}{n}\sum_{i=1}^n \xi_i \\ &\text{s.t.} \;\; y_i\big(\langle w, x_i\rangle + b\big) \geq 1 - \xi_i \quad\wedge\quad \xi_i \geq 0 \;\; \forall i = 1, \ldots n, \end{aligned} \qquad (3.7)$$

where $\lambda > 0$ is a free parameter that can be used to adjust the strength of the penalty. There is some arbitrariness in how one penalizes large ξ. In Eq.(3.7) we have essentially chosen the l_1-norm of ξ. Another common choice would be the l_2-norm.

The optimization problem in Eq.(3.7) can be written as ERM problem w.r.t. the so-called *hinge loss* $L_{hinge} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ that is defined as

$$L_{hinge}(y, y') := \max\{0, 1 - yy'\}.$$

The hinge loss provides an upper bound on the usually taken loss function for binary classification in the sense that if $y \in \{-1, 1\}$, then $\mathbb{1}_{y\neq\mathrm{sgn(h(x))}} \leq L_{hinge}(y, h(x))$. Other noticeable properties are that $y' \mapsto L_{hinge}(y, y')$ is convex and $w \mapsto L_{hinge}(y, \langle w, x\rangle + b)$ is $||x||$-Lipschitz.

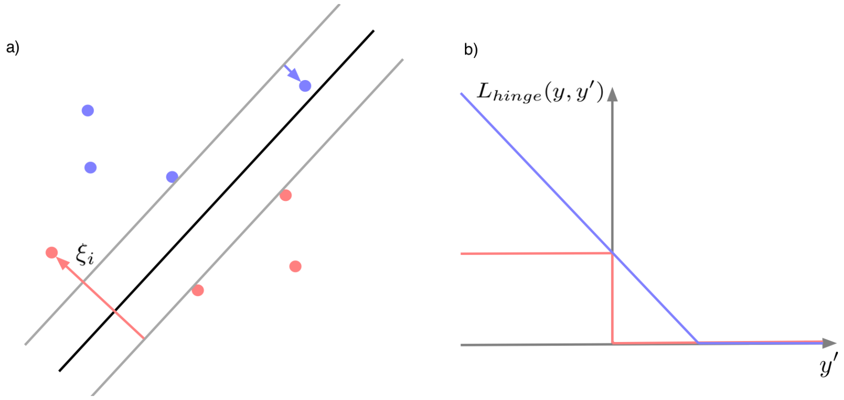


Figure 3.2: a) Outliers (points that are either inside the margin corridor, or on the wrong side) are penalized using *slack variable* ξ_i. b) The *hinge loss* (blue), plotted for the case $y = 1$, is a convex upper bound for the 0-1-loss (red) that is usually used for binary classification.

The optimization problem in Eq.(3.7) can now be written as

$$\min_{(b,w)} \frac{\lambda}{2}||w||^2 + \frac{1}{n}\sum_{i=1}^{n} \max\left\{0, 1 - y_i\big(\langle w, x_i\rangle + b\big)\right\}$$

$$= \min_{(b,w)} \frac{\lambda}{2}||w||^2 + \hat{R}_{hinge}(h), \tag{3.8}$$

where $h(x) := \langle w, x\rangle + b$. Note that Eq.(3.8) is a regularized ERM problem without additional constraints.

Theorem 3.1: Representer theorem

Let $\mathcal{H}$ be a Hilbert space, $g : \mathbb{R} \to \mathbb{R}$ non-decreasing, $f : \mathbb{R}^n \to \mathbb{R}$, $\{x_1, \ldots, x_n\} \subseteq \mathcal{H}$, $\mathcal{H}_x := \text{span}\{x_i\}_{i=1}^n$ and $F : \mathcal{H} \to \mathbb{R}$, $F(w) := g(||w||) + f(\langle w, x_1\rangle, \ldots, \langle w, x_n\rangle)$. Then

$$\inf_{w\in\mathcal{H}} F(w) = \inf_{w\in\mathcal{H}_x} F(w) \tag{3.9}$$

and if g is strictly increasing, then every minimizer of the l.h.s. of Eq.(3.9) is an element of $\mathcal{H}_x$.

Proof. We use that $\mathcal{H} = \mathcal{H}_x \oplus \mathcal{H}_x^\perp$ and that every $w \in \mathcal{H}$ admits a corresponding decomposition of the form $w = w_x + v$ where $w_x \in \mathcal{H}_x$ and $v \in \mathcal{H}_x^\perp$. Then $\langle w, x_i\rangle = \langle w_x, x_i\rangle$ holds for all i and from Pythagoras we obtain

$$g(||w||) = g\Big(\sqrt{||w_x||^2 + ||v||^2}\Big) \geq g(||w_x||).$$

Here, strict inequality holds if g is strictly increasing and $w \neq w_x$. Hence, the claims follow by replacing w by w_x in the argument of F. □

3.2 Positive semidefinite kernels

Definition 3.3 (PSD kernel). *Let $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$ and X be an arbitrary set. A map $K : X \times X \to \mathbb{K}$ is called* positive semidefinite kernel *(PSD kernel) iff for all $n \in \mathbb{N}$ and all $x \in X^n$ the $n \times n$ matrix G with entries $G_{ij} := K(x_i, x_j)$ is positive semidefinite.*

The terminology varies considerably throughout the literature. PSD kernels also run under the names positive definite kernels, positive definite symmetric kernels, kernel functions or just kernels. Recall that a matrix G is positive semidefinite iff G is hermitian, i.e., $G_{ij} = \bar{G}_{ji}$, and G has only non-negative eigenvalues. The latter condition can be replaced with

$$\forall \alpha \in \mathbb{K}^n : \sum_{i,j=1}^{n} \bar{\alpha}_i \alpha_j G_{ij} \geq 0. \tag{3.10}$$

Theorem 3.2: PSD kernels and feature maps

Let X be any set, $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$ and $K : X \times X \to \mathbb{K}$.

1. K is a PSD kernel, if there is an inner product space $\mathcal{H}$ and a map $\phi : X \to \mathcal{H}$ so that

$$K(x, y) = \langle \phi(x), \phi(y) \rangle \qquad \forall x, y \in X. \tag{3.11}$$

2. Conversely, if K is a PSD kernel, then there exists a Hilbert space $\mathcal{H}$ and a map $\phi : \mathcal{H} \to \mathbb{K}$ so that Eq.(3.11) holds.

Note: the map ϕ is often called *feature map* and the inner product space $\mathcal{H}$ the *feature space*.

Proof. 1. If K is of the form in Eq.(3.11), then for all $\alpha \in \mathbb{K}^n$ and $x \in X^n$ we have $\sum_{i,j=1}^{n} \bar{\alpha}_i \alpha_j \langle \phi(x_j), \phi(x_i) \rangle = \langle \Phi, \Phi \rangle \geq 0$ where $\Phi := \sum_{i=1}^{n} \alpha_i \phi(x_i)$. Hermiticity of the respective matrix follows from hermiticity of the inner product.

2. Assume K to be a PSD kernel and define

$$\mathcal{H}_0 := \operatorname{span} \big\{ k_x : X \to \mathbb{K} \mid \exists x \in X : k_x(y) = K(x, y) \big\} \tag{3.12}$$

the space of all finite $\mathbb{K}$-linear combination of functions of the form $y \mapsto K(x, y)$. We aim at equipping this space with an inner product. For two arbitrary elements of $\mathcal{H}_0$ given by $f(y) := \sum_i \alpha_i K(x_i, y)$ and $g(y) := \sum_j \beta_j K(x_j, y)$ define

$$\begin{aligned} \langle f, g \rangle &:= \sum_{i,j} \alpha_i \bar{\beta}_j K(x_i, x_j) \\ &= \sum_i \alpha_i \overline{g(x_i)} \;=\; \sum_j \bar{\beta}_j f(x_j), \end{aligned}$$

where the second line shows that the definition is independent of the particular decomposition of f or g. So $\langle \cdot, \cdot \rangle$ is a well defined hermitian sesquilinear form on $\mathcal{H}_0$. Moreover, since K is a PSD kernel, we have $\langle g, g \rangle \geq 0$ for all $g \in \mathcal{H}_0$. Hence, the Cauchy Schwarz inequality holds. Applying it to

$$f(y) = \sum_i \alpha_i K(x_i, y) = \langle f, k_y \rangle, \tag{3.13}$$

we obtain $|f(y)|^2 = |\langle f, k_y \rangle|^2 \leq \langle k_y, k_y \rangle \langle f, f \rangle$. This shows that $\langle f, f \rangle = 0$ implies $f = 0$ and thus $\langle \cdot, \cdot \rangle$ is indeed an inner product. Note that if we apply Eq.(3.13) to $f = k_x$, we obtain

$$k_x(y) = K(x, y) = \langle k_x, k_y \rangle. \tag{3.14}$$

So if we denote by $\mathcal{H}$ the completion of the inner product space $\mathcal{H}_0$ and define $\phi : X \to \mathcal{H}$ so that $\phi(x)$ is the isometric embedding of k_x into $\mathcal{H}$, then Eq.(3.14) implies $K(x, y) = \langle \phi(x), \phi(y) \rangle$. □

Proposition 3.4 (Building new PSD kernels)**.** *Let $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$, $K_1, K_2, \ldots$ PSD kernels on a set X and $f \in \mathbb{K}^X$. Then*

1. *$K(x, y) := f(x)\overline{f(y)}$ is a PSD kernel.*
2. *$K(x, y) := \lambda K_1(x, y)$ is a PSD kernel for all $\lambda \geq 0$.*
3. *$K(x, y) := K_1(x, y) + K_2(x, y)$ is a PSD kernel.*
4. *$K(x, y) := \lim_{n\to\infty} K_n(x, y)$ is a PSD kernel, if the limits exist in $\mathbb{K}$.*
5. *$K(x, y) := K_1(x, y)K_2(x, y)$ is a PSD kernel.*

Proof. In all cases hermiticity is rather obvious, so we only have a look at positive semidefiniteness. 1. K is PSD since $\sum_{i=1}^n \alpha_i \bar{\alpha}_j f(x_i)\overline{f(x_j)} = \big|\sum_{i=1}^n \alpha_i f(x_i)\big|^2$ is always positive. 2. and 3. are elementary consequences of the definition. 4. is implied by the closedness of the set of PSD matrices, or more explicitly by positivity of $\sum_{i=1}^m \alpha_i \bar{\alpha}_j K(x, y) = \lim_{n\to\infty} \sum_{i=1}^m \alpha_i \bar{\alpha}_j K_n(x, y)$ as a limit of positive numbers. 5. follows from the fact the set of PSD matrices is closed under taking element wise products (called *Schur products* or *Hadamard products*). □

With these tools at hand, many kernels can easily be shown to be PSD. Some of the most common examples are:

Example 3.1 (Polynomial kernels). On $X = \mathbb{R}^d$ any polynomial in $\langle x, y \rangle$ with non-negative coefficients is a PSD kernel as a consequence of 2., 3. and 5. in Prop. 3.4 together with the fact that $(x, y) \to \langle x, y \rangle$ is (the paradigm of) a PSD kernel. In particular, $K(x, y) := (1 + \langle x, y \rangle)^2$ is a PSD kernel. On $\mathbb{R}^2$ this can be obtained from the feature map $\phi(x) : \mathbb{R}^2 \to \mathbb{R}^6$, $\phi(x) := (x_1^2, x_2^2, \sqrt{2}x_1x_2, \sqrt{2}x_1, \sqrt{2}x_2, 1)$. Like in this example, all polynomial kernels have corresponding finite dimensional feature spaces.

Example 3.2 (Exponential kernels). For any $\gamma > 0$, $K(x, y) := \exp[\gamma \langle x, y \rangle]$ is a PSD kernel on $X = \mathbb{R}^d$ since it is a limit of polynomial kernels so that 4. in Prop. 3.4 applies.

Example 3.3 (Gaussian kernels). The Gaussian kernel $K(x,y) := \exp\big[-\frac{\gamma}{2}||x-y||^2\big]$ with the Euclidean norm is a PSD kernel on $X = \mathbb{R}^d$ for all $\gamma > 0$. To see this write

$$\exp\big[-\frac{\gamma}{2}||x-y||^2\big] = \underbrace{\exp[-\gamma||x||^2/2]\ \exp[-\gamma||y||^2/2]}_{f(x)\overline{f(y)}}\exp[\gamma\langle x,y\rangle]$$

and apply 1. and 5. of Prop. 3.4.

Example 3.4 (Binomial kernels). On $X := \{x \in \mathbb{R}^d \mid ||x||_2 < 1\}$ $K(x,y) := \big(1 - \langle x,y\rangle\big)^{-p}$ is a PSD kernel for any $p > 0$. This follows again from the previous proposition by noting that for $t \in (-1,1)$ the binomial series $(1-t)^{-p} = \sum_{n=0}^{\infty}(-1)^n\binom{-p}{n}t^n$ has positive coefficients $(-1)^n\binom{-p}{n} = (-1)^n\prod_{i=1}^{n}(1-p-i)/i$.

We will see in Sec.3.4 that, whereas polynomial kernels have finite dimensional feature spaces, exponential, Gaussian and binomial kernels require infinite dimensional feature space.

3.3 Reproducing kernel Hilbert spaces

For a given PSD kernel, the corresponding feature map and feature space are not unique. However, there is a canonical choice for the feature space, a so-called *reproducing kernel Hilbert space*.

Definition 3.5 (Reproducing kernel Hilbert space). *Let X be a set, $\mathbb{K} \in \{\mathbb{R},\mathbb{C}\}$ and $\mathcal{H} \subseteq \mathbb{K}^X$ a $\mathbb{K}$-Hilbert space of functions on X with addition $(f+g)(x) := f(x)+g(x)$ and multiplication $(\lambda f)(x) := \lambda f(x)$. $\mathcal{H}$ is called a* reproducing kernel Hilbert space *(RKHS) on X iff for all $x \in \mathcal{H}$ the linear functional $\delta_x : \mathcal{H} \to \mathbb{K}$, $\delta_x(f) := f(x)$ is bounded (i.e., $\sup_{f\in\mathcal{H}\setminus\{0\}} |f(x)|/||f|| < \infty$).*

Note: Since δ_x is linear, boundedness is equivalent to continuity. That is, the defining property of a RKHS is that evaluation of its functions at arbitrary points is continuous w.r.t. varying the function.

Example 3.5. If X is countable, then $l_2(X) := \{f \in \mathbb{K}^X \mid \sum_{x\in X}|f(x)|^2 < \infty\}$ equipped with $\langle f,g\rangle := \sum_{x\in X} f(x)\overline{g(x)}$ is a RKHS since for all $x \in X$ we have $|f(x)| \leq \big(\sum_{y\in X}|f(y)|^2\big)^{1/2} = ||f||$. Hence, δ_x is bounded.

Example 3.6. $L_2([0,1])$ is not a RKHS. Since its elements are equivalence classes of functions that differ on sets of measure zero, $f(x)$ is not defined. Even if we restrict to the subspace of continuous functions, where $f(x)$ is defined, its magnitude is not bounded by imposing $||f|| \leq 1$. So δ_x is not bounded.

Since $l_2(\mathbb{N})$ and $L_2([0,1])$ are isomorphic, these two examples show that Hilbert space isomorphisms do not necessarily preserve the RKHS property.

A crucial consequence of the continuity of δ_x in any RKHS is that one can invoke the Riesz representation theorem. This states that every continuous linear functional on a Hilbert space can be represented as inner product with a unique vector. In particular, if $\mathcal{H}$ is a RKHS, then for every $x \in X$ there is a

$k_x \in \mathcal{H}$ so that $f(x) = \langle f, k_x \rangle$ for all $f \in \mathcal{H}$. Since inner products are always continuous, this can be regarded as equivalent characterization of a RKHS. As k_x is an element of $\mathcal{H}$ and therefore a function on X, we can define $K : X \times X \to \mathbb{K}$, $K(x, y) := k_x(y)$. K is called the *reproducing kernel* of the RKHS $\mathcal{H}$. Using that $k_x(y)$ can itself be expressed in terms of an inner product with some element k_y, we obtain

$$K(x, y) = \langle k_x, k_y \rangle. \tag{3.15}$$

Before we relate reproducing kernel Hilbert spaces to PSD kernels, let us mention some elementary properties:

Proposition 3.6. *Let $\mathcal{H} \subseteq \mathbb{K}^X$ be a RKHS with reproducing kernel K and $k_x(y) = K(x, y)$. Let $f, f_n \in \mathcal{H}$ and $\delta_x(f) := f(x)$ for $x \in X$, $f \in \mathcal{H}$. Then*

1. *For all $x \in \mathcal{H}$ we have $||\delta_x||^2 = K(x, x)$.*
2. $\lim_{n\to\infty} ||f_n - f|| = 0 \;\Rightarrow\; \forall x \in X : \lim_{n\to\infty} f_n(x) = f(x)$.
3. $\operatorname{span}\{k_x \mid x \in X\}$ *is dense in $\mathcal{H}$.*

Proof. 1. follows with $f(x) = \langle f, k_x \rangle$ from

$$||\delta_x||^2 = \sup_{f \in \mathcal{H}\setminus\{0\}} \frac{|\langle k_x, f \rangle|^2}{||f||^2} = ||k_x||^2 = \langle k_x, k_x \rangle = K(x, x), \tag{3.16}$$

where the second equality is the one of the Cauchy Schwarz inequality. Similarly, also 2. is obtained from Cauchy Schwarz by noting that

$$\big| f_n(x) - f(x) \big| = \big| \langle k_x, f_n - f \rangle \big| \le ||k_x||\, ||f_n - f|| \to 0.$$

For 3. it suffices to show that there is no non-zero element that is orthogonal to the considered span. Indeed, suppose $f \in \mathcal{H}$ is orthogonal to all k_x, then for all $x \in X$ we have that $0 = \langle f, k_x \rangle = f(x)$, which means $f = 0$. □

Theorem 3.3: RKHS and PSD kernels

1. If $\mathcal{H}$ is a RKHS on X, then its reproducing kernel $K : X \times X \to \mathbb{K}$ is a PSD kernel.
2. Conversely, if $K : X \times X \to \mathbb{K}$ is a PSD kernel, then there is a unique RKHS $\mathcal{H} \subseteq \mathbb{K}^X$ so that K is its reproducing kernel.

Proof. 1. If K is the reproducing kernel of a RKHS $\mathcal{H}$, then by Eq.(3.15) and the properties of the inner product:

$$\begin{aligned} \forall x, y \in X : \quad K(x, y) &= \langle k_x, k_y \rangle = \overline{\langle k_y, k_x \rangle} = \overline{K(y, x)} \quad \text{and} \\ \sum_{i,j=1}^{n} \alpha_i \bar{\alpha}_j K(x_i, x_j) &= \sum_{i,j=1}^{n} \alpha_i \bar{\alpha}_j \langle k_{x_i}, k_{x_j} \rangle = \Big\| \sum_{i=1}^{n} \alpha_i k_{x_i} \Big\|^2 \ge 0. \end{aligned}$$

2. (sketch) The construction of the sought RKHS is the one in the proof of Thm.3.2. Eqs.(3.13,3.14) show that K fulfills the requirement of a reproducing kernel on $\mathcal{H}_0$. A more careful consideration shows that the relevant properties are indeed preserved when going from $\mathcal{H}_0$ to its completion $\mathcal{H}$.

To address uniqueness suppose $\mathcal{H}_1$ and $\mathcal{H}_2$ are two RKHS with reproducing kernel K. Following 3. in Prop.3.6 the space $\mathcal{H}_0 = \text{span}\{k_x | x \in X\}$ is dense in both $\mathcal{H}_1$ and $\mathcal{H}_2$. Moreover, if $f \in \mathcal{H}_0$ with $f(x) = \sum_i \alpha_i k_{x_i}$, then $||f||_l^2 = \sum_{i,j} \alpha_i \bar{\alpha}_j K(x_i, x_j)$ for $l = 1, 2$. Hence, the norms $||\cdot||_1$ and $||\cdot||_2$ coincide on $\mathcal{H}_0$.

Suppose $f \in \mathcal{H}_1$. Then there are $f_n \in \mathcal{H}_0$ so that $||f_n - f||_1 \to 0$. As $(f_n)_{n\in\mathbb{N}}$ is Cauchy in $\mathcal{H}_1$ it is also Cauchy in $\mathcal{H}_2$ and therefore there exist a $g \in \mathcal{H}_2$ so that $||f_n - g||_2 \to 0$. According to 2. in Prop.3.6 we have $f(x) = \lim_{n\to\infty} f_n(x) = g(x)$ for all $x \in X$. Hence, $f = g \in \mathcal{H}_2$ and consequently $\mathcal{H}_1 = \mathcal{H}_2$. Since the norms, and by polarization also the inner products, coincide on a dense subspace, the do so on its completion. □

3.4 Universal and strictly positive kernels

Definition 3.7 (Universal kernels)**.** *A PSD kernel $K : X \times X \to \mathbb{K}$ on a metric space X is called* universal *iff for all $\epsilon > 0$, all compact subsets $\tilde{X} \subseteq X$ and every continuous function $f : X \to \mathbb{K}$ there exists $g \in \text{span}\{k_x : X \to \mathbb{K} \mid \exists x \in X : k_x(y) = K(x, y)\}$ so that*

$$\big|g(x) - f(x)\big| \leq \epsilon \quad \forall x \in \tilde{X}. \tag{3.17}$$

Note that if $\phi : X \to \mathcal{H}$ is a feature map corresponding to K, then Eq.(3.17) means that there exists a $w \in \mathcal{H}$ so that

$$\big|\langle w, \phi(x)\rangle - f(x)\big| \leq \epsilon \quad \forall x \in \tilde{X}. \tag{3.18}$$

Corollary 3.8 (Universal kernels separate all compact subsets)**.** *Let $\phi : X \to \mathcal{H}$ be a feature map of a universal PSD kernel on a metric space X. For any pair of disjoint compact subsets $A_+, A_- \subseteq X$ there exists a $w \in \mathcal{H}$ so that for all $x \in A_+ \cup A_-$:*

$$\text{sgn}\langle w, \phi(x)\rangle = \begin{cases} +1, & x \in A_+ \\ -1, & x \in A_- \end{cases} \tag{3.19}$$

Proof. As the distance between A_+ and A_- is non-zero, we can extend the function $A_+ \cup A_- \ni x \mapsto \mathbb{1}_{x\in A_+} - \mathbb{1}_{x\in A_-}$ to a continuous function f on X. By universality there exists a $w \in \mathcal{H}$ for each $\epsilon \in (0, 1)$ so that $|\langle w, \phi(x)\rangle - f(x)| \leq \epsilon$ for all $x \in A_+ \cup A_-$. Hence,

$$\langle w, \phi(x)\rangle \begin{cases} \geq 1 - \epsilon, & x \in A_+ \\ \leq \epsilon - 1, & x \in A_- \end{cases} \tag{3.20}$$

Note that in this case the sets are separated with margin $(1 - \epsilon)/||w||$. □

Theorem 3.4: Taylor criterion for universality

Let $f(z) := \sum_{n=0}^{\infty} a_n z^n$ be a power series with radius of convergence $r \in (0,\infty]$ and $X := \{x \in \mathbb{R}^d \mid ||x||_2 < \sqrt{r}\}$. If $a_n > 0$ for all n, then $K : X \times X \to \mathbb{R},\ K(x,y) := f(\langle x,y\rangle)$ is a universal PSD kernel.

Proof. First note that K is well defined since $|\langle x,y\rangle| \leq ||x||_2||y||_2 < r$. Using multinomial expansion we can write

$$\begin{aligned} K(x,y) &= \sum_{n=0}^{\infty} a_n \left(\sum_{k=1}^{d} x_k y_k\right)^n = \sum_{n=0}^{\infty} a_n \sum_{\substack{k_1+\cdots+k_d=n\\ k_1,\ldots,k_d\geq 0}} \frac{n!}{k_1!\cdots k_d!}\prod_{i=1}^{d}(x_i y_i)^{k_i} \\ &= \sum_{k_1,\ldots,k_d\geq 0} \underbrace{a_{k_1+\ldots+k_d}\frac{(k_1+\ldots+k_d)!}{k_1!\cdots k_d!}}_{=:c_k} \prod_{i=1}^{d} x_i^{k_i} \prod_{j=1}^{d} y_j^{k_j}. \end{aligned} \tag{3.21}$$

This enables us to introduce a feature map $\phi : X \to l_2\big(\mathbb{N}_0^d\big)$ as $\phi_k(x) := \sqrt{c_k}\prod_{i=1}^{d} x_i^{k_i}$ for $k \in \mathbb{N}_0^d$ so that $K(x,y) = \langle\phi(x),\phi(y)\rangle$. Since all a_n's are strictly positive, the same holds true for all c_k's. Consequently, $\text{span}\{\phi_k\}_{k\in\mathbb{N}_0^d}$ is the space of all polynomials and by the Stone-Weierstrass theorem dense in the set of continuous functions on compact domains. The claim then follows from the observation that every finite linear combination of functions of the form $x \mapsto \phi_k(x)$ can be regarded as an inner product $\langle w,\phi(x)\rangle$ for some vector w. Since the latter has only finitely many non-zero components, it is indeed an element of $l_2\big(\mathbb{N}_0^d\big)$. □

Corollary 3.9. *On $X = \mathbb{R}^d$ the following are universal PSD kernels:*

1. *Exponential kernel:* $K(x,y) := \exp\big(\gamma\langle x,y\rangle\big)$, $\gamma > 0$.
2. *Gaussian kernel:* $K(x,y) := \exp\big(-\frac{\gamma}{2}||x-y||_2^2\big)$, $\gamma > 0$.

Proof. Universality of the exponential kernel follows directly from Thm.3.4 with $a_n = \tau^n/n!$. This in turn can be used to prove universality of the Gaussian kernel: if $\phi : X \to \mathcal{H}$ is a feature map of the exponential kernel, then $\tilde{\phi} : x \mapsto \phi(x)/||\phi(x)||$ is a feature map of the Gaussian kernel. Now take any compact subset $\tilde{X} \subseteq X$ and define $c := \sup_{x\in\tilde{X}} ||\phi(x)||^{-1}$. By universality of the exponential kernel, for every continuous function $f : X \to \mathbb{R}$ there is a $w \in \mathcal{H}$ so that

$$\Big|f(x)||\phi(x)|| - \langle w,\phi(x)\rangle\Big| \leq \frac{\epsilon}{c} \quad \forall x \in \tilde{X}.$$

Dividing by $||\phi(x)||$ and taking the supremum over $x \in \tilde{X}$ on the resulting r.h.s. leads to $\big|f(x) - \langle w,\tilde{\phi}(x)\rangle\big| \leq \epsilon$ for all $x \in \tilde{X}$. □

Proposition 3.10 (Strict positivity of universal kernels)**.** *Let $K : X \times X \to \mathbb{K}$ be a universal PSD kernel on a metric space X. Then K is strictly positive definite, i.e., for all $n \in \mathbb{N}$, every set of n distinct points $x_1,\ldots,x_n \in X$ and all $\alpha \in \mathbb{K}^n\backslash\{0\}$ we have $\sum_{i,j=1}^{n}\alpha_i\bar{\alpha}_j K(x_i,x_j) > 0$.*

Proof. Assume K is not strictly positive definite, i.e., $\sum_{i,j=1}^n \alpha_i \bar{\alpha}_j K(x_i, x_j) = 0$ for some $\alpha \in \mathbb{K}^n \backslash \{0\}$ and $x \in X^n$. Expressing this in terms of the canonical feature map $\phi : X \to \mathcal{H}$, where $\mathcal{H}$ is the corresponding RKHS, we obtain that $\sum_{i=1}^n \alpha_i \phi(x_i) = 0$ since it has vanishing norm. Now for an arbitrary function induced by the kernel via $g(x) := \sum_{j=1}^m \beta_j \langle \phi(x), \phi(y_j)\rangle$ we obtain $\sum_{i=1}^n \alpha_i g(x_i) = \sum_{i,j} \alpha_i \beta_j \langle \phi(x_i), \phi(y_j)\rangle = 0$. Hence, the set of functions induced by the kernel cannot be dense in the set of continuous functions on the compact set $\tilde{X} := \bigcup_{i=1}^n \{x_i\}$ since any continuous function f for which $\sum_{i=1}^n \alpha_i f(x_i) \neq 0$ cannot be approximated to arbitrary accuracy. So K cannot be universal. □

Proposition 3.11 (Properties of strictly positive definite kernels)**.** *Let $K : X \times X \to \mathbb{K}$ be a strictly positive definite kernel on a set X. That is, for all $n \in \mathbb{N}$, every set of n distinct points $x_1, \ldots, x_n \in X$ and all $\alpha \in \mathbb{K}^n \backslash \{0\}$ we have $\sum_{i,j=1}^n \alpha_i \bar{\alpha}_j K(x_i, x_j) > 0$ and $K(x_i, x_j) = \overline{K(x_j, x_i)}$. Then:*

1. *Every corresponding feature space is infinite dimensional.*
2. *Every corresponding feature map is injective.*
3. *If A_+, A_- are disjoint finite subsets of X and $\phi : X \to \mathcal{H}$ is any feature map corresponding to K, then there is a $w \in \mathcal{H}$ and $b \in \mathbb{R}$ so that*

$$\mathrm{Re}\langle w, \phi(x)\rangle \begin{cases} > b, & \textit{if } x \in A_+ \\ < b, & \textit{if } x \in A_- \end{cases} \tag{3.22}$$

Proof. 1. If $\phi : X \to \mathcal{H}$ is any feature map for K and $d := \dim(\mathcal{H}) < \infty$, then any set of $n > d$ vectors $\{\phi(x_i)\}_{i=1}^n$ is linearly dependent. Therefore, there is an $\alpha \in \mathbb{K}^n \backslash \{0\}$ so that $0 = \sum_{i,j=1}^n \alpha_i \bar{\alpha}_j \langle \phi(x_i), \phi(x_j)\rangle = \sum_{i,j=1}^n \alpha_i \bar{\alpha}_j K(x_i, x_j)$, which implies that K is not strictly positive definite.

2. As argued in the proof of 1., if $x \neq y$, then $\phi(x)$ and $\phi(y)$ have to be linearly independent. So in particular ϕ is injective.

3. The central observation is again linear independence of the set of vectors $\{\phi(x)\}_{x \in A_+ \cup A_-}$. If we define $C_\pm := \mathrm{conv}\{\phi(x)\}_{x \in A_\pm}$ as the convex hulls of the images of the sets A_+ and A_- under ϕ, then linear independence implies that C_+ and C_- are disjoint sets. Moreover, they are closed and bounded convex subsets contained in finite dimensional subspace so that we can invoke the geometric Hahn-Banach separation theorem for compact convex sets to arrive at Eq.(3.22). □

Theorem 3.5: Translation invariant kernels

Let μ be a finite non-negative Borel measure on $X := \mathbb{R}^d$ and denote by $\chi \in C(X)$ its Fourier transform

$$\chi(x) := \int_X e^{-ix \cdot z} d\mu(z). \tag{3.23}$$

Then $K(x, y) := \chi(x - y)$ is a PSD kernel on X. Moreover, K is strictly positive definite, if the complement of the largest open set $U \subseteq X$ that satisfies $\mu(U) = 0$ has non-zero Lebesgue measure.

Proof. Consider distinct points $x_1, \ldots, x_n \in X$ and $\alpha \in \mathbb{C}^n \setminus \{0\}$. Then

$$\begin{aligned} \sum_{k,j=1}^{n} \alpha_k \bar{\alpha}_j K(x_k, x_j) &= \sum_{k,j=1}^{n} \alpha_k \bar{\alpha}_j \int_X e^{-i(x_k - x_j)\cdot z} d\mu(z) \\ &= \int_X \underbrace{\left| \sum_{k=1}^{n} \alpha_k e^{-i x_k \cdot z} \right|^2}_{=:\psi(z)} d\mu(z) \ \geq 0. \end{aligned} \tag{3.24}$$

So K is a PSD kernel. Moreover, strict inequality holds in Eq.(3.24) unless the support of μ is contained in the zero set $\psi^{-1}(\{0\})$. However, $\psi^{-1}(\{0\})$ always has zero Lebesgue measure so that every μ whose support has non-zero Lebesgue measure leads to a strictly positive definite kernel. □

3.5 Rademacher bounds

... to be completed ...

Theorem 3.6: Rademacher bound for bounded inner products

Let $\rho, r > 0$ be positive constants, $x_1, \ldots, x_n$ points in a real Hilbert space $\mathcal{H}$ so that $||x_i|| \leq r$ for all i and $\mathcal{G} := \{g : \mathcal{H} \to \mathbb{R} \mid g(z) = \langle z, w \rangle, \ ||w||^{-1} \geq \rho\}$. With $G_{ij} := \langle x_i, x_j \rangle$ the empirical Rademacher complexity of $\mathcal{G}$ w.r.t. $\{x_1, \ldots x_n\}$ satisfies

$$\hat{\mathcal{R}}(\mathcal{G}) \leq \frac{\operatorname{tr}[G]^{1/2}}{n\rho} \leq \frac{r}{\rho\sqrt{n}}. \tag{3.25}$$

Proof. The first inequality follows from

$$\begin{aligned} \hat{\mathcal{R}}(\mathcal{G}) &= \frac{1}{n} \mathbb{E}_\sigma \left[\sup_{||w|| \leq 1/\rho} \langle \sum_{i=1}^{n} \sigma_i x_i, w \rangle \right] \\ &\leq \frac{1}{n\rho} \mathbb{E}_\sigma \left\| \sum_{i=1}^{n} \sigma_i x_i \right\| \ \leq \ \frac{1}{n\rho} \left[\mathbb{E}_\sigma \left\| \sum_{i=1}^{n} \sigma_i x_i \right\|^2 \right]^{1/2} \\ &= \frac{1}{n\rho} \left[\mathbb{E}_\sigma \sum_{i,j=1}^{n} \sigma_i \sigma_j \langle x_i, x_j \rangle \right]^{1/2} \ = \ \frac{1}{n\rho} \left[\sum_{i=1}^{n} \langle x_i, x_i \rangle \right]^{1/2}. \end{aligned}$$

Here the first inequality is implied by Cauchy-Schwarz and the second by Jensen's inequality (applied to the concave square root function). The last step in the chain follows from the fact that if $i \neq j$, then $\mathbb{E}_\sigma[\sigma_i \sigma_j] = \mathbb{E}_\sigma[\sigma_i]\ \mathbb{E}_\sigma[\sigma_j] = 0$ since the Rademacher variables are independent and uniform.

The second inequality in Eq.(3.25) uses in addition that $\sum_{i=1}^n \langle x_i, x_i \rangle \leq nr^2$. □

Bibliography

[1] Amiran Ambroladze, Emilio Parrado-Hernández, and John Shawe-Taylor. Complexity of pattern classes and the Lipschitz property. *Theor. Comput. Sci.*, 382(3):232–246, 2007.

[2] Peter L Bartlett, Vitaly Maiorov, and Ron Meir. Almost Linear VC-Dimension Bounds for Piecewise Polynomial Networks. *Neural Comput.*, 10(8):2159–2173, 1998.

[3] P.L. Bartlett, P.M. Long, and R.C. Williamson. Fat-shattering and the learnability of real-valued functions. *J. Comput. Syst. Sci.*, 52(3):434–452, 1996.

[4] Eric B. Baum and David Haussler. What Size Net Gives Valid Generalization? *Neural Comput.*, 1(1):151–160, 1989.

[5] Shai Ben-David, N. Cesa-Bianchi, D. Haussler, and P. Long. Characterization of learnability for classes of n-valued functions. *J. Comput. Syst. Sci.*, 50:74–86, 1995.

[6] Shai Ben-David and Michael Lindenbaum. Lcalization vs. Identification of Semi-Algebraic Sets. *Mach. Learn.*, 32:207–224, 1998.

[7] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, 36(4):929–965, 1989.

[8] Ralph P. Boas and Harold P. Boas. *A Primer of Real Functions.* Cambridge University Press, 1996.

[9] Amit Daniely, Sivan Sabato, Shai Ben-David, and Shai Shalev-Shwartz. Multiclass Learnability and the ERM principle.

[10] Christian J.J. Despres. The Vapnik-Chervonenkis dimension of norm on Rd. 2014.

[11] R. M. Dudley. Balls in Rk do not cut all subsets of k + 2 points. *Adv. Math. (N. Y).*, 31(3):306–308, 1979.

[12] Paul W Goldberg and Mark R Jerrum. Bounding the Vapnik-Chervonenkis dimension of concept classes parameterized by real numbers. *Mach. Learn.*, 18(2):131–148, 1995.

[13] Nick Harvey, Chris Liaw, and Abbas Mehrabian. Nearly-tight VC-dimension bounds for piecewise linear neural networks.

[14] W. Hoeffding. Probability inequalities for sums of bounded random variables. *J. Am. Stat. Assoc.*, 58(301):13–30, 1963.

[15] Jean Jacod and Philip Protter. *Probability Essentials.* Springer Berlin / Heidelberg, 2 edition, 2004.

[16] Marek Karpinski and Angus Macintyre. Polynomial Bounds for VC Dimension of Sigmoidal and General Pfaffian Neural Networks. *J. Comput. Syst. Sci.*, 54(1):169–176, 1997.

[17] Pascal Koiran and Eduardo D Sontag. Neural Networks with Quadratic VC Dimension. *J. Comput. Syst. Sci.*, 54(1):190–198, 1997.

[18] W G Maass. Neural networks with superlinear {VC} dimension. *Neural Comput.*, 6:877–884, 1994.

[19] Colin McDiarmid. On the method of bounded differences, 1989.

[20] Allan Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numer.*, 8:143, 1999.

[21] A. Sakurai. Tight bounds for the VC-dimension of piecewise polynomial networks. *Adv. Neural Inf. Process. Syst.*, 11:323–329, 1998.

[22] Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Learnability, Stability and Uniform Convergence. *J. Mach. Learn. Res.*, 11:2635–2670, 2010.