Mathematical Machine Learning

Yahya Saleh, Tizian Wenzel, Kamal Sharma

March 8, 2024

Contents

Introduction

Underlying the success of artificial intelligence are learning algorithms, i.e., algorithms that learn from data to perform a certain task. We start this book by two concrete examples of supervised learning algorithms. In the first example we consider the problem of approximating functions from pointwise evaluations using linear regression. In the second example we look at the task of classifying hand-written digits. In these two examples we identify and familiarize ourselves with the main components of learning algorithms; *datasets*, *a hypothesis class*, and *optimization algorithms*. We further identify important aspects of supervised learning algorithms, such as overfitting, and underfitting. Finally, we motivate in these two examples problems at the forefront of research in mathematical machine learning, namely *the curse of dimensionality (CoD)* and double/multiple descent phenomenon.

I.1 Examples of Supervised Learning

In supervised learning tasks the dataset D is made up of two components, input variables $D_x = \{x_i\}_{i=1}^N$ and targets $D_y = \{y_i\}_{i=1}^N$. The dataset is assumed to be generated by an unknown function f : input → target. The goal in a supervised learning task is to approximate the unknown function f pointwise, i.e., to find a function h such that $h(x) \approx f(x)$ for any x, whether it belongs to D_x or not. The target value can take finitely many values, e.g., target $\in \{0, 1, \ldots, M\}$. In such a case, the supervised learning task is called a *classification task*. If the target can take infinitely many values, the learning task is called a regression task.

Supervised learning problems are approached by first choosing a *hypothesis space* $\mathfrak{H}$, in which one looks for an approximation to the unknown function f. For example, if the data x is one-dimensional and the target take values in $\mathbb{R}$ one can define the hypothesis class to be the set of all affine mappings, i.e.,

$$\mathfrak{H} = \big\{ f \mid f(x) = ax + b,\ \ a, b, \in \mathbb{R} \big\}. \tag{I.1}$$

Then, one define a loss function l that measures how well a hypothesis function h approximates an unknown function f at a point x. Using the dataset D, the supervised learning problem can be then formulated as an optimization problem

$$\min_{h \in \mathfrak{H}} \frac{1}{N} \sum_{i=1}^{N} l(h(x_i), y_i). \tag{I.2}$$

While there are many alternatives to solve these optimization problems, the by-far most used algorithms are variants of the gradient-descent algorithm.

Let's look at some concrete examples.

Example I.1. [Regression] ☛ I.1 Let x a variable that takes values in the interval $[-1, 1]$. And assume we have access to a dataset $D = \{(x_i, y_i)_{i=1}^{100}\}$ generated by the unknown function

$$f(x) = \cos(5x)\exp(-x).$$

To learn a function h that approximates f let your hypothesis class be the class of affine functions (I.1). Let the loss function be the absolute error, i.e.,

$$\begin{aligned} l(h(x_i), y_i) &= |h(x_i) - y_i| \\ &= |ax_i + b - y_i|. \end{aligned}$$

Use a gradient-descent-like algorithm to choose the best hypothesis h, i.e., the best scalars a and b.

Change your hypothesis class to the class of all polynomials up to the tenth order and repeat the optimization process. Which class is better for optimization?

Example I.2. [Classification] ☛ I.2

Figure (??) shows two hypotheses that we learned to fit the data in I.1. The figure depicts an interesting phenomenon; a certain hypothesis h can reproduce the training data accurately but still fail to generalize well to unseen data. This phenomenon is termed *overfitting* In fact, a hypothesis that fits the training data less well might generalize better to unseen examples.

The example also highlights the difference between fitting and interpolating.

Solutions to overfitting:

Increasing the size of the dataset.

Reducing the complexity of the hypothesis space, regulariazation, linking to chapter 2, and to Occam Razor's principle.

A formal definition of learning.

I.2 Curse of Dimensionality

Programming exercise and theorem about approximating smooth functions in higher dimensions.

Example I.3. [Classification] ☛ I.3 tba.

I.3 Approximating Highly-Oscillatory Functions

Example I.4. [Classification] ☛ I.4 tba.

I.4 Validity of Ocaam Razor and Double-Descent Phenomenon

Example I.5. [Double descent] ☛ I.5 tba.

Wait! What is what?

Here is a list of questions that help you check your understanding of key concepts inside this chapter?

Empirical Risk Minimization Principles

Machine Learning Models

Modern Machine Learning

Modern Mathematical Machine Learning