Mathematics of Machine Learning

Martin Lotz

WARWICK
THE UNIVERSITY OF WARWICK

Practical Information

All the lecture notes, exercises, and additional material can be found under

http://homepages.warwick.ac.uk/staff/Martin.Lotz/courses/learning/

and on Moodle.

Online lecture: Thursday 15-16 on MS Teams + **2 pre-recorded lectures / week**

Plenty of material is made available:

- **Lecture notes (more detailed, textbook style), lecture slides/write-up, and video recordings of all lectures (on Echo360 and YouTube)**

Lecture notes contain more information than what is strictly necessary for exam, but it is essential for understanding the material that you read them. You should make use of all the material provided!

Tutorials / Supervision class / Assignments

Assignments are worth 15% of mark

Assignment consists of handing in the solution to one problem worth 3 points each week (there will be 6 assignments, total marks capped at 15 points)

- **You can choose from 2-3 problems per sheet**
- **Problems released on Monday of week n need to be handed by Monday of week n+1 at 12:00**
- **Solutions will be published towards end of module**

Solutions and background discussed in tutorials

Tutorials: Tuesday 13-14 on MS Teams (in separate channel)

Tutors: Haoran Ni and Yijie Zhou (say hi!)

Prerequisites

Main prerequisites:

- **Core modules from the first two years (Mathematics) at Warwick (Linear Algebra, Analysis) or eqivalent**
- **Familiarity with Probability Theory (random variables, independence, conditional probability and expectation, normal distribution)**
- **Willingness to work independently and learn new mathematical concepts as needed**
- **A summary and refresher on probability is available on the module page and on Moodle**

Computing

The module is about mathematical foundations and no prior programming experience is required.

However, I will occasionally use examples based on Python code, and you will profit from learning Python!

Some recommended resources:

- **Anaconda Python:** https://www.anaconda.com/products/individual
- **Scikit Learn:** https://scikit-learn.org/stable/
- **Pytorch:** https://pytorch.org/

There are many tutorials and online sources available!

What is Machine Learning?

Learning is the process of turning experience into knowledge.

Knowledge is measured by the ability to perform certain tasks.

Machine Learning studies algorithms and models for computer systems to carry out certain tasks independently, based on the results of a learning process.

- Classification
- Pattern recognition
- Ranking
- Clustering
- Regression
- Decision support

What is Machine Learning?

- What can be learned?
- How do we assess the results of a learning process?
- How efficiently can we learn a task?

We will study some of these questions using tools from approximation theory, statistics, and optimization theory.

The Formal Setting

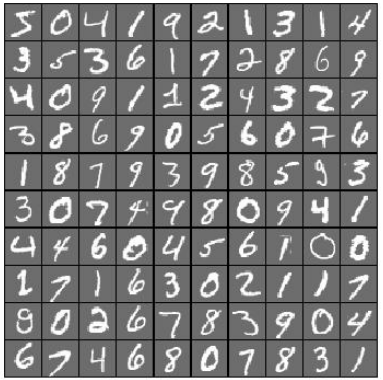
In Machine Learning, the goal is to come up with (or learn) a function

$$h \colon \mathcal{X} \to \mathcal{Y}$$

$\mathcal{X}$ **: input space (or space of features)**
$\mathcal{Y}$ **: output space (or space of labels or responses)**
h **: classifier or predictor**

Example: handwritten digit recognition

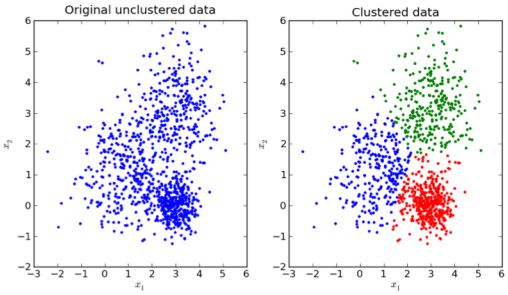
$\mathcal{X}$**: images,** $\mathcal{Y} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$



Types of Learning

In Unsupervised Learning we do not have any information about the data.

Example: clustering



In Supervised Learning we have a set of input-output pairs

$$(x_i, y_i) \in \mathcal{X} \times \mathcal{Y},\ 1 \leq i \leq m$$

from which we would like to infer the function h

Example: natural language processing, handwriting recognition

Approximation Theory Perspective

Assumption: there is an underlying ``true'' function

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

that generated the seen data.

Goal: approximate this function with a function h from a class $\mathcal{H}$

Examples: indicator functions, linear functions (linear regression), polynomials, neural networks

Linear Regression

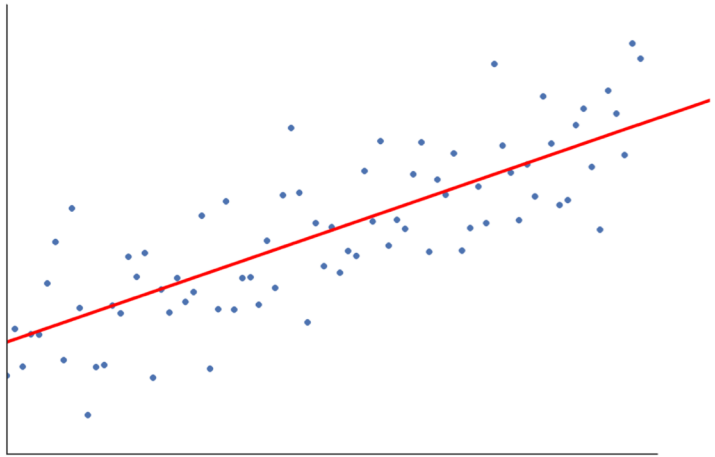
Given

$$(x_i, y_i) \in \mathbb{R}^n \times \mathbb{R},\ 1 \leq i \leq m$$

Find a linear rule

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_n X_n$$

that approximates the data best in some sense

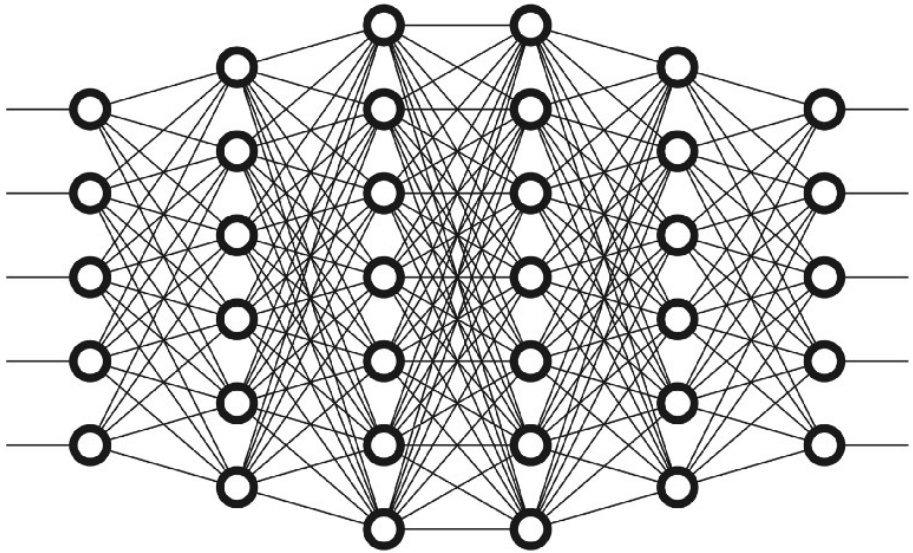


Neural Networks

A Neural Network is a composite function of the form

$$f(x) = \alpha_L \circ f_L \circ \cdots \circ \alpha_1 \circ f_1$$

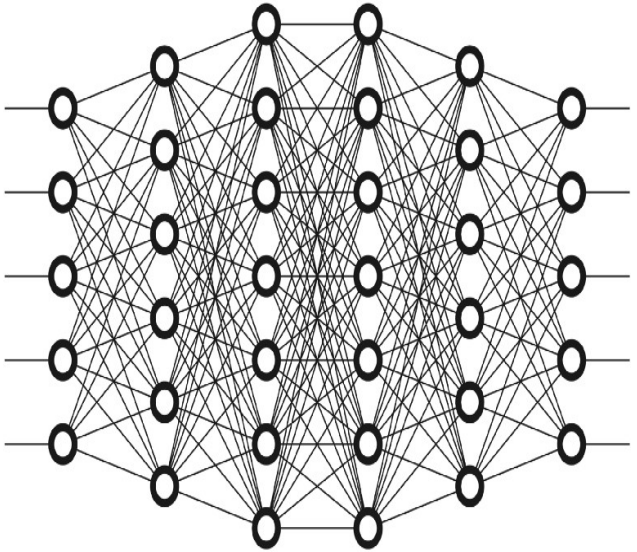
f_i : **linear,** α_i : **activation function**



Neural Networks and Mathematics

Applications of Neural Networks

- Natural language processing (NLP)
- Medical diagnostics
- Self-driving cars
- Anomaly detection
- Voice recognition
- Financial forecasting
- Medical imaging
- Drug discovery
- Manufacturing
- Generating realistic data (`deep fakes')
- Generating music
- Playing games
- …and Mathematics! → IMO Grand Challenge



Can every function be approximated?

Theorem (Weierstrass) Let f be a continuous function on the interval $[a, b]$. Then for every $\epsilon > 0$ there exists a polynomial p such that

$$\|f - p\|_\infty = \max_{x \in [a,b]} |f(x) - p(x)| \leq \epsilon$$

Problems:

- **No control of the size of the approximating polynomial**
- **No efficient procedure for finding such a polynomial**

Universal Approximation Theorem

Theorem (Cybenko) Let f be a continuous function on the interval $[a, b]$. Then for every $\epsilon > 0$ there exists a neural network p such that

$$\|f - p\|_\infty = \max_{x \in [a,b]} |f(x) - p(x)| \leq \epsilon$$

- **Neural networks can be found through training**
- **Training can be carried out (relatively) efficiently using backpropagation**

Optimization Perspective

How do we assess the quality of an approximation?

Loss function:

$$L\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$$

measures mismatch of prediction and actual value.

Given training data $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y},\ 1 \leq i \leq m$, find function h from a class of functions with minimal loss:

$$\underset{h \in \mathcal{H}}{\text{minimize}}\ \ \hat{R}(h) := \frac{1}{m} \sum_{i=1}^{m} L(h(x_i), y_i).$$

The function $\hat{R}(h)$ is the empirical risk or empirical error.

Optimization Perspective

Examples of loss functions

Quadratic loss: $L(h(x), y) = (y - h(x))^2, \quad y \in \mathbb{R}$ **(Regression)**

Binary loss:
$$\mathbf{1}\{h(x) \neq y\} = \begin{cases} 1 & h(x) = y \\ 0 & h(x) \neq 0 \end{cases}$$ **(Classification)**

Logistic loss: $L(h(x), y) = \log\left(1 + e^{-h(x)y}\right).$

Statistical Point of View

Notation

Let $(\mathcal{X}, \Omega, \mathbb{P})$ be a probability space.

- **Probability of event $A \in \Omega$: $\mathbb{P}\{A\}$**
- **Random variable: measurable $X \colon \mathcal{X} \to \mathbb{R}^n, \quad \mathbb{P}\{X \in A\} := \mathbb{P}\{X^{-1}(A)\}$**
- **The random variable is discrete, if only countable many values. Then statements such as $\mathbb{P}\{X = k\}$ make sense.**
- **Density of absolutely continuous random variable: $\mathbb{P}\{X \in A\} = \int_A f(x)\ \mathrm{d}x$**
- **Expected value / mean: $\mathbb{E}[X] = \int_{\mathbb{R}^n} x f(x)\ \mathrm{d}x$ or $\mathbb{E}[X] = \sum_k k \mathbb{P}\{X = k\}$**

Review of Conditional Expectation

Conditional probability / Bayes (X, Y discrete)

$$\mathbb{P}(X=x \mid Y=y) = \frac{\mathbb{P}(X=x, Y=y)}{\mathbb{P}(Y=y)} = \frac{\mathbb{P}(Y=y \mid X=x) \cdot \mathbb{P}(X=x)}{\mathbb{P}(Y=y)}$$

Example Rolling dice, X: result, $Y = \begin{cases} 0 & \text{even} \\ 1 & \text{odd} \end{cases}$

$$\mathbb{P}(X=3 \mid Y=1) = \frac{\mathbb{P}(X=3, Y=1)}{\mathbb{P}(Y=1)} = \frac{1}{3}$$

$\mathbb{P}(X=3, Y=1) = \frac{1}{6}$, $\mathbb{P}(Y=1) = \frac{1}{2}$

Continuous random var. density $g(x \mid y) = \frac{g(x,y)}{g(y)}$

Review of Conditional Expectation

Conditional Expectation $\mathbb{E}[X|Y=y] = \sum_x x \cdot \mathbb{P}(X=x|Y=y)$

or $\int_x x\, g(x|y)\, dx$

$\mathbb{E}[X|Y]$ is random variable with values $\mathbb{E}[X|Y=y]$

Iterated expectation $\mathbb{E}[\mathbb{E}[X|Y]] = \mathbb{E}[X]$

[Reason: $\mathbb{E}[X] = \sum_x \mathbb{P}(X=x)\cdot x = \sum_{x,y} \mathbb{P}(X=x|Y=y)\mathbb{P}(Y=y)\cdot x$

$= \sum_y \mathbb{E}[X|Y=y]\cdot \mathbb{P}(Y=y) = \mathbb{E}[\mathbb{E}[X|Y]]$]

Statistical Point of View

Generalization Error:

$$R(h) = \mathbb{E}[L(h(X), Y)]$$

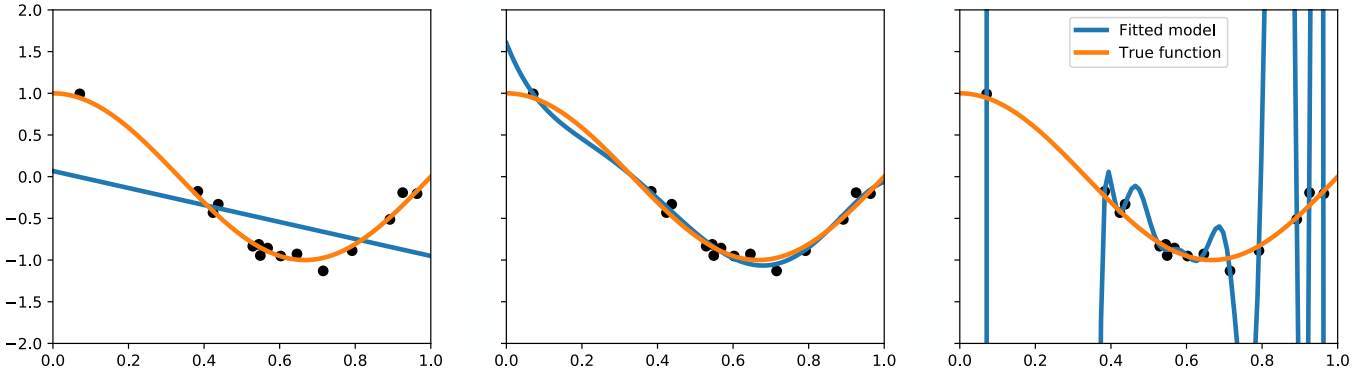
Given data $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y},\ 1 \leq i \leq m$, we can think of the empirical risk

$$\hat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} L(h(x_i), y_i).$$

as an estimate.

Learning vs Fitting

Important: the best classifier / hypothesis might not be the one that best fits the observed data!



Empirical Risk Minimization

Given a class of function (hypothesis space) $\mathcal{H}$ **:**

$$\hat{h} = \text{argmin}_{h \in \mathcal{H}} \hat{R}(h)$$

where

$$\hat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} L(h(x_i), y_i).$$

is the empirical risk and $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y},\ 1 \leq i \leq m$

This is what we can solve in practice!

Stochastic Optimization Methods

$$\min_x f(x) \quad , \quad f(x) = \sum_{i=1}^{n} f_i(x) \, , \; n \text{ large}$$

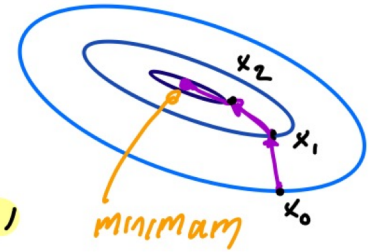
Gradient descent Begin with $x_0 \in \mathbb{R}^d$,

for $i \geq 0$: $x_{i+1} = x_i - \alpha_i \cdot \nabla f(x_i)$

hard to compute!

[Hope: $x_i \xrightarrow{i \to \infty}$ minimizer of f]

Stochastic gradient descent:



Select a few of the f_i at random, compute these gradients.

Standard Method for training neural networks!

Overview

This course will consist of roughly three parts:

1. **Statistical Learning Theory**

2. **Optimization**
   - **Stochastic Gradient Descent and related methods**
   - **Support Vector Machines**

3. **Foundations of Deep Learning (theory and limitations), including:**
   - **Universal approximation**
   - **Convolutional networks**
   - **Generative models**
   - **Sequence models**
   - **Reinforcement learning**

That’s all for today !