Introduction to Machine Learning

Introduction: Learners

compstat-lmu.github.io/lecture_i2ml

COMPONENTS OF A LEARNER

Summarizing what we have seen before, nearly all supervised learning algorithms can be described in terms of three components:

Learning = Hypothesis Space + Risk + Optimization

- **Hypothesis Space:** Defines (and restricts!) what kind of model f can be learned from the data.
- **Risk:** Quantifies how well a specific model performs on a given data set. This defines how to compare observed values to predictions and allows us to rank candidate models in order to choose the best one.
- **Optimization:** Defines how to search for the best model in the **hypothesis space**, typically guided by the metric used for the **risk**.

SUPERVISED LEARNING, FORMALIZED

A **learner** (or **inducer**) $\mathcal{I}$ is a *program* or *algorithm* which

- receives a **training set** $\mathcal{D} \in \mathcal{X} \times \mathcal{Y}$
- and uses an **optimization** procedure to find

$$\hat{f} = \arg\min_{f \in \mathcal{H}} \mathcal{R}_{\text{emp}}(f)$$

- for a given **hypothesis class** $\mathcal{H}$ of **models** $f : \mathcal{X} \to \mathbb{R}^g$,
- based on a **risk** function $\mathcal{R}_{\text{emp}}(f)$ that quantifies the performance of $f \in \mathcal{H}$ on $\mathcal{D}$.

As before, we can also adapt this concept to finding $\hat{\boldsymbol{\theta}}$ for parametric models.

(This does not cover all special cases, but it's a useful framework for most supervised ML problems.)

LEARNING AS EMPIRICAL RISK MINIMIZATION

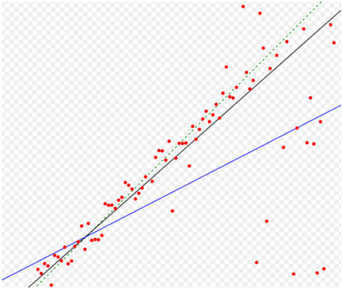
- By decomposing learners into these building blocks,
  - we have a framework to understand how they work,
  - we can more easily evaluate in which settings they may be more or less suitable, and
  - we can tailor learners to specific problems by clever choice of each of the three components.
- There will, for instance, be optimization procedures that work well for a certain combination of hypothesis space and risk function but perform poorly on others.
- In fact, it is a commonly acknowledged problem that no universally best learner exists.

EXAMPLE OF A LEARNER

So what could a learner look like? Let us consider a linear regression task with a single feature and a single target variable.

- The **hypothesis space** in univariate linear regression is the set of all linear functions, with $\boldsymbol{\theta} = (\theta_0, \theta)$:

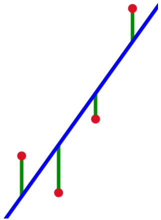
$$\mathcal{H} = \{f(\mathbf{x}) = \theta_0 + \theta\mathbf{x} : \theta_0, \theta \in \mathbb{R}\}$$



EXAMPLE OF A LEARNER

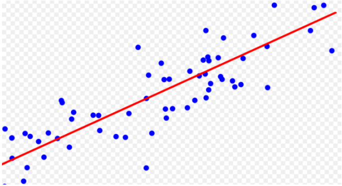
- We might use the mean squared error as our loss function, punishing larger distances between observations and regression line more severely:

$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - \theta_0 - \theta \mathbf{x}^{(i)})^2$$



- **Optimization** will usually mean deriving the ordinary-least-squares (OLS) estimator $\hat{\theta}$ analytically. We might, however, also use gradient descent or some other optimization procedure.

EXAMPLE OF A LEARNER



VARIETY OF LEARNING COMPONENTS

Hypothesis Space : $\begin{cases} \text{Step functions} \\ \text{Linear functions} \\ \text{Sets of rules} \\ \text{Neural networks} \\ \text{Voronoi tesselations} \\ \text{...} \end{cases}$

Risk : $\begin{cases} \text{Mean squared error} \\ \text{Misclassification rate} \\ \text{Negative log-likelihood} \\ \text{Information gain} \\ \text{...} \end{cases}$

Optimization : $\begin{cases} \text{Analytical solution} \\ \text{Gradient descent} \\ \text{Combinatorial optimization} \\ \text{Genetic algorithms} \\ \text{...} \end{cases}$