Chapter 1

A formal learning model

1.1 Probably Approximated Correct (PAC) learning

Definition 1.1.1 (PAC learnability)**.** A hypothesis class $\mathcal{H}$ is PAC learnable if there exist a function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm with the following property: $\forall \varepsilon, \delta \in (0,1)$, for every distribution $\mathcal{D}$ over $\mathcal{X}$, and for every labeling function $f : \mathcal{X} \to \{0,1\}$, if the realizable assumption holds with respect to $\mathcal{H}, \mathcal{D}, f$, then when running the learning algorithm on $m \geq m_{\mathcal{H}}(\varepsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$ and labeled by f, the algorithm returns a hypotesis h such that, with probability of at least $1 - \delta$ (over the choice of the examples), $L_{(\mathcal{D},f)}(h) \leq \varepsilon$.

The function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ determines the sample complexity of learning $\mathcal{H}$. So it determines how many samples are required to guarantee a probably approximately correct solution.

Corollary 1.1.0.1. *Every finite class is PAC learnable with sample complexity:*

$$m_{\mathcal{H}}(\varepsilon, \delta) \leq \left\lceil \frac{1}{\varepsilon} \log\left(\frac{|\mathcal{H}|}{\delta}\right) \right\rceil \tag{1.1}$$

Now we want to generalize by:

- Removing the realizability assumption.
  For practical learning tasks, this assumption may be too strong and it isn't necessarily met.

- Extending to learning problems beyond the simple binary classification.
  One may wish to predict real valued numbers or a label picked from a finite set of labels. Therefore our analysis can be extended to many othere scenarios by allowing a variety of loss functions.

1.2 Agnostic PAC learning

We introduce a more realistic model for the data-generating distribution. From now on, let $\mathcal{D}$ be a probability distribution over $\mathcal{X} \times \mathcal{Y}$ (the meaning of the notation hasn't changed). $\mathcal{D}$ is a **joint distribution** over domain points and labels and it can be viewed as the composition of two parts:

- A distribution $\mathcal{D}_x$ over unlabeled domain points (also called **marginal distribution**).

- A **conditional probability** $\mathcal{D}((x,y)|x)$ over labels for each domain point.

1.2.1 Empirical and true error revised

One can measure how likely h is to make an error when labeled points are randomly drawn according to $\mathcal{D}$ previously introduced. So we redefine the true error of a prediction rule h:

$$L_{\mathcal{D}}(h) := \underset{(x,y)\sim\mathcal{D}}{\mathbb{P}}[h(x) \neq y] := \mathcal{D}(\{(x,y) \ : \ h(x) \neq y\}) \tag{1.2}$$

We would like to find a predictor h for which the error in Eq. 1.2 will be minimized. However, the learner doesn't know the data generating $\mathcal{D}$, but he has access only to the training set S. Hence the definition of the empirical risk remains the same as before:

$$L_S(h) := \frac{|\{i \in [m] \ : \ h(x_i) \neq y_i\}|}{m} \tag{1.3}$$

The goal is to find some hypotesis $h \ : \ \mathcal{X} \to \mathcal{Y}$ that (probably approximately) minimizes the true risk, $L_{\mathcal{D}}(h)$.

1.2.2 Bayes optimal predictor

Given any probability distribution $\mathcal{D}$ over $\mathcal{X} \times \{0,1\}$, the best label predicting function from $\mathcal{X}$ to $\{0,1\}$ will be:

$$f_{\mathcal{D}}(x) = \begin{cases} 1 & \text{if } \mathbb{P}[y=1|x] \geq \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \tag{1.4}$$

Proposition 1.2.1. *For every probability distribution $\mathcal{D}$, the Bayes optimal predictor $f_{\mathcal{D}}$ is optimal, in the sense that no other classifier $g \ : \ \mathcal{X} \to \{0,1\}$ has a lower error. Mathematically:*

$$L_{\mathcal{D}}(f_{\mathcal{D}}) \leq L_{\mathcal{D}}(g) \qquad \forall g \tag{1.5}$$

Clearly, we can't hope that the learning algorithm will find a hypotesis whose error is smaller than the minimal possible error of the Bayes predictor. We require that the learning algorithm will find a predictor whose error is not much larger than the best possible error of a predictor in some given benchmark hypotesis class. So, we generalize the definition of PAC learning.

Definition 1.2.1 (Agnostic PAC learnability)**.** A hypotesis class is PAC learnable if there exist a function $m_{\mathcal{H}} \ : \ (0,1)^2 \to \mathbb{N}$ and a learning algorithm with the following property: for every $\varepsilon, \delta \in (0,1)$ and for every distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, when running the learning algorithm on $m \geq m_{\mathcal{H}}(\varepsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$, the algorithm returns a hypotesis h such that, with probability of at least $1-\delta$ (over the choice of the m training examples):

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \varepsilon \tag{1.6}$$

In the agnostic case, the learner is required to achieve a small error relative to the best error achievable by the hypotesis class and not in absolute terms. In other words, we drop the requirement of finding the best predictor, but we don't want to be too far from it.

1.2.3 Scope of the learning problems

We want to extend our model so that it can be applied to a wide variety of learning tasks, such as:

- **Multiclass Classification**:
  Our classification doesn't have to be binary. The label set can be much more complex, for example it can be a set of real numbers.

- **Regression**:11 In this task, one wishes to find some simple pattern in the data, i.e. a functional relationship between the $\mathcal{X}$ and $\mathcal{Y}$ components of the data. For example, one wishes to find a linear function that describes the correlation between a sample of $\mathcal{X}$ and the corresponding value inside $\mathcal{Y}$. However, for this type of tasks, the measure of success has to be changed. We may evaluate the quality of a hypotesis function, $h : \mathcal{X} \to \mathcal{Y}$, by the *expected square difference* between the true labels and their predicted values, namely:

$$L_{\mathcal{D}}(h) := \underset{(x,y)\sim\mathcal{D}}{\mathbb{E}}[h(x) - y]^2 \tag{1.7}$$

To extend the reach of machine learning techniques we must generalize our formalism of the measure of success.

1.2.4 Generalized loss functions and common examples

Given any set $\mathcal{H}$ (our hypotheses or models) and some domain Z, let ℓ be any function from $\mathcal{H} \times Z$ to the set of non-negative real numbers, ℓ, namely $\mathcal{H} \times z \to \mathbb{R}_+$. We call such functions **loss functions**.
We now define the **risk function** to be the expected loss of a classifier $h \in \mathcal{H}$ with respect to a probability distribution $\mathcal{D}$ over Z, namely:

$$L_{\mathcal{D}}(h) := \underset{z\sim\mathcal{D}}{\mathbb{E}}[\ell(h, z)] \tag{1.8}$$

In other words, we consider the expectation of the loss of h over objects z picked randomly according to $\mathcal{D}$. Similarly, we define the **empirical risk** to be the expected loss over a given sample $S = (z_1, \ldots, z_m) \in Z^m$, namely:

$$L_S(h) := \frac{1}{m}\sum_{i=1}^{m} l(h, z_i) \tag{1.9}$$

Some of the most common examples of losses are given below:

- **0-1 loss**:
  Our random variable z ranges over the set of pairs $\mathcal{X} \times \mathcal{Y}$ and the loss function is:

$$\ell_{0-1}(h, (x, y)) := \begin{cases} 0 & \text{if } h(x) = y \\ 1 & \text{if } h(x) \neq y \end{cases} \tag{1.10}$$

  It is used in binary and multiclass classification.

- **Square loss (L2)**:
  Our loss function is:

$$\ell_{\text{sq}}(h, (x, y)) := [h(x) - y]^2 \tag{1.11}$$

  This loss is used in regression problems to penalize few large errors.

- **Absolute value loss (L1)**:
  Just like before, but with modulus and not the square:

$$\ell_{\text{abs}}(h,(x,y)) := |h(x) - y| \tag{1.12}$$

  It is used in regression tasks to penalize many small errors.

To summarize, we formally define agnostic PAC learnability for general loss functions.

Definition 1.2.2 (Agnostic PAC learnability for general loss functions)**.** A hypotesis class $\mathcal{H}$ is PAC learnable with respect to a set Z and a loss function $\ell : \mathcal{H} \times Z \to \mathbb{R}_+$, if there exist a function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm with the following property: for every $\varepsilon, \delta \in (0,1)$ and for every distribution $\mathcal{D}$ over Z, when running the learning algorithm on $m \geq m_{\mathcal{H}}(\varepsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$, the algorithm returns $h \in \mathcal{H}$ such that, with probability of at least $1 - \delta$ (over the choice of the m training examples):

$$l_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \varepsilon \tag{1.13}$$

where $L_{\mathcal{D}}(h) = \mathbb{E}_{z \sim \mathcal{D}}[\ell(h,z)]$

Remark 1. We can consider $\ell(h, \bullet) : Z \to \mathbb{R}_+$ a random variable. Hence $L_{\mathcal{D}}(h)$ is the expected value of this random variable.