Empirical Risk Minimization and PAC Learning

2

In this chapter, we discuss the theoretical foundations of machine learning, presenting the framework of empirical risk minimization (Sec. 2.1) in which to frame learning problems, the notion of inductive bias, and the main results of algorithmic learnability, encapsulated in the definition of PAC learning (Sec. 2.2) and of complexity of a set of hypothesis, namely VC-dimension and Rademacher complexity (Sec. 2.2.4 and 2.2.6). We will finish the chapter with an introduction to the information theoretical notion of Kullback-Leibler divergence (Sec. 2.3) and its link with maximum likelihood and empirical risk. The presentation of this chapter follows [6] and [7].

2.1 Empirical Risk Minimization

2.1.1 Notation and problem formalization

Consider the supervised learning scenario, in which there are an input and an output space. We will use the following notation:

- input space: $X \subseteq \mathbb{R}^n$ (real features or one-hot-encoding of categorical variables)
- output space: $Y = \mathbb{R}$, or $\{0, 1\}$, or $\{0, \dots, k\}$ depending on the problem at hand (i.e. regression, binary classification, multi-class classification).

We will work in a probabilistic framework. They key ingredient is the joint probability distribution between inputs and outputs $p(x, y) \in Dist(X \times Y)$ (with $x \in X, y \in Y$), called **data generating distribution**.

We assume that there is a functional relationship between inputs and outputs that we want to understand, given by a labelling function $f : X \to Y$. Re-writing the data generating distribution as $p(x, y) = p(x)p(y|x)$, it is sometimes the case that $p(y|x) = p(y|f(x))$ (e.g. in classification problems, or in regression problems in which an additive measurement error is assumed).

The input to our learning algorithm is a dataset D, where $|D| = N$, sampled from a probability distribution $D \sim p^N(x, y)$, i.e. it is a set of input-output pairs $D = \{(x_i, y_i)|i = 1, \dots, N\}$ such that $(x_i, y_i) \sim p(x, y)$. We assume these pairs to be independent (this is not true in general, however it is not a very restrictive assumption). Ideally we would like to recover f, starting from D.

A key point in Machine Learning is that, whenever we want to learn something, we need to make hypotheses on good candidate models for our task. So, since we are going to learn a function mapping inputs to

outputs (possibly phrased in probabilistic terms), we need to choose a set of functions that are likely to contain our true model, or at least to approximate it well.
This set is called **hypothesis class**, defined as $\mathcal{H} = \{h : X \to Y\}$. Typically these functions are chosen to be parametric, i.e. $h = h_\theta$ with $\theta \in \Theta \subset \mathbb{R}^k$ for some k.

Remark: restricting the set of models to consider is actually what enables us to learn (without any assumption, we cannot learn anything)!

$\mathcal{H}$ encodes our **inductive bias**, that is, the assumptions made to learn the target function and to generalize beyond training data. We stress once again that without inductive bias there is no learning.

2.1.2 Risk and empirical risk

Consider our set of hypotheses $\mathcal{H}$, a function $h \in \mathcal{H}$ and the joint probability distribution $p(x, y)$.

Definition 2.1.1 *The **loss function** $l(x, y, h) \in \mathbb{R}_{\geq 0}$, measures the error that we commit in using h to predict y from x. Intuitively, the higher its value, the worst our prediction.*

Examples:

- $0-1$ loss: $l(x, y, h) \equiv \mathbb{I}(h(x) \neq y)$, with $y \in \{0, 1\}$. That is, we have no error if we predict correctly the label of x, an error of 1 otherwise.
- squared loss $l(x, y, h) \equiv (h(x) - y)^2$, with $y \in \mathbb{R}$.

Remark: the loss function acts on a single input-output pair.

Definition 2.1.2 *The **risk** (or **generalization error**) is defined as*

$$R(h) = \mathbb{E}_{x,y\sim p(x,y)}[l(x, y, h)]$$

The risk is therefore a property of the hypothesis function h, i.e. each h comes with an associated risk. Our goal is to find a function h that minimizes the risk.

Remark: the risk depends on the true data distribution (which is unknown).

Definition 2.1.3 *The **empirical risk** (or **training error**) is defined as :*

$$\hat{R}(h) = \frac{1}{N} \sum_{i=1}^{N} l(x_i, y_i, h)$$

It is an empirical approximation, according to our sample, of the actual risk. This is what we can practically optimize.

Risk minimization principle: find $h^\star \in \mathcal{H}$ s.t. $h^\star = \text{argmin}_{h\in\mathcal{H}} R(h)$. That is, we need to find the hypothesis that minimizes the risk.

Definition 2.1.4 *If l is the $0-1$ loss and $\exists h \in \mathcal{H}$ s.t. $p(h(x) = f(x)) = 1$ (with $f(x)$ true class), then $\mathcal{H}$ has the **realizability property** and $R(h^\star) = 0$.*

Hypothesis sets with the realizability property contain the true model (in general this is not the case in practice).

Empirical risk minimization principle: find $h_D^\star = \text{argmin}_{h\in\mathcal{H}} \hat{R}(h)$. This minimum can be combinatorially hard to achieve, so sometimes we relax this framework and we only find a *good* solution. In what follows, we are going to address the problem of computing $R(h_D^\star)$, i.e. risk associated to the optimal solution.

2.1.3 Bias-Variance Trade-off

In this section, we want to analyze the generalization error and decompose it according to the sources of error that we are going to commit.

In what follows, we will use the squared loss (hence we will focus on regression problems). Considering $h \in \mathcal{H}$, an explicit expression of the generalization error committed when choosing hypothesis h is:

$$R(h) = \mathbb{E}_p[l(x, y, h)] = \iint (h(x) - y)^2 p(x, y) dx dy$$

Theorem 2.1.1 *The minimizer of the generalization error R is:*

$$g(x) = \mathbb{E}[y|x] = \int y p(y|x) dy$$

so that $g = \text{argmin}_h R(h)$, if $g \in \mathcal{H}$.

Proof. We can prove it by rewriting our risk as:

$$\begin{aligned} R(h) &= \iint (h(x) - y)^2 p(x, y) dx dy = \\ &= \iint (h(x) + g(x) - g(x) - y)^2 p(x, y) dx dy = \\ &= \iint [(h(x) - g(x))^2 + (g(x) - y)^2 + 2(h(x) - g(x))(g(x) - y)] p(x, y) dx dy \end{aligned}$$

Consider the term:

$$\begin{aligned} &\iint 2(h(x) - g(x))(g(x) - y) p(x, y) dx dy = \\ &= \int 2(h(x) - g(x)) p(x) \int (g(x) - y) p(y|x) dy dx = 0 \end{aligned}$$

since

$$\int (g(x) - y) p(y|x) dy = g(x) - \int y p(y|x) dy = 0$$

by definition of g. So that:

$$R(h) = \int (h(x) - g(x))^2 p(x)dx + \iint (g(x) - y)^2 p(x, y)dxdy$$

where the second term does not depend on h and expresses the idea of how noisy is our regression problem (this is something intrinsic to the problem).
The first term depends on h (actually it is the only thing that we can optimize when we minimize w.r.t. h), and it holds that:

$$\int (h(x) - g(x))^2 p(x)dx = 0 \leftrightarrow h(x) = g(x)$$

and so we proved that $g(x)$ is a minimizer. □

Remark: our goal is to to evaluate $R(h_D^\star)$, with $D \sim p^N(x, y)$.

By previous computations (evaluated at the solution of the empirical risk minimization problem), it holds that:

$$R(h_D^\star) = \int (h_D^\star(x) - g(x))^2 p(x)dx + noise$$

Consider now the average over all possible datasets:

$$\begin{aligned}\mathbb{E}_D[R(h_D^\star)] &= \int \mathbb{E}_D[(h_D^\star(x) - g(x))^2]p(x)dx = \\ &= \int \mathbb{E}_D[(h_D^\star(x) + \mathbb{E}_D[h_D^\star(x)] - \mathbb{E}_D[h_D^\star(x)] - g(x))^2]p(x)dx\end{aligned}$$

Carrying on the same computations as before and observing that:

$$\mathbb{E}_D[h_D^\star(x) - \mathbb{E}_D[h_D^\star(x)]] = 0$$

we get that the expected generalization error of our empirical risk minimizer is:

$$\begin{aligned}\mathbb{E}_D[R(h_D^\star)] &= \underbrace{\int (\mathbb{E}_D[h_D^\star(x) - g(x)])^2 p(x)dx}_{bias^2} \\ &+ \underbrace{\int \mathbb{E}_D[(h_D^\star(x) - \mathbb{E}_D[h_D^\star(x)])^2]p(x)dx}_{variance} \\ &+ \underbrace{\iint (g(x) - y)^2 p(x, y)dxdy}_{noise}\end{aligned}$$

The first term $\int (\mathbb{E}_D[h_D^\star(x) - g(x)])^2 p(x)dx$ captures the squared difference between the average predictor across all datasets (that we obtain from empirical risk minimization) and the optimal predictor. If this difference is small, on average across all datasets our empirical risk minimization is going to work well; if it is large, across all datasets our empirical risk minimization is going to work bad. We call this term

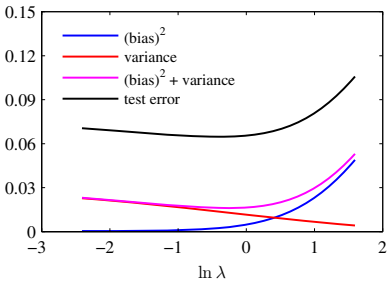

Figure 2.1: Bias-variance tradeoff. $\ln \lambda$ is a proxy for model complexity. Taken from Bishop.

squared bias because it essentially captures the distortion of our empirical risk minimization predictor, hence of our set of hypothesis $\mathcal{H}$.

The second term $\int \mathbb{E}_D[(h_D^\star(x) - \mathbb{E}_D[h_D^\star(x)])^2]p(x)dx$ encodes the variance, across all datasets, of our predictor, that is, how far each single instance of the empirical risk minimizer differs from the average predictor. We call this the *variance term*. The larger the variance, the more the intrinsic variability of the dataset is going to impact on what we can reconstruct.

The third term $\iint (g(x)-y)^2 p(x,y)dxdy$ is the noise (as observed before).

Remark: high variance essentially means that we are in a region of overfitting, high bias means that we are in a region of underfitting.

Since our goal is to minimize the empirical risk, we face a trade-off, called the **bias-variance trade-off**. On one hand, choosing $\mathcal{H}$ to be a very rich class might lead to overfitting (increasing the variance). On the other hand, choosing $\mathcal{H}$ to be a very small set might lead to underfitting (increasing the bias). A visual depiction of the trade-off is shown in Figure 2.1.

2.2 PAC Learning

2.2.1 Basic definitions

In what follows, our goal is to measure how much we can learn as a function of the model complexity. This results in the PAC (Probably Approximately Correct) learning framework, which encodes the notion of model complexity and gives also bounds on the error that we commit.

We are going to explore this framework in the context of (binary) classification, i.e. $y \in \{0,1\}$, using the $0-1$ loss.

Consider an hypothesis set $\mathcal{H}$ with the realizability property, i.e. $\exists \bar{h} \in H$ s.t. $p_{x,y}(\bar{h}(x) = y) = 1$, since $y \in \{0,1\}$ then $\exists f : X \to Y$ s.t. $p_{x,y}(\bar{h}(x) = f(x))$ (that is, our hypothesis set contains the true function).

Definition 2.2.1 *A realizable hypothesis set $\mathcal{H}$ is **PAC-learnable** iff $\forall \varepsilon, \delta \in$*

$(0,1), \forall p(x,y), \exists m_{\varepsilon,\delta} \in \mathbb{N}$ *s.t.* $\forall m \geq m_{\varepsilon,\delta}, \forall D \sim p^m, |D| = m$ *then*

$$p_D(R(h_D^\star) \leq \varepsilon) \geq 1 - \delta.$$

This means that, fixing two parameters $\varepsilon, \delta \in (0,1)$, governing our precision, and a data generating distribution $p(x,y)$, we can find a number $m_{\varepsilon,\delta}$ of observations (as a function of the parameters ε, δ), such that we are guaranteed to learn the true function with error bounded by ε (assuming that the true function is in $\mathcal{H}$) with high probability $(1-\delta)$, provided we have at least $m_{\varepsilon,\delta}$ data points in D. Note that the probability here is over the dataset D, meaning that our learning will succeed for a fraction $1-\delta$ of sampled datasets.

In the following, we consider a more general setting, in which we relax the realizability assumption, and furthermore we assume that we have at our disposal an algorithm A that takes D as input and returns a function h of $\mathcal{H}$ as output, ideally the minimizer of the empirical risk, but practically a good solution.

Definition 2.2.2 *Given an hypothesis set $\mathcal{H}$ (not necessarily realizable) and an algorithm A, $\mathcal{H}$ is **agnostic PAC-learnable** iff $\forall \varepsilon, \delta \in (0,1), \forall p(x,y)$, $\exists m_{\varepsilon,\delta} \in \mathbb{N}$ s.t. $\forall D \sim p^m$, $|D| = m \geq m_{\varepsilon,\delta}$*

$$p_D\left(R(h_D^A) \leq min_{h \in H} R(h) + \varepsilon\right) \geq 1 - \delta$$

being h_D^A the result of applying A to $\mathcal{H}$ and D.

In other words, there exists a number $m_{\varepsilon,\delta}$ of data points such that the algorithm learns a function having error close ($\leq \varepsilon$) to the minimum with high probability $(1-\delta)$.

In both definitions, we have a bound on the generalization error in terms of ε and δ and, in order for this bound to hold, we need to have enough data points. Typically:

- $m_{\varepsilon,\delta}$ depends polynomially on $\frac{1}{\varepsilon}, \frac{1}{\delta}$ (since we want the number of observations to increase moderately with the complexity of the problem);
- A should run in polynomial time.

2.2.2 Finite hypotheses sets

An hypothesis set is said to be **finite** if $\mathcal{H}$ is s.t. $|\mathcal{H}| < \infty$.
Using combinatorial arguments, we can prove that finite hypothesis sets are agnostic PAC-learnable with:

$$m_{\varepsilon,\delta} \leq \left\lceil \frac{2 \log\left(\frac{2|\mathcal{H}|}{\delta}\right)}{\varepsilon^2} \right\rceil$$

hence with polynomial dependency on ε and δ. In this framework, $\log(|\mathcal{H}|)$ is a measure of the complexity of the set $\mathcal{H}$.

Remark: if $\mathcal{H}$ is described by d parameters of type double when represented in a computer (64 bits), it holds that $|\mathcal{H}| \leq 2^{d \cdot 64}$, so we have a finite

set of hypothesis, hence we can provide a bound on every implementable set of hypothesis functions. In this case

$$m_{\varepsilon,\delta} \leq \frac{128d + 2\log(\frac{2}{\delta})}{\varepsilon^2},$$

i.e. we have linear dependency on the number of parameters.

2.2.3 Pac learning example

We consider an example introduced by Kearns and Vazirani in the book "An Introduction to Computational Learning Theory".

The goal is to learn a target axis-aligned rectangle R lying in $\mathbb{R}^2$ using a sample of m labeled training examples $(\mathbf{x}, z)$ with $\mathbf{x} \in \mathbb{R}^2$ and $z \in \{-1, 1\}$ distributed according to a distribution $p(\mathbf{x}, z)$. The hypothesis set $\mathcal{H}$ is the set of all axis-aligned rectangles lying in $\mathbb{R}^2$. We assume there is a true rectangle of this kind such that all positive points are inside it, and all negative points are outside.

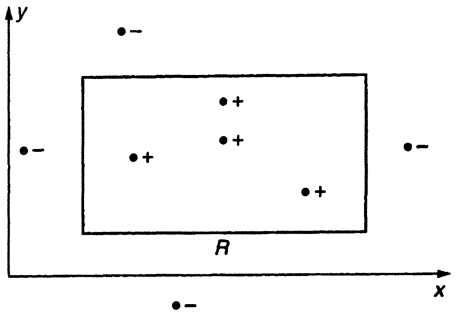

Figure 2.2: The target rectangle R with a sample of positive and negative examples

We consider our hypothesis h to be the axis-aligned rectangle R' with the smallest area that includes all of the positive examples and none of the negative ones.

What is the minimum number $m_{\varepsilon,\delta}$ of training examples so that, with probability at least $1 - \delta$, h has en error at most ε with respect to the true rectangle and the distribution $p(\mathbf{x}, z)$?

Solution

We have to find an $m_{\varepsilon,\delta}$ such that $\forall m > m_{\varepsilon,\delta}\ P(\text{error}(h_m) > \varepsilon) \leq \delta$. First of all, let's notice that the error error(h_m) is equal to the area of the target rectangle R minus the area of the internal rectangle $R' = h_m$ that we found.

Then, given an arbitrary error bound ε, we build within R 4 rectangles having each one an area of $\varepsilon/4$, on the sides of R (see Figure 2.3).

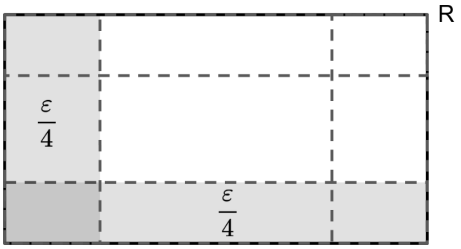

Figure 2.3: The 4 rectangles of area $\varepsilon/4$

Let's consider the m observations drawn from $p(\mathbf{x}, y)$. If each of the four rectangles defined above contains at least one point, we have $\text{error}(h_m) \leq \varepsilon$, because the area difference between R and R' would be fully covered by the 4 rectangles (see Figure 2.4).

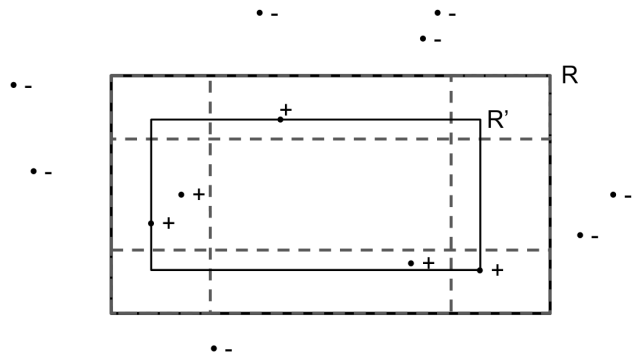

Figure 2.4: Configuration showing event B

If we call this event B, we have:

$$B \implies \text{error}(h_m) \leq \varepsilon$$

equivalently, by modus tollens:

$$\text{error}(h_m) > \varepsilon \implies \neg B$$

This implies:

$$P(\neg B) \geq P(\text{error}(h_m) > \varepsilon)$$

$P(\neg B)$ is the probability that at least one of the 4 rectangles doesn't contain any of the m points. This is probability is $(1 - \varepsilon/4)^m$ for a single rectangle, hence we have $P(\neg B) \leq 4(1 - \varepsilon/4)^m$. Thus the following chain of inequalities holds:

$$4(1 - \varepsilon/4)^m \geq P(\neg B) \geq P(\text{error}(h_m) > \varepsilon)$$

Now, let $m_{\varepsilon,\delta}$ be such that:

$$\delta \geq 4(1 - \varepsilon/4)^{m_{\varepsilon,\delta}} \geq P(\neg B) \geq P(\text{error}(h_m) > \varepsilon)$$

Finding $m_{\varepsilon,\delta}$ that satisfy this inequality would prove that $\forall m > m_{\varepsilon,\delta}\, P(\text{error}(h_m) > \varepsilon) \leq \delta$. We then require:

$$4(1 - \varepsilon/4)^{m_{\varepsilon,\delta}} \leq \delta$$

and we use the inequality $(1-k) \leq e^{-k}$ to obtain:

$$m_{\varepsilon,\delta} \geq (4/\varepsilon) \cdot ln(4/\delta)$$

In summary, provided a sample of at least $(4/\varepsilon) \cdot ln(4/\delta)$ examples in order to choose an hypothesis rectangle R', we can assert that with probability at least $1-\delta$, R' will misclassify a new point (drawn according to the same distribution from which the sample was chosen) with probability at most ε.

This proves that our hypothesis set $\mathcal{H}$ is PAC-learnable.

2.2.4 VC Dimension (Vapnik–Chervonenkis)

Consider a class of hypotheses functions $\mathcal{H} = \{h : X \to \{0,1\}\}$ and a subset $C = \{c_1, \ldots, c_m\} \subseteq X$ of input points.

Define $\mathcal{H}_C = \{(h(c_1), \ldots, h(c_m)) | h \in \mathcal{H}\}$, the set of all tuples of Booleans obtained by applying all possible hypothesis functions $h \in \mathcal{H}$ to all points in C. We say that $\mathcal{H}$ **shatters** the set C iff $|\mathcal{H}_C| = 2^m$.

Practically, this means that for any label assignment to points in C, we have a function in our hypothesis set which is able to match such an assignment. Namely, we can exactly describe every possible dataset with inputs in C.

Definition 2.2.3 *The **VC dimension of** $\mathcal{H}$ is defined as:*

$$VCdim(\mathcal{H}) = max\{m | \exists C \subseteq X, |C| = m \; s.t. \; \mathcal{H} \; shatters \; C\}$$

Remark: In calculating the VC dimension , it is enough that we *find one set of* m points that can be shattered, it is not necessary that we are able to shatter any m points.

Examples:

- $\mathcal{H}_a = \{h : \mathbb{R} \to \{0,1\} | h = \mathbb{1}_{\geq a}, a \in \mathbb{R}\}$ (threshold functions) has $VCdim(\mathcal{H}_a) = 1$
- $\mathcal{H}_{a+} = \{\mathbb{1}_{\geq a}, \mathbb{1}_{\leq a} | a \in \mathbb{R}\}$ has $VCdim(\mathcal{H}_{a+}) = 2$
- $\mathcal{H}_I = \{\mathbb{1}_{[a,b]} | a \leq b, a, b \in \mathbb{R}\}$ (intervals) has $VCdim(\mathcal{H}_I) = 2$
- $VCdim(\mathcal{H}_I \cup (1 - \mathcal{H}_I)) = 3$
- $\mathcal{H}_\ell = \{\mathbb{1}_{ax+by+c \geq 0}(x,y) | a, b, c \in \mathbb{R}\}$ (lines in $\mathbb{R}^2$) has $VCdim(\mathcal{H}_\ell) = 3$
- $\mathcal{H}_B = \{\mathbb{1}_B(x) \mid B \text{ is an axis-aligned box in } \mathbb{R}^2\}$ has $VCdim(\mathcal{H}_B) = 4$

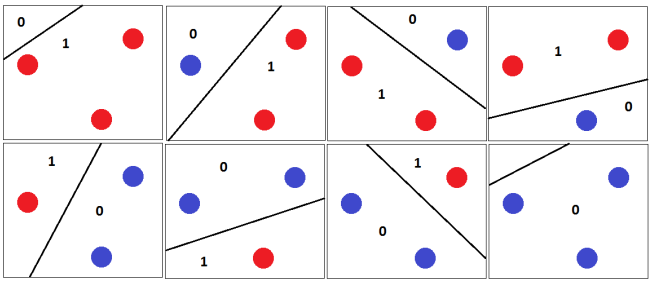

Figure 2.5: Proof that $VCdim(\mathcal{H}_\ell) \geq 3$

2.2.5 VC dimension and PAC learning

In what follows, we will explore the reasons why VC dimension is crucial for PAC learnability.

Proposition 2.2.1 *If $\mathcal{H}$ shatters C, $|C| \geq 2m$, then we cannot learn $\mathcal{H}$ with m samples.*

Hence, there will be an assignment of m samples to classes in which we are going to commit a large error.

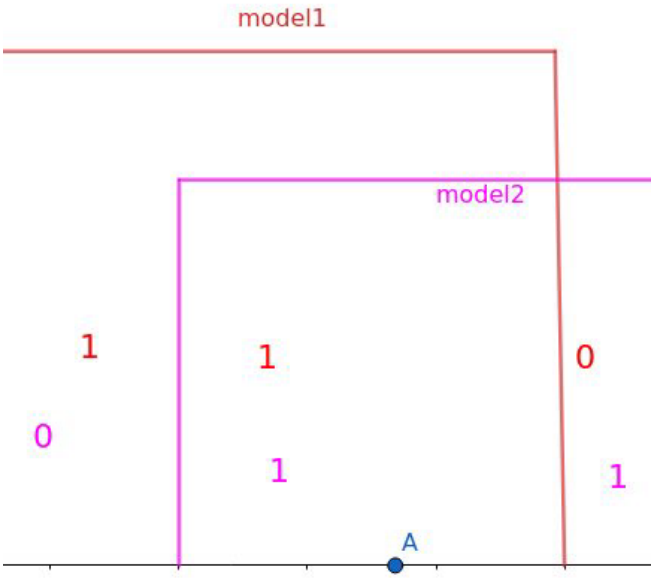

Figure 2.6: Visual interpretation of the theorem: it is impossible to train a model of type $\mathcal{H}_{a+}$ with only a point A with known classification (suppose 1) because the points different from A could have any classification.

If the $VCdim(\mathcal{H}) = \infty$, then $\mathcal{H}$ is not (agnostic) PAC learnable, indeed:

Theorem 2.2.2 *$\mathcal{H}$ is (agnostic) PAC-learnable iff $VCdim(\mathcal{H}) < \infty$.*

In this case: $\exists c_1, c_2$ s.t.

$$c_1 \frac{VCdim(\mathcal{H}) + log(\frac{1}{\delta})}{\varepsilon^2} \leq m_{\varepsilon,\delta} \leq c_2 \frac{VCdim(\mathcal{H}) + log(\frac{1}{\delta})}{\varepsilon^2}$$

Hence VC dimension gives us control on what we can or cannot learn.

2.2.6 Rademacher Complexity *

Consider the data generating distribution $p(x, y)$, a dataset $D \sim p^m$ and an hypotheses class $\mathcal{H} = \{h : X \to \{-1, 1\}\}$.

Definition 2.2.4 *A distribution $\sigma = (\sigma_1, \dots, \sigma_m)$ s.t. $\sigma_i \in \{-1, 1\}$ $\forall i$ and $p(\sigma_i = 1) = 0.5$ is called **Rademacher distribution**.*

Definition 2.2.5 *The **data-dependent Rademacher complexity** is defined as:*

$$\hat{\mathcal{R}}_D(\mathcal{H}) = \mathbb{E}_\sigma \left[\sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^{m} \sigma_i h(x_i) \right]$$

Remark: this is a property both of the function h and of the dataset D.

Observation: $\sum_{i=1}^{m} \sigma_i h(x_i) = \sigma \cdot h(\underline{x})$ is the scalar product of the Rademacher distribution σ with the function h evaluated on our dataset, so that $\frac{1}{m} \sigma \cdot h(\underline{x}) \in [-1, 1]$ essentially is a measure of correlation of h with random noise σ.

Hence, for a specific choice of noise σ, we are going to look at the dataset and choose the best h that correlates with this noise; then we take the expectation w.r.t. σ.

Definition 2.2.6 *The **data-independent Rademacher complexity** is defined as:*

$$\mathcal{R}_m(\mathcal{H}) = \mathbb{E}_{D \sim p^m}[\hat{\mathcal{R}}_D(\mathcal{H})]$$

Remark: this definition takes into account the data generating mechanism p.

Fix $\mathcal{H}$ and $p(x, y)$, then $\forall \delta > 0$ with probability at least $1 - \delta$, $\forall D \sim p^m, |D| = m$, $\forall h \in \mathcal{H}$ we have:

$$R(h) \leq \hat{R}_D(h) + \underbrace{\mathcal{R}_m(\mathcal{H}) + \sqrt{\frac{log(\frac{1}{\delta})}{2m}}}_{\varepsilon_1}$$

$$R(h) \leq \hat{R}_D(h) + \underbrace{\hat{\mathcal{R}}_D(\mathcal{H}) + 3\sqrt{\frac{log(\frac{2}{\delta})}{2m}}}_{\varepsilon_2}$$

Remark: computing the Rademacher complexity can be challenging, as it requires the solution of an optimization problem for any possible value of the Rademacher distribution.

2.2.7 Rademacher complexity and VC dimension *

Definition 2.2.7 *The **growth function** $\Pi_{\mathcal{H}}$ is defined as:*

$$\Pi_{\mathcal{H}} : \mathbb{N} \to \mathbb{N},\ \Pi_{\mathcal{H}}(m) = \max_{C \subseteq X, |C|=m} |\mathcal{H}_C|$$

with $\mathcal{H}_C = \{(h(c_1), \ldots, h(c_m)) | h \in \mathcal{H}, C = \{c_1, \ldots, c_m\}\}$.

Hence the growth function describes how the complexity of what we can explain with our hypothesis set $\mathcal{H}$ increases with the cardinality of the data points that we have.

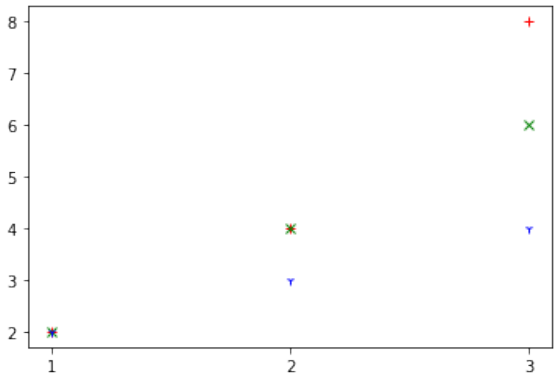

Figure 2.7: Plot of the growth functions of $\mathcal{H}_a$ (blue), $\mathcal{H}_{a+}$ (green) , $\mathcal{H}_\ell$ (red) for $1 \leq m \leq 3$

We can moreover define the $VCdim(\mathcal{H})$ in terms of the growth function:

$$VCdim(\mathcal{H}) = \max\{m | \Pi_{\mathcal{H}}(m) = 2^m\}$$

Intuitively, this comes from the fact that if a set C is shattered by $\mathcal{H}$, then $|\mathcal{H}_C| = 2^m$.

Lemma 2.2.3 *Sauer Lemma:*

$$\Pi_{\mathcal{H}}(m) \leq \sum_{i=0}^{d} \binom{m}{i} \leq \left(\frac{e \cdot m}{d}\right)^d \leq O(m^d)$$

with $d = VCdim(\mathcal{H})$

Moreover it also holds that:

$$\mathcal{R}_m(\mathcal{H}) \leq \sqrt{\frac{2 \log \Pi_{\mathcal{H}}(m)}{m}}$$

We can say that $\mathcal{R}_m(\mathcal{H})$ and $VCdim(\mathcal{H})$ are essentially equivalent, in the sense that when the VC dimension is ∞ then the upper bound on the Rademarcher complexity is independent of m and greater than 1 (if you substitute it into the PAC bound using Rademacher complexity, you get an error which remains always large, no matter how large is m) Otherwise, when the VC dimension is $< \infty$, the bound tends to go to 0 as m grows to infinity.

Summarizing, we have ways to measure the complexity of our hypotheses space: if this complexity is finite (in the sense of VC dimensionality), then what we have as a result is that we can constrain the error and provide bounds that tell us that this error is going towards 0, as we increase the data points (i.e. we will eventually learn). Instead if the VC dimension is infinite, no matter how many data points we have, we are not going to be able to learn, because the complexity of our model is too high, hence it can always overfit the data.

Hence, in order to be able to actually learn, we need to put some constraints in our hypothesis set, and this formalizes our inductive bias.

2.2.8 ERM and Maximum Likelihood

In the maximum likelihood framework:

- we have a dataset $D = \{(x_i, y_i)\}_{i=1,\dots,m}$ s.t. $D \sim p^m, p = p(x, y)$
- we factorize the data generating distribution as: $p(x, y) = p(x)p(y|x)$ and we make hypothesis on $p(y|x)$, trying to express this conditional probability in a parametric form $p(y|x) = p(y|x, \theta)$
- we consider the log-likelihood $L(\theta; D) = \sum_{i=1}^{m} \log p(y_i|x_i, \theta)$

Then we apply the maximum likelihood principle, according to which:

$$\theta_{ML} = \operatorname*{argmax}_{\theta} L(\theta; D) = \operatorname*{argmin}_{\theta} -L(\theta; D)$$

It holds that:

$$\begin{aligned} \operatorname*{argmin}_{\theta} -L(\theta; D) &= \operatorname*{argmin}_{\theta} -\frac{1}{m} \sum_{i=1}^{m} \log p(y_i|x_i, \theta) \\ &\approx \operatorname*{argmin}_{\theta} \mathbb{E}_{p(x,y)}[-\log p(y|x, \theta)] \end{aligned}$$

since the average is an empirical approximation of the expectation.

Definition 2.2.8 $-\frac{1}{m} \sum_{i=1}^{m} \log p(y_i|x_i, \theta)$ *is known as* ***cross entropy****.*

Essentially, in the maximum likelihood framework, the loss function is $l(x, y, \theta) = -\log p(y|x, \theta)$, so we can recast the maximum likelihood principle as a minimization of the (empirical) risk.

For what concerns the space of hypotheses functions $\mathcal{H}$, typical choices are:

- for regression problems: $h(x, \theta) = \mathbb{E}_y[p(y|x, \theta)]$
- for classification problems: $h(x, \theta) = \operatorname{argmax}_y p(y|x, \theta)$ (i.e. the Bayes decision rule)

A typical choice of $p(y|x, \theta)$ in case of regression problems is: $\mathcal{N}(h_\theta(x), \sigma^2)$ so that $-\log(p(y|x, \theta)) \propto (y - \log(x))^2$, that is, the loss as sum of squares comes from the probabilistic assumption that the noise model on the data is Gaussian, centered around the true value.

2.3 KL divergence

Consider a probability distribution $p(x)$, than $-\log p(x)$ is a measure of **self-information**. Indeed, if $p(x) = 1$ then $-\log p(x) = 0$ (no self-information), describing substantially our (lack of) surprise in observing the event. If instead $p(x) = 0$ then $-\log p(x) = \infty$. In general, the more rare the event is, i.e. the lower is $p(x)$, the more self-information it carries (the more surprise its occurrence brings), i.e. the larger is $-\log p(x)$.

In an information-theoretic sense, the **entropy** is a measure of the information that is carried by a random phenomenon, expressed as the expected amount of self-information that is conveyed by a realization of the random phenomenon.

Entropy is formally defined as:

$$H[p] = \mathbb{E}_p[-\log p(x)] = -\int p(x) \log p(x) dx$$

for the continuous case, and:

$$H[p] = -\sum_i p(x_i) \log p(x_i)$$

for the discrete case.

In the discrete case, the maximum entropy is achieved for the uniform distribution and it is equal to $\log K$, with K number of events that can happen. In the continuous case, for a fixed variance, the distribution that maximizes entropy is the Gaussian. The entropy is always 0 if we have a deterministic distribution.

Definition 2.3.1 *Consider two distribution p, q; the **Kullback-Leibler divergence** (also called **relative entropy**) is defined as:*

$$KL[q||p] = \int q(x) \log \frac{q(x)}{p(x)} dx$$

$KL[q||p] = 0$ iff $q = p$.

Intuitively, we are taking a sort of expected difference between p and q, expressed in terms of a log odds ratio. It tells us how different two distributions are: the larger KL the more different are p and q.

Some properties of the KL divergence:

- $KL[q||p]$ is a convex function of q and p and $KL[q||p] \geq 0$
- KL is non-symmetric, i.e. $KL[q||p] \neq KL[p||q]$
- $KL[q||p] = -H[q] - \mathbb{E}_q[\log p]$, where the first term is the entropy and the second term is known as cross-entropy between q and p.

Suppose p is fixed but unknown, $q = q_\theta$ can vary: what we usually do is trying to find the best q_θ that approximates p.

We can do this by finding $\theta^\star = \text{argmin}_\theta \, KL[q_\theta||p]$. This is at the basis of variational inference techniques.

The **mutual information** between x and y is defined as:

$$I[x, y] = KL[p(x, y)||p(x)p(y)]$$

$KL[p(x, y)||p(x)p(y)] = 0$ iff $p(x)$ and $p(y)$ are independent.
Moreover, the more dependent they are, the more different is $p(x, y)$ from the product of the marginals, the more information x carries about y and viceversa.
In other words, the higher the mutual information is, the more knowing y will tell us about x, the less residual uncertainty on x we will have.

Consider a dataset: $\underline{x} : x_1, \ldots, x_N$:

Definition 2.3.2 *The **empirical distribution** is defined as:*

$$p_{emp}(x) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(x = x_i)$$

It is an approximation of the input data generating function $p(x)$.
Practically, the more observations we have, the more the empirical distribution will look like $p(x)$.

Given a distribution q, we can compute:

$$KL[p_{emp}||q] = \mathbb{E}_{p_{emp}}[-\log q(x)] - H(p_{emp}) = -\frac{1}{N} \sum \log q(x_i) - H(p_{emp})$$

If $q = q_\theta$, this is $-\frac{1}{N} L(\theta)$ plus a constant. Hence maximizing $L(\theta)$ is essentially equivalent to minimizing the KL between p_{emp} and q_θ.
This means that we can always rephrase maximum likelihood in terms of cross-entropy.