Introduction to Machine Learning **Fall Semester**

Lecture 4

Lecturer: Lior Wolf *Scribe: Yishay Mansour*

4.1 The PAC Model and Model Selection

In this lecture we introduce the PAC model. The PAC learning model is one of the important learning models. PAC stands for *Probably Approximately Correct*, our goal is to learn a hypothesis from a hypothesis class such that in high confidence we will have a small error rate (approximately correct). We start the lecture with an intuitive example to explain the idea behind the PAC model and the differences between the PAC model and the Bayesian learning that we studied. Then, we continue to formal the PAC model, and give a few examples. In the second part of the lecture we discuss model selection strategies.

4.2 An intuitive example

Suppose we want to predict a 'typical person'. Our input is a sample of different attributes of people which includes their height and weight and their label: whether it is a typical person ('+') or not ('-'). Let H be our hypothesis class. In our example, H will be the set of all possible rectangles on a two-dimensional plane which are axis-aligned (not rotated). One example of hypothesis $h \in H$ can be: mark a person which denotes as (height, weight) to be a typical one ('+') if its description is in the range of:

$$1.60 \leq height \leq 1.90, 60 \leq weight \leq 90$$

Assuming the real target function is a rectangle (this assumption will be addressed later on). Our goal is to find the best rectangle R' that approximates the target rectangle target R. In general, we will try to learn an accurate predictor which will optimize our accuracy.

The PAC learning model is different in spirit from the Bayesian inference, where we assumed that the underlying distribution has a specific form and our goal is to estimate this distribution. In PAC model, we do not know (or assume) anything about the underlying probability distribution of the samples. In our example, the typical people distribution is unknown and finding the joint distribution of height and weight is a difficult task.

Generally, in the PAC model, we do not impose any assumptions on the underlying distribution of the examples other than that such a distribution exists and the examples are independently and identically distributed (i.i.d) according to that distribution. If the test examples were taken from a different distribution than the training examples, there is no

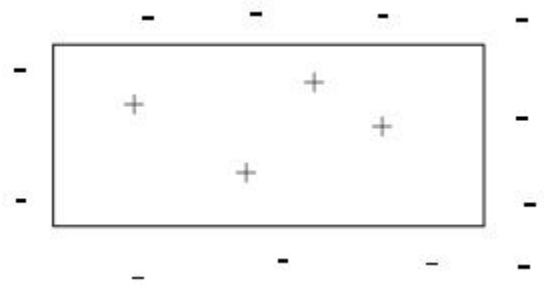


Figure 4.1: A rectangle with positive and negative examples

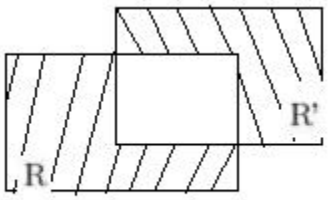


Figure 4.2: R and R' areas including two different error spaces. In our example, since $R' \subseteq R$ there exists only error of $(R - R')$.

reason to expect to learn a low error hypothesis. This critical assumption is at the base of almost all the machine learning approaches.

The learning problem we described can be viewed as follows:

- Goal: Learn rectangle R.
- Input: Examples based on data set and their label: $\langle (x, y), +/- \rangle$.
- Output: R', a good approximation rectangle of the real target rectangle R. (An example of R', see Figure 4.1)

4.2.1 A Good Hypothesis

Our goal is to find a hypothesis that will have a small error rate, smaller than an input parameter ε. Let $R \Delta R'$ be the error of R' in respect to the real target rectangle R. This can be defined by two separate areas: $(R - R') \cup (R' - R)$ (where $(R - R')$ are *false negative* and $(R' - R)$ are *false positive*) as shown in Figure 4.2.

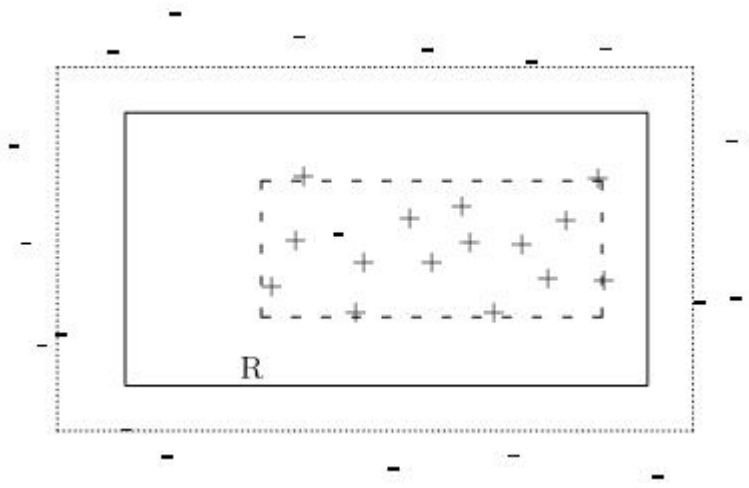


Figure 4.3: The smallest R_{min} (dashed line) and largest R_{max} (dotted line) rectangles border the rectangles which are consistent with the examples. The area between them represents the error region.

Then, our goal can be defined as to find $R' \in H$ such that with probability of at least $1 - \delta$ (confidence),

$$\Pr[error] = \mathcal{D}(R \Delta R') \leq \varepsilon,$$

where we assume that $R \in H$.

4.2.2 Learning Strategy

Let $S = \{\langle (x_1, y_1), b_1 \rangle, ..., \langle (x_m, y_m), b_m \rangle\}$ be the sample. Intuitively, we would like a rectangle that will be *consistent*, i.e., zero error on the training data S. This is possible since we assumed that there exists a rectangle $R \in H$ (the target rectangle) that labels correctly any example. Even with the restriction to output a consistent rectangle $R' \in H$ we have a lot of freedom. We can choose any rectangle from the minimal size (R_{min}) up to the maximal size (R_{max}), as shown in Figure 4.3. We will show later, that it does not matter which consistent rectangle the algorithm returns.

4.3 A formal presentation of the PAC Model

4.3.1 Preliminaries

- The goal of the learning algorithm is to learn an unknown target concept out of a known concept class, for example to learn a particular rectangle out of the set of all

rectangles. Unlike the Bayesian approach, no prior knowledge is assumed.

- Learning occurs in a stochastic setting. Examples from the target rectangle are drawn randomly according to a fixed, unknown probability distribution and are i.i.d.
- We assume that the training and testing samples are generated by the same unknown probability distribution.
- The algorithm should be efficient: the sample size required for obtaining small (ε) error with high ($1-\delta$) confidence is a function of $\frac{1}{\varepsilon}$ and $\ln\frac{1}{\delta}$. Also, we can process the sample within a time polynomial in the sample size.

4.3.2 Definition of the PAC Model

Let the set X be the *instance space* or *example space*. Let $\mathcal{D}$ be any fixed probability distribution over the instance space X. A *concept class* over X is a set

$$\mathcal{C} \subseteq \{c \mid c : X \to \{0,1\}\}.$$

Let c_t be the *target concept*, $c_t \in C$. Let $h \in \mathcal{H}$ be the *learned hypothesis*, where the *hypothesis class* $\mathcal{H}$. Generally, $\mathcal{H}$ may be a different concept class than $\mathcal{C}$. We will define the *error* of h with respect to the distribution $\mathcal{D}$ and the target concept c_t as follows:

$$error(h) = \Pr_{\mathcal{D}}[h(x) \neq c_t(x)] = D(h \Delta c_t(x))$$

Let $EX(c_t, \mathcal{D})$ be a procedure (we will sometimes call it an *oracle*) that runs in a unit time, and on each call returns a labeled example $\langle x, c_t(x)\rangle$, where x is drawn independently from $\mathcal{D}$. In the PAC model, the oracle is the *only* source of examples for the learning algorithm.

Definition Let $\mathcal{C}$ and H be concept classes over X. We say that C is *PAC learnable by* H if there exists an algorithm A with the following property: for every concept $c_t \in \mathcal{C}$, for every distribution $\mathcal{D}$ on X, and for all $0 < \varepsilon, \delta < \frac{1}{2}$, if A is given access to $EX(c_t, \mathcal{D})$ and inputs ε and δ, then with probability at least $1-\delta$, A outputs a hypothesis concept $h \in H$: If $c_t \in H$ (Realizable case), then h is satisfying

$$error(h) \leq \varepsilon$$

If $c_t \notin H$ (Unrealizable), then h is satisfying

$$error(h) \leq \varepsilon + \min_{h' \in H} error(h')$$

We say that $\mathcal{C}$ is *efficiently PAC learnable*, if A runs in time polynomial in $\frac{1}{\varepsilon}$, $\ln\frac{1}{\delta}$, n and m where n is the size of the input and m the size of the target function, (for example, the

number of bits that are needed to characterize it). Implicit, we mean that it is "easier" to learn a "simpler" target function.

4.4 Finite Hypothesis Class

In this section, we show how to learn a good hypothesis from a finite hypothesis class $\mathcal{H}$. The idea is to bound the probability that a hypothesis h is ε-*bad*, where a hypothesis h is ε-*bad* if $error(h) > \varepsilon$.

4.4.1 The Realizable case ($c_t \in \mathcal{H}$)

Generally, given $m(\varepsilon, \delta)$ examples, we will request our algorithm A to find an h which is *consistent* with the training sample S, i.e., classifies all the examples in S correctly, which means that $\forall x \in S$ we have $h(x) = c_t(x)$. The algorithm A will succeed to learn if h is not ε-bad, i.e., $error(h) < \varepsilon$. We note that at least one *consistent* hypothesis exists because we assume that $c_t \in \mathcal{H}$.

We bound the probability of algorithm A to return h that is ε-bad by bounding by δ the probability that some consistent hypothesis is ε-bad. To start, fix an h which is ε-bad:

$$\Pr[h(x_i) = c_t(x_i)\ for\ 1 \leq i \leq m] \leq (1-\varepsilon)^m < e^{-\varepsilon m},$$

where the first inequality is derived from the fact that since h is ε-bad we have $\Pr[h(x) = c_t(x)] \leq 1 - \varepsilon$ and the fact that the examples are sampled i.i.d from distribution $\mathcal{D}$. Let H_ε be the set of hypotheses that are ϵ-bad. Now we bound the probability of failure of algorithm A, as follows:

$$\begin{aligned}
& \Pr[\text{A returns an } \varepsilon\text{-bad hypothesis}] \\
= & \sum_{h \in H_\varepsilon} \Pr[\text{A returns } h] \\
\leq & \sum_{h \in H_\varepsilon} \Pr[h(x_i) = c_t(x_i)\ for\ 1 \leq i \leq m] \\
\leq & |H_\varepsilon| e^{-\varepsilon m} \\
\leq & |\mathcal{H}| e^{-\varepsilon m},
\end{aligned}$$

The first inequality follows from the fact that a necessary condition for A returning h, is for h to be consistent with the data. The second inequality follows from the analysis for a fixed ε-bad hypothesis, and the third inequality is immediate since $H_\varepsilon \subseteq \mathcal{H}$.

In order to satisfy the condition for PAC learning, we bound the failure probability by δ:

$$|\mathcal{H}| e^{-\varepsilon m} \leq \delta,$$

which lower bounds the sample size,

$$m \geq \frac{1}{\varepsilon} \ln \frac{|\mathcal{H}|}{\delta} .$$

We note that for a finite class of hypothesis H, any consistent algorithm A PAC learns $C = H$ using H. As expected the bound tells us that the larger the hypothesis class H, the larger the sample we require. However it is not clear how to implement the consistent algorithm efficiently, and in general it will not be efficient. Also, the above analysis only works for finite classes.

4.4.2 The Unrealizable case ($c_t \notin \mathcal{H}$)

In this case, we need to relax our goal, since a small error hypothesis might not exist in H. As we defined before, our goal is to learn according to the "best" hypothesis in the hypothesis class (i.e., with the minimal error). Let h^* be the hypothesis with the minimal error, i.e., for every $h \in \mathcal{H}$:

$$0 < error(h^*) \leq error(h) .$$

Let β be $\beta = error(h^*) = \min_{h \in H} error(h)$. Our goal is to find a hypothesis h that will achieve:

$$error(h) \leq \beta + \varepsilon .$$

Note that this is a generalization of our previous system (in which $\beta = 0$).

Given $m(\varepsilon, \delta)$ training examples S, we define $\widehat{error}(h)$ to be the empirical error of h on sample S. Formally:

$$\widehat{error}(h) = \frac{1}{m} \sum_{i=1}^{m} I(h(x_i) \neq c_t(x_i)),$$

where I is the indicator function.

We also need to define the algorithm, since now it may be impossible to choose a consistent h. Algorithm A will return a hypothesis $\bar{h}$ that will have the minimal empirical error. That is, $\bar{h} = \operatorname{argmin}_{h \in H} \widehat{error}(h)$ (If there exists more than one hypothesis, the algorithm will pick one of them.) This algorithm is called *Empirical Risk Minimization (ERM).*

We will now bound the sample size required for obtaining a good hypothesis using ERM. We want to choose a sample size m such that for any hypothesis, the difference between the true and the observed error is small. That is, with probability at least $1 - \delta$:

$$\forall h \in H, |\widehat{error}(h) - error(h)| \leq \frac{\varepsilon}{2}.$$

If we have a uniform good error estimation, we can derive easily that using the hypothesis $\bar{h}$ from algorithm ERM, we obtain:

$$error(\bar{h}) \leq \widehat{error}(\bar{h}) + \frac{\varepsilon}{2} \leq \widehat{error}(h^*) + \frac{\varepsilon}{2} \leq error(h^*) + \varepsilon,$$

where the first and third inequality hold since the difference between the true and the observed error is small (up to $\frac{\varepsilon}{2}$), and the second inequality holds since ERM selects the hypothesis with the minimal empirical error.

To conclude our bound for the sample size we need to bound the probability of failure in estimating $error(h)$. We will use Chernoff bound[1] and derive that,

$$\Pr[|\widehat{error}(h) - error(h)| \geq \frac{\varepsilon}{2}] \leq 2e^{-2(\frac{\varepsilon}{2})^2 m}$$

We can use Chernoff bounds since the m training examples are drawn i.i.d from the same distribution $\mathcal{D}$. We can get a bound on the sample size by requiring the probability of failure over $\mathcal{H}$ to be smaller than δ:

$$\begin{aligned}
&\Pr[\exists h \in \mathcal{H} \text{ s.t. } |\widehat{error}(h) - error(h)| \geq \frac{\varepsilon}{2}] \\
\leq &\sum_{h \in \mathcal{H}} \Pr[|\widehat{error}(h) - error(h)| \geq \frac{\varepsilon}{2}] \\
\leq &2|\mathcal{H}|e^{-2(\frac{\varepsilon}{2})^2 m} \leq \delta
\end{aligned}$$

where the first inequality is given by the union bound. Hence,

$$m \geq \frac{2}{\varepsilon^2} \ln \frac{2|\mathcal{H}|}{\delta}$$

We established a bound on the sample size for PAC learning when $c_t \notin \mathcal{H}$.

Note that the sample size depends on $\frac{1}{\varepsilon^2}$ instead of $\frac{1}{\varepsilon}$ in the realizable case ($c_t \in \mathcal{H}$). This results from the difference between requiring a single counter-example to disqualify a hypothesis (since for some hypothesis $error(h) = 0$), to the requiring many examples to disqualify a hypothesis, by estimating the observed error.

4.4.3 Example - Learning Boolean Disjunctions

To demonstrate the PAC model we consider the example of learning Boolean disjunction functions. The problem is defined as follows: Given a set of boolean variables $T = \{x_1, ..., x_n\}$ and a set of literals $L = x_1, \bar{x}_1, \ldots x_n, \bar{x}_n$, we need to learn an Or function over the literals, for example: $x_1 \vee \bar{x}_3 \vee x_5$. Let C be the set of all possible disjunctions. We have that $|C| = 3^n$, since for each variable x_i the target disjunction c_t may contain x_i, $\bar{x}_i$, or neither. We will assume $H = C$.

[1] See more about concentration bounds in Section 4.8

ELIM algorithm for learning boolean disjunctions

Given a negative example we can eliminate all literals that evaluate to 1. For example, if given an example 001 (assume 0 was the value of x_1 and x_2 and 1 was the value of x_3) which was negative we can eliminate the literals $\bar{x}_1, \bar{x}_2, x_3$. This is valid since if one of them was in the target disjunction the target function would be positive. The algorithm uses this fact and eliminates all the inconsistent literals. We also notice that the algorithm classifies the positive examples correct because the literal in the target disjunction are never eliminated, and we have $c_t \subseteq L_{\text{final}}$.

The algorithm ELIM initializes a set $L = x_1, \bar{x}_1, \ldots x_n, \bar{x}_n$, and for each negative sample Z it updates $L = L - \{z_i | z_i \in Z\}$.

From previous sections we can see that the algorithm learns when the sample size:

$$m > \frac{1}{\varepsilon} \ln \frac{|\mathcal{H}|}{\delta} = \frac{1}{\varepsilon} \ln \frac{3^n}{\delta} = \frac{n \ln 3}{\varepsilon} + \frac{1}{\varepsilon} \ln \frac{1}{\delta}$$

4.5 Infinite Hypothesis Class

We have already seen an example of PAC learning from an infinite concept class. The theoretical approach in this case will be briefly presented later in this lecture and in greater detail in a future talk. Here is another example:

Let X be the interval $[0, 1]$.

Let $\mathcal{H}$ be a concept class over X:

$$\mathcal{H} = \{c_\theta \mid 0 \leq \theta \leq 1\}$$

and

$$c_\theta = \begin{cases} 1 & x \geq \theta \\ 0 & \text{otherwise} \end{cases}$$

4.5.1 The Case $c_t \in \mathcal{H}$

Proof I

Let's choose m examples and define:

$$min\{x \mid c_t(x) = 1\} \; = \; max,$$

and

$$max\{x \mid c_t(x) = 0\} \; = \; min.$$

We shall choose a value $\theta \in [min, max]$ and return c_θ. It is enough to show that with probability $1 - \delta$ the weight of the interval $[min, max]$ under $\mathcal{D}$ is at most ε. This is sufficient

because errors will occur only on this interval. Let's analyze the probability that m samples will not be taken from the interval $[min, max]$. Suppose

$$\mathcal{D}([0_{max}, 1_{min}]) \geq \varepsilon$$

The probability that we did not sample in $[min, max]$ is $(1-\varepsilon)^m$, then using the inequality $(1-x) \leq e^{-x}$ we get:

$$(1-\varepsilon)^m \leq e^{-\varepsilon m} \leq \delta$$

so it is enough that:

$$m \geq \frac{1}{\varepsilon} ln \frac{1}{\delta}.$$

This proof is wrong - why?
Note that min and max where determined only by the sample. Therefore, in the sample there is a point between them! Consequently we can not make a probabilistic assumption on the sample, since it is already fixed. To correct we will need to define the events independent of the sample.

Proof II
We shall define parameters which do not depend on the sample. Let $max^{'}$ and $min^{'}$ be the points which are $\varepsilon/2$ close to θ. Namely, $max^{'} : D[\theta, max^{'}] = \varepsilon/2$ and $min^{'} : D[min^{'}, \theta] = \varepsilon/2$. Note that θ is the target we try to learn. The goal is to show that with high probability $(1-\delta)$: $max \in [\theta, max^{'}]$, and $min \in [min^{'}, \theta]$. In this case any value between $[min, max]$ is good and could be returned by the algorithm, since now $D[min, max] < \varepsilon$, and D is the real distribution (not just the sample).

We have that

$$Prob[x_1, x_2, \ldots, x_m \notin [min^{'}, \theta]] = (1 - \frac{\varepsilon}{2})^m < e^{\frac{-m\varepsilon}{2}}$$

The failure probability for $[\theta, max^{'}]$ is derived similarly. Finally we get:
$2e^{\frac{-m\varepsilon}{2}} < \delta$.

Hence: $m > \frac{2}{\varepsilon} ln \frac{2}{\delta}$.

Remember that $min^{'}, max^{'}$ where chosen with no dependence on the sample.

4.5.2 The Case $c_t \notin \mathcal{H}$

This case corresponds to noisy data or a more complicated labeling function than those represented by $\{c_\theta\}$.

In this case we can not find some c^*_θ which we would like to approximate because functions with a similar error may use values of θ very far from each other.

For a sample size m, there are $m+1$ possible hypotheses according to the intervals between samples (the values of θ within an interval are equivalent in the sense that they produce the same learning error.) We can thus choose the hypothesis which will minimize the error on the examples.

For a given distribution $\mathcal{D}$, let $0 = z_0 < z_1 \cdots < z_k = 1$ such that $\mathcal{D}([z_i, z_{i+1}]) = \frac{\varepsilon}{4}$ (this is called a $\frac{\varepsilon}{4}$ net). Let h_{alg} be the output of the learning algorithm, h^* an optimal hypothesis, and h_{z_i} the hypothesis with $\theta = z_i$. Our definition of the $\{z_i\}$ implies that there exist z_j and z_k such that:

$$|error(h_{z_j}) - error(h^*)| \leq \frac{\varepsilon}{4}\,,$$

and

$$|error(h_{z_k}) - error(h_{\text{alg}})| \leq \frac{\varepsilon}{4}\,,$$

We will show that when the sample size is large enough, for every z_i, with a high probability, $|error(h_{z_i}) - obs\_error(h_{z_i})| \leq \frac{\varepsilon}{4}$. That implies $error(h_{\text{alg}}) - error(h^*) \leq \varepsilon$.

Let

$$q_i = Pr_{\mathcal{D}}[c_t(x) \neq h_{z_i}(x)],$$

and $\hat{q}_i$ be the estimate of q_i calculated from the sample. Using Chernoff bounds we get

$$Pr[\ |\hat{q}_i - q_i| \geq \frac{\varepsilon}{4}] \leq e^{-(\frac{\varepsilon}{4})^2 m}\,.$$

Since the number of z_i's is $\frac{4}{\varepsilon}$, the probability of an estimation error larger than $\frac{\varepsilon}{4}$ may be bounded by

$$Pr[\ |error(h_{z_i}) - obs\_error(h_{z_i})| \geq \frac{\varepsilon}{4}] \leq \frac{4}{\varepsilon} \cdot e^{-(\frac{\varepsilon}{4})^2 m} < \delta\,.$$

Therefore, we have to choose a sample whose size is:

$$m \geq \frac{16}{\varepsilon^2}[\ln\frac{4}{\varepsilon} + \ln\frac{1}{\delta}]\,.$$

4.5.3 General ε-Net approach

An ε-net is a set $\mathcal{G} \subset \mathcal{H}$ such that for every $h \in \mathcal{H}$ there exists $g \in \mathcal{G}$ such that

$$Prob_{\mathcal{D}}[g(x) \neq h(x)] \leq \varepsilon\,.$$

The algorithm will first find an $\frac{\varepsilon}{4}$-net for $\mathcal{H}$ which will be represented by the set $\{h_{z_i}\}$, and return the optimal h_i of the net. In general, the sample size required for a confidence

of $1-\delta$ when an $\frac{\varepsilon}{4}$ of $\mathcal{G}$ is known is

$$m \geq \frac{16}{\varepsilon^2} \ln \frac{|\mathcal{G}|}{\delta} .$$

The derivation of that bound is similar to that done in the preceding section.

We can use the above argument to reduce many infinite size concept classes to finite one. Here are two interesting examples.

1. **Polynomials**. Let $f_a(x) = 1$ if $\sum_{i=0}^{d} a_i x^i > 0$ and otherwise 0, and assume that $x \in [0,1]$. With the appropriate discretization we can show that there is an ϵ-net with $O((1/\epsilon)^{d+1})$ values of a. This will translate to an "effective hypothesis class size" of $\log |H| = O(d \log(1/\epsilon))$.

2. **Hyperplanes**. Let $f_w(x) = 1$ if $\sum_{i=0}^{d} w_i x_i > 0$ and otherwise 0, and assume that $x \in [0,1]$. Again, with the appropriate discritization we can show that there is an ϵ-net with $O((1/\epsilon)^{d+1})$ values of a. This will translate to an "effective hypothesis class size" of $\log |H| = O(d \log(1/\epsilon))$.

4.5.4 The VC Dimension

The question of determining the sample size required for achieving a given confidence when the hypothesis class is infinite is generally addressed by the *VC dimension* (Vapnik-Chervonenkis).

The VC dimension of H is the maximal number of points, d, which can be labeled in all the 2^d possible combinations by concepts from $\mathcal{H}$. The VC dimension is a substitute for $\ln |\mathcal{H}|$ in the expressions for sample size bounds.

The notion of VC dimension avoids the need of discretization and the loss associated with it.

4.6 Model Selection - Introduction

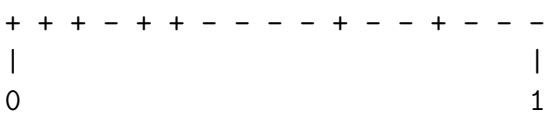
(This section is non-mandatory.)

So far, each learning model determined the number of examples needed in order to learn a concept class. However, in many real cases, only a limited number of examples is available, and the learning algorithm is supposed to come up with the best hypothesis it can from the available data.

In the algorithms discussed previously, we solved accuracy problems of our hypothesis by requiring a sufficiently large number of examples, which reduces the probability of the hypothesis' error. We now deal with the case in which this cannot be done.

4.6.1 Discussion

To demonstrate the problem, let's look at the concept class of a finite union of intervals on the line [0,1]. Let us assume that we're given the following examples in the interval [0,1] :



The target concept c_t is a set of intervals within [0,1].

Obviously, if we allow a sufficiently large number of intervals, we could easily come up with a hypothesis that is completely consistent with the data (e.g. surround every positive point with its own tiny positive interval). However, we want to predict the correct classifications also for examples other than the original training set.

Adding more intervals to our hypothesis reduces the hypothesis' error on the training set, but may increase its error on new examples. For example, a positive interval surrounding a positive point may consist in the target concept of a 2/3 negative sub-interval and a 1/3 positive sub-interval, so adding this interval to the hypothesis can increase its "real" error. This way we may get hypotheses which are overfitted to the data, and may not generalize well to new examples.

Therefore, by Occam's Razor, in such cases we prefer simpler hypotheses which may have some error on the training set, but with high probability will predict better future observations. Returning to our example, we can make a table of the amount of errors generated by a hypothesis related to the number of intervals in the hypothesis :

Number of Intervals:	0	1	2	3	4	5	6	7	...
Number of Errors:	7	3	2	1	0	0	0	0	...

We can see that the more complex the hypothesis is, the smaller its error on the given examples. Beyond a certain complexity, all hypotheses yield 0 errors. So far, we've considered only those hypotheses which yield 0 errors on the training set, but now we're limited to the given examples and these examples may not be representative of the domain. Therefore, we want to consider simpler hypotheses, which may have some errors on the training set but generalize better to new examples.

To make the things worse, there is still the problem of noise. For a hypothesis to be completely consistent with the data, it becomes very complex. However, some of the inconsistencies in the data may be due to "noise", and the true concept may be much simpler than our consistent hypothesis. In the given example, the true concept may consist of a single interval (e.g. [0, 1/2]), and the inconsistent examples were generated due to noise. In such a case, adapting our hypothesis to the data causes the noise to get into the hypothesis.

So now we have to deal with a sample set which may be too small to accurately represent the domain, and may itself be "noisy".

In the following sections we'll consider different models for dealing with this problem. But first we'll start with building the theoretical model.

4.7 Theoretical Model

4.7.1 The Setup

Let us consider the following theoretical model.

Let H_i be the class of hypotheses, all having the same complexity-level, i (where i is some definition of complexity). Clearly, we get nested hypothesis classes :

$$H_1 \subseteq H_2 \subseteq \cdots \subseteq H_i \subseteq \cdots$$

any hypothesis of a lower complexity is included in any class of hypotheses of a higher complexity. Let $H = \cup_{i=1}^{\infty} H_i$.

For the sake of simplicity, we will assume

$$|H_i| = 2^{i-1}.$$

Let c^* be the target concept. In contrast to our previous methods, we now do *not* assume that c^* is included within one of the H_i. The objective of the learning algorithm will be to produce a hypothesis which is "sufficiently close" to c^* (but not necessarily c^* itself).

4.7.2 Definitions

- $\epsilon(h)$ - the "real" error of h, i.e. the error of h over the entire domain X.

$$\epsilon(h) = Prob\big[h(x) \neq c^*(x)\big]$$

- ϵ_i - the lowest error found for any of the hypotheses in class H_i.

$$\epsilon_i = \min_{h \in H_i} \big\{\epsilon(h)\big\}$$

  Note that since $H_i \subseteq H_{i+1}$, $\epsilon_{i+1} \leq \epsilon_i$ (the probability of error decreases as the complexity level increases).

- ϵ^* - the optimal error level, i.e. the value towards which ϵ_i converges as i increases.

$$\epsilon^* = \inf_i \{\epsilon_i\}$$

  It might be that ϵ^* will not be obtained by any hypothesis $h \in H$, but it is the lower-bound on any ϵ_i and could be approximated arbitrarily well. If for some i, $c^* \in H_i$ then $\epsilon^* = 0$.

- $\hat{\epsilon}(h)$ - the observed error, i.e. the error of hypothesis h on the given examples.

$$\hat{\epsilon}(h) = \frac{1}{m} \sum_{x_i \in S} I(h(x_i) \neq c_t(x_i)) \ ,$$

  where S is the given set of m examples.

- $\hat{\epsilon}_i$ - the lowest observed error of any of the hypotheses in H_i.

$$\hat{\epsilon}_i = \min_{h \in H_i} \{\hat{\epsilon}(h)\} \ .$$

4.7.3 The Problem: Overfitting

As the complexity level i of the hypothesis increases, its error on the given data $\hat{\epsilon}_i$ is reduced. Beyond complexity level m (where m is the number of examples in the given set) all the $\hat{\epsilon}_i$ will equal 0.

This will happen even when the same hypothesis' real error-level, $\epsilon(h)$ (i.e., measured over the entire domain), is greater than 0, and even when $\epsilon^* >> 0$.

This happens because at high levels of complexity, the hypotheses (with the lowest levels of error on the given data) become too fitted to the given data. This phenomenon is called *overfitting*.

In our case, we can not require a sufficiently large set of examples. The given data may be too small to accurately represent the entire domain. The presence of noise makes the given data even less representative of the entire domain. Thus, the overfitted hypothesis might turn out to be quite far from the true concept.

The simplistic approach for finding a good hypothesis would be to choose a hypothesis g which has the lowest value of $\hat{\epsilon}(g)$:

$$g = arg \min_{h \in \cup H_i} \{\hat{\epsilon}(h)\}$$

However, using this simplistic approach for choosing g will cause us to prefer overfitted hypotheses, because they yield the lowest $\hat{\epsilon}(h)$, namely zero observed error.

4.7.4 Penalty Based Model Selection

One way to overcome the overfitting problem is to impose a complexity penalty on the complexity of the chosen hypothesis; we will then try to minimize both the observed error of the chosen hypothesis and its complexity penalty.

The chosen hypothesis g^* will, therefore, be defined as

$$g^* = arg \min_{g \in \cup H_i} \left\{ \hat{\epsilon}(g) + Penalty(g) \right\} ,$$

where $Penalty(g)$ depends on the complexity of g.

We will define a measure $d(h)$ for the complexity of a hypothesis h as the lowest complexity level i such that h is found in H_i:

$$d(h) = \min_i \left\{ h \in H_i \right\} .$$

Since the penalty is calculated based on $d(h)$, which is the first class in which h is found, the penalty will be the same for all hypotheses with the same complexity.

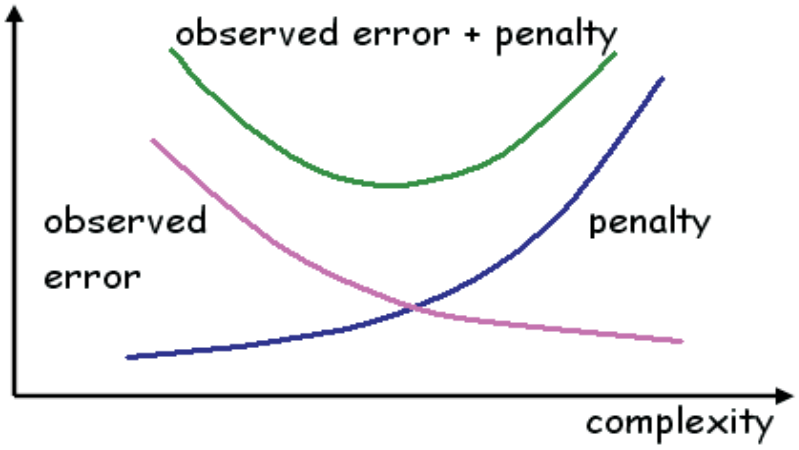


Figure 4.4: Principle of penalty based models.

Figure 4.4 shows the principle of penalty based models. As the complexity level of the hypothesis increases, its observed error is reduced but the penalty for its complexity increases. The penalty based model will try to find the minimum of the sum of the observed error and the penalty. Thus we will choose hypotheses that are not too fitted to the given examples.

4.7.5 SRM: Structural Risk Minimization

The Model

The *SRM* (*Structural Risk Minimization*) model is a penalty based model, which uses the following as the *Penalty* :

$$Penalty(h) = \sqrt{\frac{2d(h) \cdot \ln(2) + \ln(1/\delta)}{m}} \ , \tag{4.1}$$

where m is the number of examples, and δ is a confidence parameter (its meaning will be clear in the following section). This penalty defines a tradeoff between the complexity of the hypothesis and the size of the given sample. The hypothesis g^* chosen by the *SRM* model will therefore be:

$$g^* = arg \min_{g \in H} \left\{ \hat{\epsilon}(g) + Penalty(g) \right\} \tag{4.2}$$

Analysis

Let h^* be the best possible hypothesis there is in H, i.e., the hypothesis with the lowest actual error-level (error measured over the entire domain):

$$h^* = arg \min_{h \in H} \{ \epsilon(h) \} \ . \tag{4.3}$$

Let g^* be the hypothesis chosen by *SRM*, i.e. (4.2).

Theorem 4.1 (*SRM* Theorem) *With probability of at least* $1 - \delta$ *the actual error of* g^* *is smaller than or equal to the actual error of* h^* *plus twice the* SRM *complexity-penalty of* h^**. Formally :*

$$\epsilon(g^*) \leq \epsilon(h^*) + 2 \cdot Penalty(h^*) \tag{4.4}$$

Recall that by definition (of h^*) the actual error of h^* is smaller than or equal to the actual error of g^*. So, according to the *SRM* theorem, the actual error of g^* is bounded on both sides by:

$$\epsilon(h^*) \ \leq \ \epsilon(g^*) \ \leq \ \epsilon(h^*) + 2 \cdot Penalty(h^*) \tag{4.5}$$

It can be clearly seen from this inequality that the larger the number of examples (the larger m), the smaller the value of the complexity-penalty becomes, and the difference between the two hypotheses diminishes.

For the proof of the *SRM* theorem, we'll use the following claim :

Claim 4.2 *The probability that the observed error of h ($\hat{\epsilon}(h)$) will diverge from the actual error of h ($\epsilon(h)$) by more than some threshold, λ, is bounded from above:*

$$Prob\Big[|\epsilon(h) - \hat{\epsilon}(h)| \geq \lambda\Big] \leq 2e^{-\lambda^2 m} \tag{4.6}$$

Proof: This is obtained by simple application of the Chernoff Inequality. □

Proof of *SRM* Theorem

The proof consists of two stages. First, we'll bound the error of the hypothesis in any given class H_i. Second, we'll bound the error across the classes H_i.

First stage : Bounding the error in H_i
Let g_i be the hypothesis with the lowest observed error in H_i:

$$g_i = arg \min_{h \in H_i} \{\hat{\epsilon}(h)\}$$

We want to estimate the probability of difference between the actual error and the observed error of g_i:

$$Prob\Big[|\epsilon(g_i) - \hat{\epsilon}(g_i)| \geq \lambda_i\Big]$$

(we use λ_i, because it will depend on the complexity-level i).

We cannot use Claim 4.2 directly to bound this probability, because g_i is determined according to the given sample set (and in Claim 4.2 the probability is computed over all the possible sample sets).

However, we can bound this probability P by the probability that *any* hypothesis in H_i will have the difference between the actual error and observed error larger than λ_i:

$$P \leq Prob\Big[\exists h \in H_i \mid |\epsilon(h) - \hat{\epsilon}(h)| \geq \lambda_i\Big].$$

By applying the inequality of Claim 4.2 we obtain:

$$Prob\Big[\exists h \in H_i \mid |\epsilon(h) - \hat{\epsilon}(h)| \geq \lambda_i\Big] \leq |H_i| \cdot 2e^{-\lambda_i^2 m}.$$

Recall that we assumed for simplicity that $|H_i| = 2^{i-1}$, so we get :

$$Prob\Big[\exists h \in H_i \mid |\epsilon(h) - \hat{\epsilon}(h)| \geq \lambda_i\Big] \leq 2^i \cdot 2e^{-\lambda_i^2 m}. \tag{4.7}$$

Let's define this upper bound (the probability of error for any hypothesis in H_i) as δ_i , i.e.:

$$\delta_i = 2^{i-1} \cdot 2e^{-\lambda_i^2 m}. \tag{4.8}$$

Solving for λ_i we get:

$$\lambda_i^2 m = \ln\left(\frac{2^i}{\delta_i}\right),$$

$$\lambda_i = \sqrt{\frac{i \cdot \ln(2) + \ln(1/\delta_i)}{m}}\,. \tag{4.9}$$

If we set the upper bound δ_i for each class H_i to $\delta_i = \frac{\delta}{2^i}$ (i.e., splitting the confidence level δ between the different classes), then we get $\delta = \sum_i \delta_i$ and thus we can use the union bound to get :

$$Prob\Big[\forall i\ \forall h \in H_i \mid |\epsilon(h) - \hat{\epsilon}(h)| \le \lambda_i\Big] = 1 - Prob\Big[\exists i\ \exists h \in H_i \mid |\epsilon(h) - \hat{\epsilon}(h)| \ge \lambda_i\Big]$$

$$\ge 1 - \sum_i \delta_i = 1 - \delta \tag{4.10}$$

Therefore, with probability of at least $1 - \delta$,

$$|\epsilon(h) - \hat{\epsilon}(h)| \le \lambda_i \tag{4.11}$$

for any hypothesis h in $\cup H_i$.

In this case λ_i is as follows:

$$\lambda_i = \sqrt{\frac{i \cdot \ln(2) + \ln(2^i/\delta)}{m}} = \sqrt{\frac{2i \cdot \ln(2) + \ln(1/\delta)}{m}} \tag{4.12}$$

Second stage : Bounding the error across H_i
In the previous stage, we've proved that with probability of at least $1-\delta$, for any hypothesis h in $\cup H_i$,

$$|\epsilon(h) - \hat{\epsilon}(h)| \le \lambda_i$$

where λ_i depends on the complexity level i of h.
Among other hypotheses, this is also true for h^* and g^*, which leads to the following:

$$\hat{\epsilon}(h^*) \le \epsilon(h^*) + \lambda_i \tag{4.13}$$

$$\epsilon(g^*) - \lambda_j \le \hat{\epsilon}(g^*) \tag{4.14}$$

where,

- $i = d(h^*)$, i.e., i is the complexity level of h^*.
- $j = d(g^*)$, i.e., j is the complexity level of g^*.

Let's define P_i, P_j as the *SRM* complexity-penalties for h^* and g^*, respectively. Therefore, from the definition of the *SRM* model we get :

$$\hat{\epsilon}(g^*) + P_j \leq \hat{\epsilon}(h^*) + P_i \tag{4.15}$$

(otherwise the *SRM* model would not have chosen g^*).

From the three inequalities (4.13), (4.14) and (4.15) we get:

$$\epsilon(g^*) - \lambda_j + P_j \leq \epsilon(h^*) + \lambda_i + P_i \tag{4.16}$$

and therefore,

$$\epsilon(g^*) \leq \epsilon(h^*) + \lambda_i + P_i + \lambda_j - P_j \ . \tag{4.17}$$

Now, from the definition of the penalty-value for complexity-level j we get :

$$P_j = \sqrt{\frac{(2j+1)\ln(2) + \ln(1/\delta)}{m}} = \lambda_j \tag{4.18}$$

Hence the penalty is greater than the actual divergence with probability of at least $1 - \delta$.

Now we can return to inequality (4.17) and get from (4.18):

$$\epsilon(g^*) \leq \epsilon(h^*) + \lambda_i + P_i + (\lambda_j - P_j) = \epsilon(h^*) + 2 \cdot P_i \tag{4.19}$$

which proves the *SRM* theorem. □

4.8 Concentration bounds

Here are a few simple concentration bounds from probability theory.

Markov Inequality

Let X be a non-negative random variable (r.v.) and $E[X]$ the *expected value* (mean) of X, then:

$$Pr[X \geq \alpha] \leq \frac{E[x]}{\alpha}$$

4.8.1 Chebyshev Inequality

Suppose that X is arbitrary with expectation $E[X] = \eta$ and variance $Var(X)$, then:

$$Pr[|x - \eta| \geq \beta] \ \leq \ \frac{Var(x)}{\beta^2}$$

4.8.2 Chernoff Inequality

Let the sequence of random variables $X_1, \ldots, X_n$ where $X_i \ \in \ \{0, 1\}$ be independent identically distributed (i.i.d.), and $Pr[X_i = 1] = p$, then:

$$Pr[|\sum_{i=1}^{n} \frac{X_i}{n} - p| \ \geq \ \lambda] \ \leq \ 2e^{-2\lambda^2 n}$$

The last inequality is especially interesting, since it gives us a tool to argue "how fast" the "observed mean" would converge to the "true mean". It can be viewed as a "quantitative form of the Law of Large Numbers.