Machine Learning: Foundations **Fall semester, 2012**

Lecture 12: January 20, 2012

Lecturer: Yishay Mansour *Scribe: Yuval Globerson, Ilia Gorelik, Oleg Zlydenko*[1]

11.1 Model Selection - Introduction

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.

One example for such a case is when we have a class of an infinite $VCdim$. As we've seen in the previous lectures, if a concept class C has $VCdim = \infty$ then C is not learnable by any static learning algorithm, i.e., for any number of examples we will always be able to find a bad hypothesis which is consistent with the examples. So how can we learn such a class?

11.1.1 Naive Algorithm

Say we have a countable class of hypothesis $H = \{h_1, h_2, ...\}$ and a target function c^*. We will assume, for simplicity, that $c^* \in H$. We will describe a PAC algortihm that learns H.

Algorithm

```
For each $i$:
Sample $m_i = \frac{1}{\epsilon} \cdot \ln(\frac{1}{\delta_i})$ examples.
If $h_i$ is consistent with the $m_i$ samples, stop and return $h_i$.
```

Analysis

1. The algorithm will always stop. Since $c^* \in H$, there exists j s.t. $h_j = c^*$. Therefore, the algorithm will stop at most after j iterations.

[1]Based on a scribe written by Oana Sidi, Inbal Avraham and Vera Vsevolohzky (January, 2011), Gil Freundlich (June, 1996) and Roi Yehoshua, Ophir Gvirtzer, Zohar Ganon (May,2002).

2. Assume the algorithm stopped at iteration number i.

   $Pr[h_i\ is\ consistent | error(h_i) \geq \epsilon] = (1 - error(h_i))^{m_i} \leq e^{-\epsilon \cdot m_i} = \delta_i$

3. In order for the algorithm to be PAC, we demand that $\sum_{i=1}^{\infty} \delta_i = \delta$. If this is true, then by Union Bound:

   $$Pr[\exists i\ s.t\ h_i\ is\ consistent\ on\ m_i | error(h_i) \geq \epsilon] \leq \delta \qquad (11.1)$$

   This can be achieved by choosing $\delta_i = \frac{\delta}{2^i}$, or $\delta_i = \frac{\delta}{i^2} \cdot \frac{6}{\pi^2}$, or, indeed, any series that converges to δ.

4. We will require $m = \sum_{i=1}^{j} m_i$ examples. If we reuse examples, then m is reduced to $\max_{i \leq j} m_i = m_j$.

This gives us an algorithm that can handle classes H with $VCdim = \infty$. However, the number of examples becomes a function of the complexity of c^*, in addition to ϵ and δ.

11.1.2 Further Discussion

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

```
+ + + - + + - - - - - + - - + - - -
|                                 |
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.

11.2 Theoretical Model

11.2.1 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 $VCdim(H_i) = i$). 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 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).

11.2.2 Definitions

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

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

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

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

  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.

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

11.2.3 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, since classes with $VCdim \geq m$ include all the possible classifications of m points, and thus one of their hypotheses must be consistent with the data. (For simplicity, we assume that any set of m points is shattered)

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.

11.3 Fighting Overfitting

11.3.1 Penalty Based Models

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

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

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:

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

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.

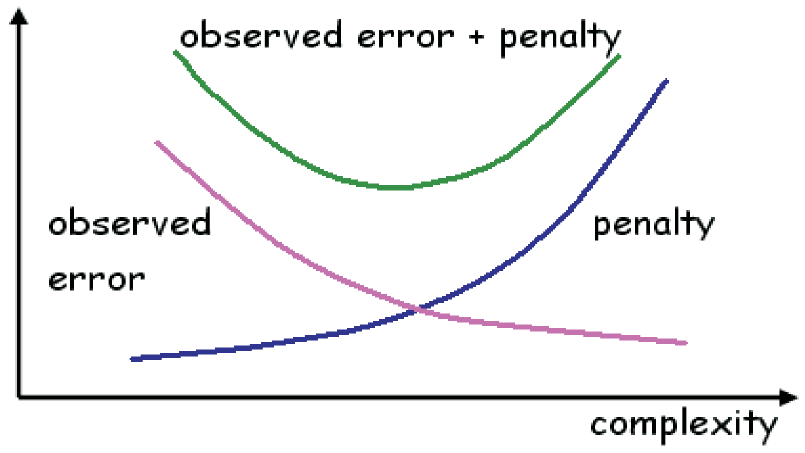

Figure 11.1: Principle of penalty based models.

Figure 11.1 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.

11.3.2 SRM: Structural Risk Minimization

The Model

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

$$Penalty(h) = \sqrt{\frac{2d(h) \cdot \ln(2) + \ln(1/\delta)}{m}} \ , \tag{11.2}$$

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:

$$g^* = arg \min_{g \in H} \left\{ \hat{\epsilon}(g) + Penalty(g) \right\} \tag{11.3}$$

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):

$$h^* = arg \min_{h \in H} \{\epsilon(h)\} \ . \tag{11.4}$$

Let g^* be the hypothesis chosen by *SRM*, i.e. (11.3).

Theorem 11.1 (*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 :*

$$\epsilon(g^*) \leq \epsilon(h^*) + 2 \cdot Penalty(h^*) \tag{11.5}$$

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:

$$\epsilon(h^*) \ \leq \ \epsilon(g^*) \ \leq \ \epsilon(h^*) + 2 \cdot Penalty(h^*) \tag{11.6}$$

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 11.2 *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:*

$$Prob\Big[|\epsilon(h) - \hat{\epsilon}(h)| \geq \lambda\Big] \leq 2e^{-\lambda^2 m} \tag{11.7}$$

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 11.2 directly to bound this probability, because g_i is determined according to the given sample set (and in Claim 11.2 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 11.2 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 :

$$Prob\Big[\exists h \in H_i \mid |\epsilon(h) - \hat{\epsilon}(h)| \geq \lambda_i\Big] \leq 2^i \cdot 2e^{-\lambda_i^2 m}. \tag{11.8}$$

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

$$\delta_i = 2^{i-1} \cdot 2e^{-\lambda_i^2 m}. \tag{11.9}$$

Solving for λ_i we get:

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

$$\lambda_i = \sqrt{\frac{i \cdot \ln(2) + \ln(1/\delta_i)}{m}} \ . \tag{11.10}$$

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 :

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

$$\ge 1 - \sum_i \delta_i = 1 - \delta \tag{11.11}$$

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

$$|\epsilon(h) - \hat{\epsilon}(h)| \le \lambda_i \tag{11.12}$$

for any hypothesis h in $\cup H_i$.

In this case λ_i is as follows:

$$\lambda_i = \sqrt{\frac{i \cdot \ln(2) + \ln(2^i/\delta)}{m}} = \sqrt{\frac{2i \cdot \ln(2) + \ln(1/\delta)}{m}} \tag{11.13}$$

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:

$$\hat{\epsilon}(h^*) \le \epsilon(h^*) + \lambda_i \tag{11.14}$$

$$\epsilon(g^*) - \lambda_j \le \hat{\epsilon}(g^*) \tag{11.15}$$

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 :

$$\hat{\epsilon}(g^*) + P_j \leq \hat{\epsilon}(h^*) + P_i \tag{11.16}$$

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

From the three inequalities (11.14), (11.15) and (11.16) we get:

$$\epsilon(g^*) - \lambda_j + P_j \leq \epsilon(h^*) + \lambda_i + P_i \tag{11.17}$$

and therefore,

$$\epsilon(g^*) \leq \epsilon(h^*) + \lambda_i + P_i + \lambda_j - P_j \ . \tag{11.18}$$

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

$$P_j = \sqrt{\frac{(2j+1)\ln(2) + \ln(1/\delta)}{m}} = \lambda_j \tag{11.19}$$

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

Now we can return to inequality (11.18) and get from (11.19):

$$\epsilon(g^*) \leq \epsilon(h^*) + \lambda_i + P_i + (\lambda_j - P_j) = \epsilon(h^*) + 2 \cdot P_i \tag{11.20}$$

which proves the *SRM* theorem. □

In Practice

In theory, we can consider all classes H_i in $\cup H_i$, even when i goes to infinity, and search for the best hypothesis g^*. However, in practice, we have to stop at some level of complexity, i_{max}; what should this level be? By the time i reaches m, the hypotheses are complex enough to reach an observed error of zero, $\hat{\epsilon}_m = 0$ (we assumed $VCdim(H_i) = i$, and any subset of i points is shattered). Beyond this level there is no need to search, because the observed error will remain 0, and only the complexity penalty will rise (therefore we will never choose those hypotheses, because we have ones with lower complexity penalties).

11.4 Cross Validation

The Model

The *SRM* model tackled the overfitting problem by imposing a complexity penalty on the "price" of a hypothesis, which will steer us to prefer simpler hypotheses rather than complex ones. The model shows that the chosen hypotheses will not be too fitted to the given examples; this will enable the hypotheses to correctly classify also new examples which were not used in the learning process.

The *Cross Validation* method does not change the price of the hypothesis, but leaves it to be the observed error, $\hat{\epsilon}(h)$. To overcome the overfitting problem, the *Cross Validation* splits the given set of examples, S, into two sets, S_1 and S_2. The set S_1, is used as the training sample set in the learning process; this yields for each H_i some hypothesis which is estimated to be the best according to the training set. The examples of the other set, S_2, are then used as a test set, to test the error of the chosen hypotheses on the "new" examples. The chosen hypothesis will be the one with the lowest observed error on the "test" set, S_2. Therefore, *Cross Validation* deals with the overfitting problem by estimating how bad a hypothesis is when learning new examples (how tightly fit it is to the training sample set).

We denote by γ the fraction ($0 < \gamma < 1$) of the original set, S, which is reserved as the test set, S_2. Therefore, if the original set S contains m examples, then the test set S_2 contains γm examples, and the training set S_1 contains $(1-\gamma)m$ examples. Usually γ will be small, because after choosing the best hypotheses according to the training set S_1, the number of candidate hypotheses is reduced and thus we need less examples to choose the best one of the selected hypotheses according to the test set S_2.

We will divide the algorithm into two stages:

1. **Learning from S_1:**
   From each hypotheses class, H_i, we choose the best hypothesis g_i according to the training sample set S_1, i.e., the hypothesis which has the lowest observed error on S_i. Let
   $$g_i = arg\min_{h \in H_i}\{\hat{\epsilon}_1(h)\},$$
   where $\hat{\epsilon}_1(h)$ is the observed error of hypothesis h on the training sample set S_1.

   This will yield a set of hypotheses, G, with one hypothesis for each class H_i.

   Note that, for practicality's sake, we take $1 \le i \le m$; the best hypotheses from classes with complexity greater than or equal to m will have already become completely fitted to the data, and yield $\hat{\epsilon}_1(h) = 0$. We will therefore assume that $|G| = m$. (Alternatively, we will include g_i in G only if $\hat{\epsilon}_1(g_i) < \hat{\epsilon}_1(g_{i-1})$, and this can happen at most m times.)

2. **Testing on S_2**:
   From G we now choose the hypothesis which has the lowest error on the test sample set S_2. Let
   $$g^* = arg \min_{g_i \in G} \{\hat{\epsilon}_2(g_i)\},$$
   where $\hat{\epsilon}_2(h)$ is the observed error of hypothesis h on the test sample set S_2.

Analysis

The analysis will show that *Cross Validation* approximates *SRM.*

Theorem 11.3 *Let $\epsilon_{CV}(m)$ be the error of* Cross Validation *(CV) on m samples, and $\epsilon_A(m)$ be the error of some algorithm A on m samples.*
With probability $1-\delta$,

$$\epsilon_{CV}(m) \leq \epsilon_A((1-\gamma)m) + 2 \cdot \sqrt{\frac{\ln(2m/\delta)}{\gamma m}}$$

Proof:
We'll assume that algorithm A chooses the best hypothesis g_k from some class H_k, by learning from the training sample set S_1. This is the case for the *SRM* algorithm, since as we've seen the complexity penalty is the same for all hypotheses in H_k, thus if the algorithm chooses an hypothesis g_k from class H_k we're guaranteed that g_k is the best hypothesis in H_k (the hypothesis with the lowest observed error on S_1). The penalty only chooses between the best hypotheses in different classes.

Cross Validation, however, may choose a different hypothesis, g_j, because it better classifies the examples from the test set S_2. Note that g_j belongs to H_j, which is a different class from H_k. As shown before, g_k was the best in class H_k, and therefore inserted into G; it is the only member of H_k in G. If g_j is not g_k, then it comes from a different class.

First, we would like to bound the difference between the observed error and the actual error (both on the test set S_2) of any hypothesis g_i in G (the set of best hypothesis from each H_i, as chosen by *Cross Validation*).

We will use Claim 11.2 from the analysis of *SRM*, and state that the probability that a hypothesis g_i in G will have the difference between its observed error on S_2 and its actual error larger than λ, is bounded as follows:

$$Prob\Big[|\epsilon(g_i) - \hat{\epsilon}_2(g_i)| \geq \lambda\Big] \;\leq\; 2 \cdot e^{-\lambda^2 \gamma m} \tag{11.21}$$

where $\hat{\epsilon}_2(g_i)$ is the observed error on the test set S_2 for hypothesis g_i.

Therefore, we can bound the probability that *any* hypothesis in G will have the difference between its actual error and observed error on S_2 larger than λ by:

$$Prob\Big[\exists g \in G \mid |\epsilon(g) - \hat{\epsilon}_2(g)| \geq \lambda\Big] \leq |G| \cdot 2e^{-\lambda^2 \gamma m} \tag{11.22}$$

Since $|G| = m$ we get:

$$Prob\Big[\exists g \in G \mid |\epsilon(g) - \hat{\epsilon}_2(g)| \geq \lambda\Big] \leq m \cdot 2e^{-\lambda^2 \gamma m} \tag{11.23}$$

If we set this upper bound to δ, we get:

$$\delta = m \cdot 2e^{-\lambda^2 \gamma m}$$

Solving for λ leads to:

$$\lambda^2 \gamma m = \ln(2m/\delta)$$

$$\lambda = \sqrt{\frac{\ln(2m/\delta)}{\gamma m}} \tag{11.24}$$

Thus we get :

$$Prob\Big[\exists g \in G \mid |\epsilon(g) - \hat{\epsilon}_2(g)| \geq \lambda\Big] \leq \delta \tag{11.25}$$

Therefore, with probability $1 - \delta$ we have for any g_i in G :

$$|\epsilon(g_i) - \hat{\epsilon}_2(g_i)| \leq \lambda. \tag{11.26}$$

From this we get:

$$\epsilon(g_j) - \lambda \leq \hat{\epsilon}_2(g_j) \tag{11.27}$$

and

$$\hat{\epsilon}_2(g_k) \leq \epsilon(g_k) + \lambda. \tag{11.28}$$

Since *Cross Validation* preferred g_j to g_k, we know:

$$\hat{\epsilon}_2(g_j) \leq \hat{\epsilon}_2(g_k). \tag{11.29}$$

Now, from the three inequalities (11.27), (11.28) and (11.29) we can get:

$$\epsilon(g_j) - \lambda \leq \hat{\epsilon}_2(g_j) \leq \hat{\epsilon}_2(g_k) \leq \epsilon(g_k) + \lambda \tag{11.30}$$

$$\epsilon(g_j) \leq \epsilon(g_k) + 2\lambda. \tag{11.31}$$

We note that:

- $\epsilon(g_j)$ is $\epsilon_{CV}(m)$, the error of *Cross Validation* when learning from m examples.
- $\epsilon(g_k)$ is $\epsilon_A((1-\gamma)m)$, the error of algorithm A when learning from $(1-\gamma)m$ examples.

- $\lambda = \sqrt{\frac{\ln(2m/\delta)}{\gamma m}}$

Thus we get:

$$\epsilon_{CV}(m) \ \leq \ \epsilon_A((1-\gamma)m) + 2 \cdot \sqrt{\frac{\ln(2m/\delta)}{\gamma m}} \ , \tag{11.32}$$

which proves Theorem 11.3. □

In Practice

The analysis showed that *Cross Validation* chooses a hypothesis which, with probability $1-\delta$, is within a very close range (2λ) to the hypothesis chosen by *SRM*.

Also, *Cross Validation* tests with S_2 only a small subset of the hypotheses ($|G| = m$), only the ones that are best (in their classes) in relation to S_1.

However, the analysis compared how *Cross Validation* learns from m examples with how algorithm A learns from $(1-\gamma)m$ examples. This note is important, because there are many cases in which reducing the number of examples might significantly increase the learning error. In such cases, $\epsilon_A((1-\gamma)m)$, A's learning error from $(1-\gamma)m$ examples, is only a weak boundary on $\epsilon_{CV}(m)$, the error of *Cross Validation*'s learning from m examples, and therefore is not very useful. This is typical in cases where there is a "turning-point", a number of examples beyond which learning becomes easy (few errors), and under which learning is difficult (many errors).

For example, consider the problem of learning a vector x of length n, where the examples are vectors v_i (of length n as well) s.t. $\langle v_i, x\rangle = 0$. Say we have $m \geq n$, (i.e we have m examples), but $(1-\gamma)m \leq n$. In this case, an $\epsilon_A((1-\gamma)m)$ is close to 1, but $\epsilon_{CV}(m) = 0$ (assuming that we have n linearly independent vectors v_i).

It is advisable, therefore, to select a γ which will not cause such a difference in the error-level of A (estimate A's learning error as a function of m, and select a good γ).

In practice, we could use *10-fold Cross Validation*. We would randomly choose $9/10m$ examples to be our training set, and test on the remaining $1/10m$ examples. By repeating this process, we could estimate the error rate we'd expect from the eventual hypothesis. After that, assuming we trust the algorithm, we could use all m examples. This would cause a better hypothesis, which we wouldn't be able to test.

11.5 MDL: Minimum Description Length

Consider the typical formulation of a definition in everyday speech. We prefer to define new concepts by comparing them to a simple similar concept, and describing the differences between the two. For example, we would define a car as a wagon with a motor; this is much simpler (shorter) than defining it as a four wheeled land vehicle propelled by the force of an engine. This pattern of concept definition is useful because a great deal of the information is conveyed concisely in the simple concept (here "wagon"), and the list of corrections is manageable (here "with a motor"). In many contexts, the length of the description is a criteria for choosing a description; for example, we may want to transmit the definition, and strive to minimize the length of transmission.

Of course, there is a balance to maintain between the complexity of the concept and the amount of corrections. The more complex the concept described by the hypothesis, the longer the description will be, but the description of the exceptions will be shorter. Conversely, the simpler the concept, the shorter its own description, but the longer the description of the exceptions.

For example, consider Alice wants to send classifications $y_1, ..., y_m$ of samples $x_1, ..., x_m$ to Bob. One way to do this is to send all the classifications. Another way to do this is to send a hypothesis h and a set of corrections $y_{k1}, ..., y_{kl}$.

The *Minimal Description Length* model proposes that when learning a new concept, we do not have to aim for the most accurate hypothesis; we can represent an accurate hypothesis by describing a simple hypothesis, which is similar to the concept but is not accurate, and supplying a list of corrections (examples which should be classified differently than the described hypothesis does). The overall description-length is the sum of the description length of the hypothesis and the corrections.

MDL proposes to find a hypothesis which minimizes the overall description length:

$$g^* = arg \min_{h \in \cup H_i} \left\{ size(corrections(h)) + size(h) \right\} ,$$

where $size(h)$ is the description size of hypothesis h, and $size(corrections)$ is the description size of the examples misclassified by h.

This model can be viewed as a penalty-based model : $size(corrections(h))$ is a function of the observed error of h, and $size(h)$ acts as the complexity-penalty. The chosen hypothesis g^* can be described as:

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

where $f(\hat{\epsilon}(h))$ corresponds to the description size of the observed error of h, and $Penalty(h)$ is the description size of h.

This model is related to the *MAP - Maximum A Posteriori* approach. In *MAP* we choose the hypothesis with the maximum a-posteriori probability given the data. The a-posteriori probability of a hypothesis h given the data D can be computed by Bayes rule as follows :

$$Pr[h|D] = \frac{Pr[D|h] \cdot Pr[h]}{Pr[D]}$$

The chosen hypothesis according to the *MAP* approach is :

$$g^* = arg \max_{h \in \cup H_i} Pr[h|D]$$

Since $Pr[D]$ doesn't depend on h, we can get :

$$g^* = arg \max_{h \in \cup H_i} Pr[D|h] \cdot Pr[h]$$

By taking log, we get :

$$g^* = arg \max_{h \in \cup H_i} log(Pr[D|h]) + log(Pr[h])$$

And by changing the sign :

$$g^* = arg \min_{h \in \cup H_i} log(\frac{1}{Pr[D|h]}) + log(\frac{1}{Pr[h]})$$

The expression $\frac{1}{Pr[D|h]}$ corresponds to the description size of the corrections of h in relation to the data D and the expression $\frac{1}{Pr[h]}$ corresponds to the description size of h (this is a concept taken from Information Theory, highly connected to the Enthropy term). Hence, the problem of finding a minimum description length hypothesis is equivalent to the problem of finding a maximum a-posteriori hypothesis.

11.6 Summary

Model Selection deals with finding a good hypothesis based on a given number of examples.

We introduced the problem of overfitting our hypothesis to the particular examples. This problem arises only when we are limited in the number of examples, and therefore cannot require a large number of examples to sufficiently reduce the probability of error.

We presented two approaches for dealing with the overfitting problem: *Structural Risk Minimization* and *Cross Validation.*

Both of these methods deal with the overfitting problem by repressing the tendency towards overfitted hypotheses, which stems from the desire to minimize the hypothesis' error-level on the given data. *Structural Risk Minimization* does this by imposing a penalty on complexity; *Cross Validation* does it by learning the concept from only a portion of the given data, and then testing the hypothesis on the remaining portion of the data.

We showed that the *Cross Validation* approach approximates the results of the *Structural Risk Minimization.*

We ended by introducing a third method, *Minimum Description Length*, which strives to minimize the description-length of the chosen hypothesis. We showed that this method is basically a variation on the theme of *Structural Risk Minimization.* It values a hypothesis based on its observed error (amount of corrections which have to be described) and simplicity of the hypothesis (its description-length).