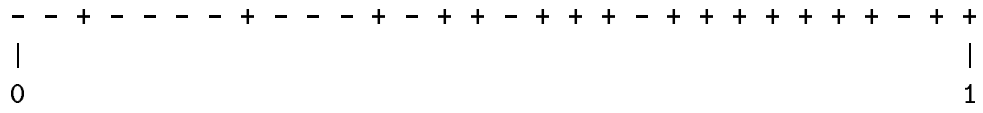
Machine Learning: Foundations		Spring semester, 1996/7
Lecture 9	**Model Selection**	May 7, 1997
Lecturer: Yishay Mansour		*Scribe: Gil Freundlich (June, 1996)*

9.1 Introduction

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

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

To demonstrate the problem, let us assume that the available examples are points in the interval [0,1].



The 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 completly consistent with the data (e.g. surround every positive point with its own tiny positive interval). However, we want to predict correct classifications also for examples other than the original training set of examples. Therefore, by Occam's Razor, we prefer the simplest hypothesis (which would be consistent with the given data). We can make a table of the amount of errors generated by a hypothesis related to the number of intervals in the hypothesis, and choose the simplest hypothesis which yields no errors.

For our example:

Number of Intervals:	0	1	2	3	4	5	6	7	8	9	...
Number of Errors:	30	6	5	4	3	2	1	0	0	0	...

Note that beyond a certain complexity, all hypotheses yield 0 errors. All we have to do now is pick the hypothesis whith the smallest number of intervals which is completely consistent with the observed data (0 errors).

This kind of solution will be accurate for the given examples, but how representative are these examples of the rest of the domain ? In the models learned previously, we solved this by requiring a sufficiently large set of examples; now, however, we are limited to the given examples.

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

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

9.2 Formalization

9.2.1 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 $VCdim(Hi) \leq i$). Clearly,

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

(any hypothesis of a lesser complexity can be included in a hypothesis of a greater complexity).

For the sake of simplicity, we will assume

$$|H_i| = 2^i$$

Let c_t be the the target concept.

In contrast to our previous methods, we now do *not* assume that c_t is included within one of the H_i.

The object of the learning algorithm will be to produce a hypothesis which is "sufficiently close" to c_t (but not necessarily c_t itself).

9.2.2 Definitions

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

$$\epsilon(h) = Prob[h \neq c_t]$$

ϵ_i ϵ_i is 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 $Hi \subseteq H_{i+1}$, therefore $\epsilon_{i+1} \leq \epsilon_i$ (the probability of error decreases as the complexity level increases).

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

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

ϵ^* will not necessarily actually be obtained by any hypothesis h, but it is the lower-bound on any ϵ_i.

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

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

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

9.2.3 The Problem: Over Fitting

Since $\hat{\epsilon}_i$ is observed on the given data, beyond some complexity-level i, 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 these high levels of complexity, the hypotheses (with lowest levels of errors on the given data) are too fitted to the given data, excluding anything but the given data. This phenomenon is called "overfitting".

In our case, we can not require a sufficently 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. The overfitted hypothesis might turn out quite far from the true concept.

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

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

NOTE: The *arg* min function returns the *argument* for which the minimum value has been calculated (rather than the minimum value itself).

However, using this simplistic approach for choosing g will cause us to prefer overfitted hypotheses, because they yield the lowest $\hat{\epsilon}(h)$, 0.

9.3 Fighting Overfitting

9.3.1 SRM: Structural Risk Minimization

The Model

One way to overcome the Overfitting problem is to impose a complexity tax (penalty) on the price of the chosen hypothesis; we will then try to minimize the price of the chosen hypothesis.

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

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

where $Penalty$ reflects something about the complexity of g.

This penalty 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.

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

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

The *SRM* (*Structural Risk Minimization*) model uses the following as the $Penalty$

$$Penalty(h) = \sqrt{\frac{2d(h)}{m}}$$

where m is the number of examples.

The hypothesis g^* chosen by the *SRM* model will therefore be:

$$g^* = arg \min_{g \in \cup H_i} \left\{ \hat{\epsilon}(g) + \sqrt{\frac{2d(g)}{m}} \right\}$$

Notice that the penalty is calculated based on $d(g)$, which is the (first) class in which g is found; therefore, the penalty the same for all hypothises with the same complexitgy ($d(h)$).

Analysis

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

$$h^* = arg \min_{h \in \cup H_i} \{\epsilon(h)\}$$

Let g^* be the hypothesis chosen by *SRM*, i.e.:

$$g^* = arg \min_{g \in \cup H_i} \left\{ \hat{\epsilon}(g) + \sqrt{\frac{2 * d(g)}{m}} \right\}$$

Theorem 9.1 *The actual error of g^* is smaller than or equal to the actual error of h^* plus (a constant times) the* SRM *complexity-penalty of h^*; more specifically:*

$$\epsilon(g^*) \leq \epsilon(h^*) + 2 * \sqrt{\frac{2d(h^*)}{m}}$$

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 bound on both sides:

$$\epsilon(h^*) \ \leq \ \epsilon(g^*) \ \leq \ \epsilon(h^*) + 2 * \sqrt{\frac{2d(h^*)}{m}}$$

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.

Claim 9.2 *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:*

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

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

We will use this inequality in our analysis.

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

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

Let us estimate the probability of difference between the actual error and the observed error of gi:

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

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

This is smaller than the probability that *any* hypothesis in H_i will show a big difference between the actual error and observed error:

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

which, by applying the inequality of our claim:

$$\leq |H_i| * 2e^{-\lambda_i^2 m}$$

Recall that we assumed for simplicity that $|H_i| = 2^i$, so we get that for any hypothesis h:

$$Prob\Big[|\epsilon(h) - \hat{\epsilon}(h)| \geq \lambda_i\Big] \leq 2^i * 2e^{-\lambda_i^2 m}$$

Let us name this upper bound δ_i (the probability of error for any hypothesis in H_i):

$$\delta_i = 2^i * 2e^{-\lambda_i^2 m}$$

Solving for λ_i we get:

$$\lambda_i^2 m = \ln\left(\frac{2^{i+1}}{\delta_i}\right)$$

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

Let us define an overall δ, signifying the probability of error for any hypothesis from any class in $\cup H_i$, as the sum of all δ_i:

$$\delta = \sum_i [\delta_i] = \sum_i [2^i * 2e^{-\lambda_i^2 m}] = 2e^{-\lambda_i^2 m} * \sum_i [2^i]$$

Now we can reformulate the above inequality, and say that for any hypothesis h in $\cup H_i$:

$$Prob\Big[|\epsilon(h) - \hat{\epsilon}(h)| \geq \lambda_i\Big] \leq \delta_i \leq \delta$$

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

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

for any hypothesis h in $\cup H_i$. Among other hypotheses, this is true also for h^* and g^*, which leads to the following:

$$\hat{\epsilon}(h^*) \leq \epsilon(h^*) + \lambda_i$$

$$\epsilon(g^*) - \lambda_j \leq \hat{\epsilon}(g^*)$$

where,

- $i = d(h^*)$ — i.e. h^* was found in class H_i
- $j = d(g^*)$ — i.e. g^* was found in class H_j
- P_i, P_j — are the *SRM* complexity-penalties for h^* and g^*, respectively.

Additionally,

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

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

From the last three inequalities we get:

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

and therefore,

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

We will set the penalty-value for P_j such that

$$P_j \geq \lambda_j$$

(the penalty is greater than the actual divergence), which would yield

$$\lambda_j - P_j \leq 0$$

Recall that

$$\lambda_j = \sqrt{\frac{(j+1)ln(2) + ln(1/\delta_j)}{m}}$$

We will, therefore, set the complexity-penalty for a hypothesis to be the *SRM* complexity-penalty. In the case of P_j, this will result:

$$P_j = \sqrt{\frac{2d(g^*)}{m}} = \sqrt{\frac{2j}{m}} \geq \sqrt{\frac{(j+1)ln(2) + ln(1/\delta_j)}{m}}$$

So, we have

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

(because $\lambda_j - P_j \leq 0$).

Since for P_i, too, the relation holds:

$$\lambda_i \leq P_i$$

we get

$$\epsilon(g^*) \leq \epsilon(h^*) + \lambda_i + P_i \leq \epsilon(h^*) + P_i + P_i = \epsilon(h^*) + 2 * P_i$$

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 ? The assumption is that by the time i reaches m (or some constant times m, e.g. $2m$), the hypotheses are complex enough to reach an observed error of zero, $\hat{\epsilon}_m = 0$. 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 penatlies).

9.3.2 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 set of given examples, S, into two sets, S_1 and S_2. Only one of the sets, S_1, is used as training samples in the learning process; this yields some hypothesis which is estimated to be the best. The examples of the other set, S_2, are then used as a test set, to test the error of the chosen hypothesis 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.

We will devide the algorithm into two stages:

1. **Learning from** S_1:
   From each hypotheses class, H_i, we will choose the best hypothesis, g_i, which will have the lowest observed error on the training sample-set S_1.

   $$g_i = arg \min_{h \in H_i} \{\hat{\epsilon}_1(h)\}$$

   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 $1 \leq i \leq m$; the best hypotheses from classes with complexity greater than or equal to m (or some constant times m, e.g. $2m$) will have already become completely fitted to the data, and yield $\hat{\epsilon}_1(h) = 0$. We will therefore assume that $|G| = m$.

2. **Testing on S_2**:
   From G we now choose the hypothesis which has the lowest error on the test sample set S_2.

$$g^* = arg \min_{g_i \in G} \{\hat{\epsilon}_2(g_i)\}$$

   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*.

Claim 9.3 *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.*
In probability $1 - \delta$,

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

Algorithm A chooses a hypothesis g_k ($g_k \in H_k$), by learning from the training sample set S_1. Note that the penalty depends only on the complexity, and the complexity is the same for all hypotheses in H_k; this means that the g_k is the best in class H_k even without considering the penalty (the penalty only chooses between the best hypotheses in different classes).

Cross Validation, however, chose 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 than 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 our claim from the analysis of *SRM*, and state that the probability that a hypothesis g_i in G shows a big difference between its observed error on S_2 and its actual error on S_2 is bound from above, as follows:

$$Prob\Big[|\epsilon_2(g_i) - \hat{\epsilon}_2(g_i)| \geq \lambda\Big] \leq 2 * e^{-\lambda^2 \gamma m}$$

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

This is smaller than the probability that *any* hypothesis in G will show a big difference between the actual error and observed error:

$$\leq Prob\Big[\exists g \in G \; and \; |\epsilon_2(g) - \hat{\epsilon}_2(g)| \geq \lambda\Big]$$

which, by applying the previous inequality:

$$\leq |G| * 2e^{-\lambda^2 m}$$

Recall that we are assuming that $|G| = m$; therefore we get:

$$Prob\Big[|\epsilon_2(g_i) - \hat{\epsilon}_2(g_i)| \geq \lambda\Big] \leq m * 2e^{-\lambda^2 \gamma m}$$

Let us define δ, our probability of error, as this upper bound:

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

Solving for λ leads to:

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

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

Since for any g_i in G we have:

$$Prob\Big[|\epsilon_2(g_i) - \hat{\epsilon}_2(g_i)| \geq \lambda\Big] \leq \delta$$

we therefore established that for any g_i in G:

$$Prob\Big[|\epsilon_2(g_i) - \hat{\epsilon}_2(g_i)| \leq \lambda\Big] \geq (1 - \delta)$$

From this we get:

$$\epsilon_2(g_j) - \lambda \leq \hat{\epsilon}_2(g_j)$$

and

$$\hat{\epsilon}_2(g_k) \leq \epsilon_2(g_k) + \lambda$$

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

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

From the last three inequalities we get:

$$\epsilon_2(g_j) - \lambda \leq \hat{\epsilon}_2(g_j) \leq \hat{\epsilon}_2(g_k) \leq \epsilon_2(g_k) + \lambda$$

$$\epsilon_2(g_j) \leq \epsilon_2(g_k) + 2\lambda$$

We note that:

- $\epsilon_2(g_j)$ is $\epsilon_{CV}(m)$, the error of *Cross Validation* when learning from m examples.
- $\epsilon_2(g_k)$ is $e_A((1-\gamma)m)$, the error of algorithm A when learning from $(1-\gamma)m$ examples.
- $\lambda = \sqrt{\frac{ln(2m/\delta)}{\gamma m}}$

Thus we get:

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

In Practice

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

Also, *Cross Validation* tests with S_2 only a small subset of the hypotheses ($|G|$, which is assumed here to equal m), only the ones that are best (in their class) for S_1.

Still, *Cross Validation* has an "error" margine of 2λ from *SRM*.

Additionally, and more importantly, 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 a case, the fact that $e_{CV}(m)$, the error of *Cross Validation*'s learning from m examples, is bounded by $e_A((1-\gamma)m)$, A's learning error from $(1-\gamma)m$ examples; this upper bound may now be a large number, and therefore 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).

It is adviseable, 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).

9.3.3 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: "an X is a Y which [differences between X's and Y's]". 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 usefull because a great deal of the information is convayed 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 trnasmission.

Of course, there is a balance to maintain between the complexity of the concdept 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.

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 hyhpothesis, which is similar to the concept but 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:

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

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

Note that this model is basically a *Structural Risk Minimization* 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:

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

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.

9.4 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 given 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 proglem: *Structural Risk Minimzation* and *Cross Validation.*

Both these methods dealt with the Overfitting problem by repressing the tendancy towards overfitted hypotheses, a tendency 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 given data.

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

We then introduced 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, too, values a hypothesis based on its observed error (amount of corrections which have to be described) and simplicity of the hypothesis (its description-length).