2.997 Decision-Making in Large-Scale Systems March 10
MIT, Spring 2004 Handout #14

Lecture Note 11

1 Complexity and Model Selection

In this lecture, we will consider the problem of *supervised learning*. The setup is as follows. We have pairs (x, y), distributed according to a joint distribution $P(x, y)$. We would like to describe the relationship between x and y through some function $\hat{f}$ chosen from a set of available functions $\mathcal{C}$, so that $y \approx \hat{f}(x)$. Ideally, we would choose $\hat{f}$ by solving

$$\hat{f} = \text{argmin}_{g \in \mathcal{C}} \mathrm{E}\left[(y - \hat{f})^2 | x, y \sim P\right] \qquad \text{(test error)}$$

However, we will assume that the distribution P is not known, but rather, we only have access to samples (x_i, y_i). Intuitively, we may try to solve

$$\min_{\hat{f}} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{f}(x_i)\right)^2 \qquad \text{(training error)}$$

instead. It also seems that, the richer the class $\mathcal{C}$ is, the better the chance to correctly describe the relationship between x and y. In this lecture, we will show that this is not the case, and the appropriate complexity of $\mathcal{C}$ and the selection of a model for describing how x and y related must be guided by how much data is actually available. This issue is illustrated in the following example.

2 Example

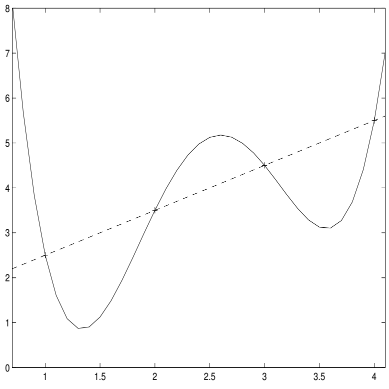
Consider fitting the following data by a polynomial of finite degree:

x	1	2	3	4
y	2.5	3.5	4.5	5.5

Among several others, the following polynomials fit the data perfectly:

$$\hat{y} = x + 1.5$$

$$\hat{y} = 2x^4 - 20x^3 + 70x^2 - 99x + 49.5$$



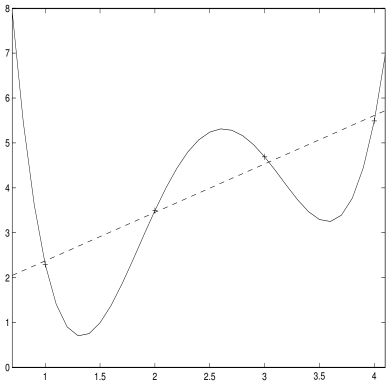
Which polynomial should we choose?

Now consider the following (possibly noisy) data:

x	1	2	3	4
y	2.3	3.5	4.7	5.5

Fitting the data with a first-degree polynomial yields $\hat{y} = 1.03x + 1.3$; fitting it with a fourth-degree polynomial yields (among others) $\hat{y} = 2x^4 - 20.0667x^3 + 70.4x^2 - 99.5333x + 49.5$.

Which polynomial should we choose?



3 Training error vs. test error.

It seems intuitive in the previous example that a line may be the best description for the relationship between x and y, even though a polynomial of degree 3 describes the data perfectly in both cases and no linear function is able to describe the data perfectly in the second case. Is the intuition correct, and if so, how can we decide on an appropriate representation, if relying solely on the training error does not seem completely reasonable?

The essence of the problem is as follows. Ultimately, what we are interested in is the ability of our fitted curve to predict *future* data, rather than simply explaining the observed data. In other words, we would like to choose a predictor that minimizes the expected error $|y(x) - \hat{y}(x)|$ over all possible x. We call this the *test error*. The average error over the data set is called the *training error*.

We will show that training error and test error can be related through a measure of the *complexity* of the class of predictors being considered. Appropriate choice of a predictor will then be shown to require balancing the training error and the complexity of the predictors being considered. Their relationship is described in Fig. 1, where we plot test and training errors versus complexity of the predictor class $\mathcal{C}$ when the number of samples is fixed. The main difficulty is that, as indicated in Fig. 1, there exists a tradeoff between the complexity and the errors, i.e., training error and the test error; while the approximation error over the sampled points goes to zero as we consider richer approximation classes, the same is not true for the test error, which we are ultimately interested in minimizing. This is due to the fact that, with only finite amount of data and noisy observations y_i, if the class $\mathcal{C}$ is too rich we may run into overfitting — fitting the noise in the observations, rather than the underlying structure linking x and y. This leads to poor generalization from the training error to the test error.

We will investigate how bounds on the test error based on the training error and the complexity of $\mathcal{C}$ may be developed for the special case of *classification problems* — i.e., problems where $y \in \{-1, +1\}$, which may be seen as an indicating whether x_i belongs in a certain set or not. The ideas and results easily generalize to general function approximation.

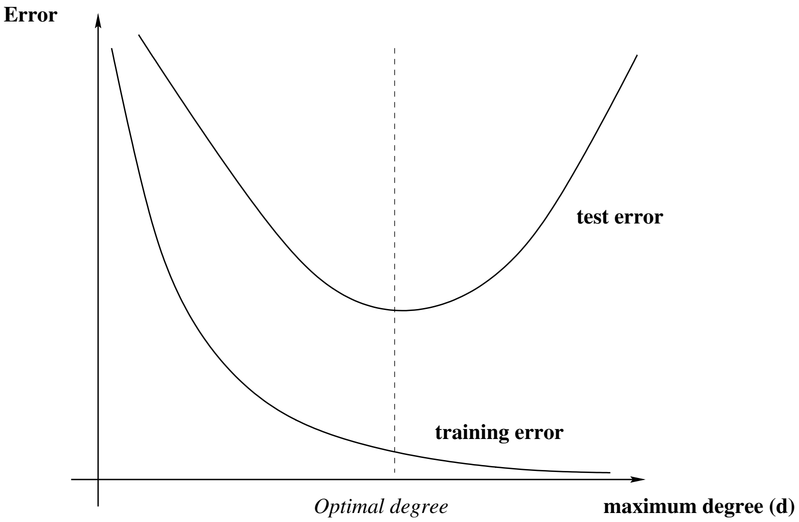


Figure 1: Error vs. degree of approximation function

3.1 Classification with a finite number of classifiers

Suppose that, given n samples $(x_i, y_i), i = 1, \ldots, n$, we need to choose a *classifier* h_i from a finite set of classifiers $f_1, \ldots, f_d$.

Define

$$\begin{aligned} \epsilon(k) &= \mathrm{E}[|y - f_k(x)|] \\ \hat{\epsilon}_n(k) &= \frac{1}{n}\sum_{i=1}^{n} |y_i - f_k(x_i)|. \end{aligned}$$

In words, $\epsilon(k)$ is the test error associated with classifier f_k, and $\hat{\epsilon}_n(k)$ is a random variable representing the training error associated with classifier f_k over the samples $(x_i, y_i), i = 1, \ldots, n$. As described before, we would like to find $k^* = \arg\min_k \epsilon(k)$, but cannot compute directly. Let us consider using instead

$$\hat{k} = \arg\min_k \hat{\epsilon}_n(k)$$

We are interested in the following question: How does the test error $\epsilon(\hat{k})$ compare to the optimal error $\epsilon(k^*)$?

Suppose that

$$|\hat{\epsilon}_n(k) - \epsilon(k)| \leq \epsilon, \quad \forall k, \tag{1}$$

for some $\epsilon > 0$. Then we have

$$\begin{aligned} \epsilon(k) &\leq \hat{\epsilon}_n(k) + \epsilon \\ \text{test error} &\leq \text{training error } + \epsilon, \end{aligned}$$

and

$$\begin{aligned} \epsilon(\hat{k}) &\leq \hat{\epsilon}_n(\hat{k}) + \epsilon \\ &\leq \hat{\epsilon}_n(k^*) + \epsilon \\ &\leq \epsilon(k^*) + 2\epsilon. \end{aligned}$$

In words, if the training error is close to the test error for all classifiers f_k, then using $\hat{k}$ instead of k^* is near-optimal. But can we expect (1) to hold?

Observe that $|y_i - f_k(x_i)|$ are i.i.d. Bernoulli random variables. From the strong law of large numbers, we must have

$$\hat{\epsilon}_n(k) \to \epsilon(k) \quad \text{w.p. } 1.$$

This means that, if there are sufficient samples, (1) should be true. Having only finitely many samples, we face two questions:

(1) How many samples are needed before we have high confidence that $\hat{\epsilon}_n(k)$ is close to $\epsilon(k)$?

(2) Can we show that $\hat{\epsilon}_n(k)$ approaches $\epsilon(k)$ equally fast for all $f_k \in \mathcal{C}$?

The first question is resolved by the Chernoff bound: For i.i.d. Bernoulli random variables x_i, $i = 1, \ldots, n$, we have

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} x_i - \mathrm{E}x_1\right| > \epsilon\right) \leq 2\exp(-2n\epsilon^2)$$

Moreover, since there are only finitely many functions in $\mathcal{C}$, uniform convergence of $\hat{\epsilon}_n(k)$ to $\epsilon(k)$ follows immediately:

$$\begin{aligned} P(\exists k : |\hat{\epsilon}(k) - \epsilon(k)| > \epsilon) &= P\left(\cup_k \{|\hat{\epsilon}(k) - \epsilon(k)| > \epsilon\}\right) \\ &\leq \sum_{k=1}^{d} P\left(\{|\hat{\epsilon}(k) - \epsilon(k)| > \epsilon\}\right) \\ &\leq 2d\exp(-2n\epsilon^2). \end{aligned}$$

Therefore we have the following theorem.

Theorem 1 *With probability at least $1 - \delta$, the training set (x_i, y_i), $i = 1, \ldots, n$, will be such that*

$$\textit{test error} \leq \textit{training error} + \epsilon(d, n, \delta)$$

where

$$\epsilon(d, n, \delta) = \sqrt{\frac{1}{2n}\left(\log 2d + \log\frac{1}{\delta}\right)}.$$

4 Measures of Complexity

In Theorem 1, the error $\epsilon(d, n, \delta)$ is on the order of $\sqrt{\log d}$. In other words, the more classifiers are under consideration, the larger the bound on the difference between the testing and training errors, and the difference grows as a function of $\log d$. It follows that, for our purposes, $\log d$ captures the complexity of $\mathcal{C}$. It turns out that, in the case where there are infinitely many classifiers to choose from, i.e., $m = \infty$, a different notion of complexity leads to a bound similar to that in Theorem 1

How can we characterize complexity? There are several intuitive choices, such as the degrees of freedom associated with functions in $\mathcal{S}$ or the length required to describe any function in that set (description length). In certain cases, these notions can be shown to give rise to bounds relating the test error to the training error. In this class, we will consider a measure of complexity that holds more generally — the Vapnik-Chernovenkis (VC) dimension.

4.1 VC dimension

The VC dimension is a property of a class $\mathcal{C}$ of functions — i.e., for each set $\mathcal{C}$, we have an associated measure of complexity, $d_{VC}(\mathcal{C})$. d_{VC} captures how much variability there is between different functions in $\mathcal{C}$. The underlying idea is as follows. Take n points $x_1, \ldots, x_n$, and consider binary vectors in $\{-1, +1\}^n$ formed by applying a function $f \in \mathcal{C}$ to (x_i). How many different vectors can we come up with? In other words, consider the following matrix:

$$\begin{bmatrix} f_1(x_1) & f_1(x_2) & \ldots & f_1(x_n) \\ f_2(x_1) & f_2(x_2) & \ldots & f_2(x_n) \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

where $f_i \in \mathcal{C}$. How many distinct rows can this matrix have? This discussion leads to the notion of *shattering* and to the definition of the VC dimension.

Definition 1 (Shattering) *A set of points $x_1, \ldots, x_n$ is shattered by a class $\mathcal{C}$ of classifiers if for any assignment of labels in $\{-1, 1\}$, there is $f \in \mathcal{C}$ such that $f_(x_i) = y_i, \forall i$.*

Definition 2 *VC dimension of $\mathcal{C}$ is the cardinality of the largest set it can shatter.*

Example 1 *Consider $|\mathcal{C}| = d$. Suppose $x_1, x_2, dots, x_n$ is shattered by $\mathcal{C}$. We need $d \geq 2^n$ and thus $n \leq \log d$. This means that $d_{VC}(\mathcal{C}) \leq \log d$.*

Example 2

Consider $\mathcal{C} = \{\text{hyperplanes in } \Re^2\}$, Any two points in $\Re^2$ can be shattered. Hence, $d_{VC}(\mathcal{C}) \geq 2$. Consider any three points in $\Re^2$, $\mathcal{C}$ can shatter these three points. Hence $d_{VC}(\mathcal{C}) \geq 3$. Since $\mathcal{C}$ cannot shatter any four points in $\Re^2$, hence $d_{VC}(\mathcal{C}) \geq 3$. It follows that $d_{VC}(\mathcal{C}) = 3$. Moreover, it can be shown that, if $\mathcal{C} = \{\text{hyperplanes in } \Re^n\}$, then $d_{VC}(\mathcal{C}) = n + 1$.

Example 3 *If $\mathcal{C}$ is the set of all convex sets in $\Re^2$, we can show that $d_{VC}(\mathcal{C}) = \infty$.*

It turns out that the VC dimension provides a generalization of the results from the previous section, for finite sets of classifiers, to general classes of classifiers:

Theorem 2 *With probability at least $1 - \delta$ over the choice of sample points (x_i, y_i), $i = 1, \ldots, n$, we have*

$$\epsilon(f) \leq \hat{\epsilon}_n(f) + \epsilon(n, d_{VC}(\mathcal{C}), \delta), \quad \forall f \in \mathcal{C},$$

where

$$\epsilon(n, d_{VC}(\mathcal{C}), \delta) = \sqrt{\frac{d_{VC}\left(\log(\frac{2n}{d_{VC}}) + 1\right) + \log(\frac{1}{4\delta})}{n}}$$

Moreover, a suitable extension to bounded real-valued functions, as opposed to functions taking value in $\{-1, +1\}$, can also be obtained. It is called the *Pollard dimension* and gives rise to results analogous to Theorem 1 and 2.

Definition 3 *Pollard dimension of $\mathcal{C} = \{f_\alpha(x)\} = \max_s d_{VC}\left(\{I(f_\alpha(x) > s(x))\}\right)$*

5 Structural Risk Minimization

Based on the previous results, we may consider the following approach to selecting a class of functions $\mathcal{C}$ whose complexity is appropriate for the number of samples available. Suppose that we have several classes $\mathcal{C}_1 \subsetneqq \mathcal{C}_2 \subsetneqq \dots \mathcal{C}_p$. Note that complexity increases from $\mathcal{C}_1$ to $\mathcal{C}_p$. We have classifiers $f_1, f_2, \dots, f_p$ which minimizes the training error $\hat{\epsilon}_n(f_i)$ within each class. Then, given a confidence level δ, we may found upper bounds on the test error $\epsilon(f_i)$ associated with each classifier:

$$\epsilon(f_i) \leq \hat{\epsilon}_n(f_i) + \epsilon(d_{VC}, n, \delta),$$

with probability at least $1 - \delta$, and we can choose the classifier f_i that minimizes the above upper bound. This approach is called *structural risk minimization.*

There are two difficulties associated with structural risk minimization: first, the upper bound provided by Theorems 1 and 2 may be loose; second, it may be difficult to determine the VC dimension of a given class of classifiers, and rough estimates or upper bounds may have to be used instead. Still, this may be a reasonable approach, if we have a limited amount of data. If we have a lot of data, an alternative approach is as follows. We can split the data in three sets: a training set, a validation set and a test set. We can use the training set to find the classifier f_i within each class $\mathcal{C}_i$ that minimizes the training error; use the validation set to estimate the test error of each selected classifier f_i, and choose the classifier $\hat{f}$ from $f_1, \dots, f_p$ with the smallest estimate; and finally, use the test set to generate an estimate of the test error associated with $\hat{f}$.