**ECE901 Spring 2007 Statistical Learning Theory** **Instructor: R. Nowak**

# Lecture 3: Introduction to Complexity Regularization

We ended the previous lecture with a brief discussion of overfitting. Recall that, given a set of $n$ data points, $D_n$, and a space of functions (or *models*) $\mathcal{F}$, our goal in solving the learning from data problem is to choose a function $\widehat{f}_n \in \mathcal{F}$ which minimizes the expected risk $E\left[R(\widehat{f}_n)\right]$, where the expectation is being taken over the distribution $P_{XY}$ on the data points $D_n$. One approach to avoiding overfitting is to restrict $\mathcal{F}$ to some subset of all measurable function. To gauge the performance of a given $f$ in this case, we examine the difference between the expected risk of $f$ and the Bayes' risk (called the *excess risk*).

$$E\left[R(\widehat{f}_n)\right] - R^* = \underbrace{\left(E[R(\widehat{f}_n)] - \inf_{f\in\mathcal{F}} R(f)\right)}_{\text{estimation error}} + \underbrace{\left(\inf_{f\in\mathcal{F}} R(f) - R^*\right)}_{\text{approximation error}}$$

The *approximation error* term quantifies the performance hit incurred by imposing restrictions on $\mathcal{F}$. The *estimation error* term is due to the randomness of the training data, and it expresses how well the chosen function $\widehat{f}_n$ will perform in relation to the best possible $f$ in the class $\mathcal{F}$. This decomposition into stochastic and approximation errors is similar to the bias-variance tradeoff which arises in classical estimation theory. The approximation error is like a bias squared term, and the estimation error is like a variance term. By allowing the space $\mathcal{F}$ to be large[1] we can make the approximation error as small as we want at the cost of incurring a large estimation error. On the other hand, if $\mathcal{F}$ is very small then the approximation error will be large, but wthe estimation error may be very small. This tradeoff is illustrated in Figure 1.

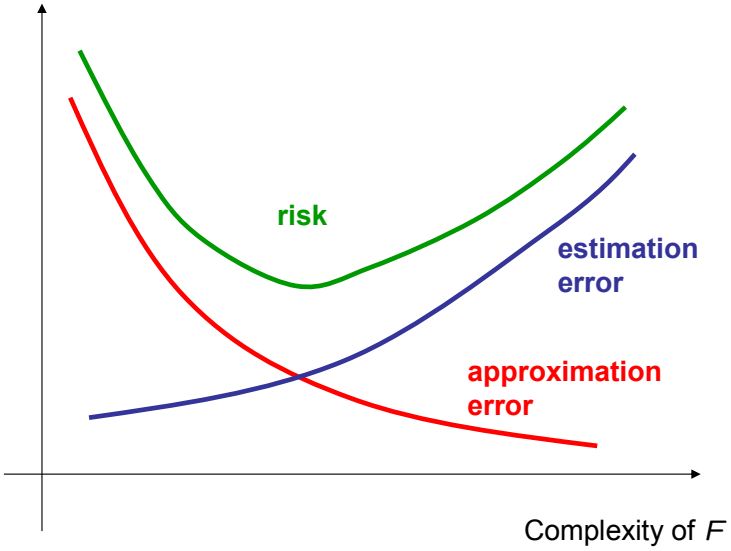



Figure 1: Illustration of tradeoff between estimation and approximation errors as a function of the size (complexity) of the $\mathcal{F}$.

[1] When we say $\mathcal{F}$ is large, we mean that $|\mathcal{F}|$, the number of elements in $\mathcal{F}$, is large.

Why is this the case? We do not know the true distribution $P_{XY}$ on the data, so instead of minimizing the expected risk of we design a predictor by minimizing the empirical risk:

$$\begin{aligned} \widehat{f}_n &= \arg\min_{f\in\mathcal{F}} \widehat{R}_n(f), \\ \widehat{R}_n(f) &= \frac{1}{n}\sum_{i=1}^{n} \ell(f(X_i), Y_i). \end{aligned}$$

If $\mathcal{F}$ is very large then $\widehat{R}_n(f)$ can be made arbitrarily small and the resulting $\widehat{f}_n$ can "overfit" to the data since $\widehat{R}_n(f)$ is not be a good estimator of the true risk $R(\widehat{f}_n)$.

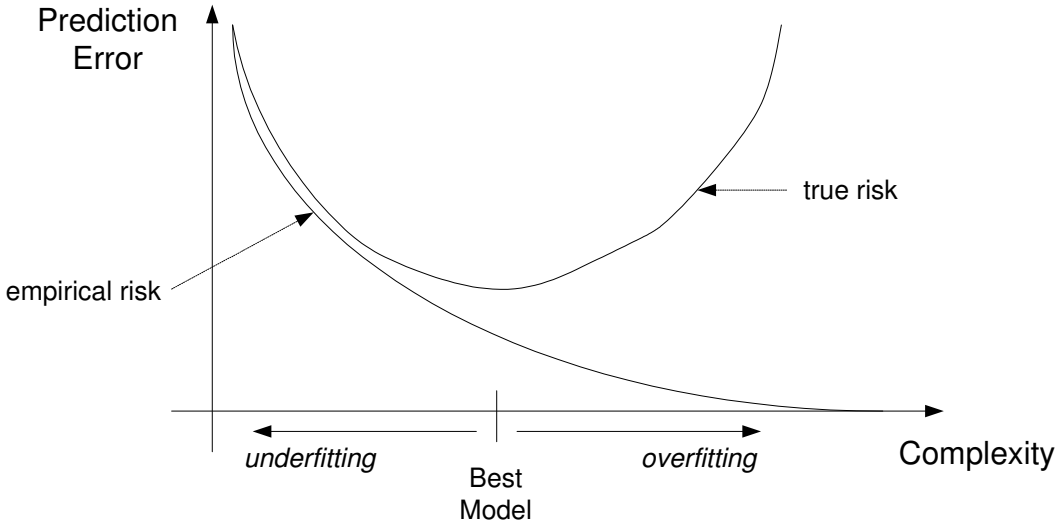



Figure 2: Illustration of empirical risk and the problem of overfitting to the data.

The behavior of the true and empirical risks, as a function of the size (or *complexity*) of the space $\mathcal{F}$, is illustrated in Figure 2. Unfortunately, we can't easily determine whether we are over or underfitting just by looking at the empirical risk.

# 1 Strategies To Avoid Overfitting

Picking

$$\widehat{f}_n = \arg\min_{f\in\mathcal{F}} \widehat{R}_n(f)$$

is problematic if $\mathcal{F}$ is large. We will examine two general approaches to dealing with this problem:

1. Restrict the size or dimension of $\mathcal{F}$ (e.g., restrict $\mathcal{F}$ to the set of all lines, or polynomials with maximum degree $d$). This effectively places an upper bound on the estimation error, but in general it also places a lower bound on the approximation error.

2. Modify the empirical risk criterion to include an extra cost associated with each model (e.g., higher cost for more complex models):

$$\widehat{f}_n = \arg\min_{f\in\mathcal{F}} \left\{ \widehat{R}_n(f) + C(f) \right\}.$$

The cost is designed to mimic the behavior of the estimation error so that the model selection procedure avoids models with a estimation error. Roughly this can be interpreted as trying to balance the tradeoff illustrated in Figure 1. Procedures of this type are often called complexity penalization methods.

**Example 1** *Revisit the polynomial regression example (Lecture 2, Ex. 4), and incorporate a penalty term $C(f)$ which is proportional to the degree of $f$, or the derivative of $f$. In essence, this approach penalizes for functions which are too "wiggly", with the intuition being that the true function is probably smooth so a function which is very wiggly will overfit the data.*

How do we decide how to restrict or penalize the empirical risk minimization process? Approaches which have appeared in the literature include the following.

## 1.1 Method of Sieves (Grenander, 1981)

Perhaps the simplest approach is to try to limit the size of $\mathcal{F}$ in a way that depends on the number of training data $n$. The more data we have, the more complex the space of models we can entertain. Let the class of candidate functions grow with $n$. That is, take

$$F_1, \mathcal{F}_2, \ldots, \mathcal{F}_n, \ldots$$

where $|\mathcal{F}_i|$ grows as $i \to \infty$. In other words, consider a sequence of spaces with increasing complexity or degrees of freedom depending on the number of training data samples, $n$.

Given samples $\{X_i, Y_i\}_{i=1}^n$ i.i.d. distributed according to $P_{XY}$, select $f \in \mathcal{F}_n$ to minimize the empirical risk

$$\widehat{f}_n = \arg \min_{f \in \mathcal{F}_n} \widehat{R}_n(f).$$

In the next lecture we will consider an example using the method of sieves. The basic idea is to design the sequence of model spaces in such a way that the excess risk decays to zero as $n \to \infty$. This sort of idea has been around for decades, but Grenander's method of sieves is often cited as a nice formalization of the idea: Grenander (1981), *Abstract Inference*, Wiley, New York.

## 1.2 Complexity Penalization Methods

### 1.2.1 Bayesian Methods (Bayes, 1764)

In certain cases, the empirical risk happens to be a (log) likelihood function, and one can then interpret the cost $C(f)$ as reflecting prior knowledge about which models are more or less likely. In this case, $e^{-C(f)}$ is like a prior probability distribution on the space $\mathcal{F}$. The cost $C(f)$ is large if $f$ is highly improbable, and $C(f)$ is small if $f$ is highly probable.

Alternatively, if we restrict $\mathcal{F}$ to be small, and denote the space of all measurable functions as $\mathbb{F} = \mathcal{F} \cup \mathcal{F}^c$, then it is essentially as if we have placed a uniform prior over all functions in $\mathcal{F}$, and zero prior probability on the functions in $\mathcal{F}^c$.

### 1.2.2 Description Length Methods (Rissanen, 1978)

Description length methods represent each $f$ with a string of bits. More complicated functions require more bits to represent. Accordingly, we can then set the cost $c(f)$ proportional to the number of bits needed to describe $f$ (the *description length*). This results in what is known as the minimum description length (MDL) approach where the minimum description length is given by

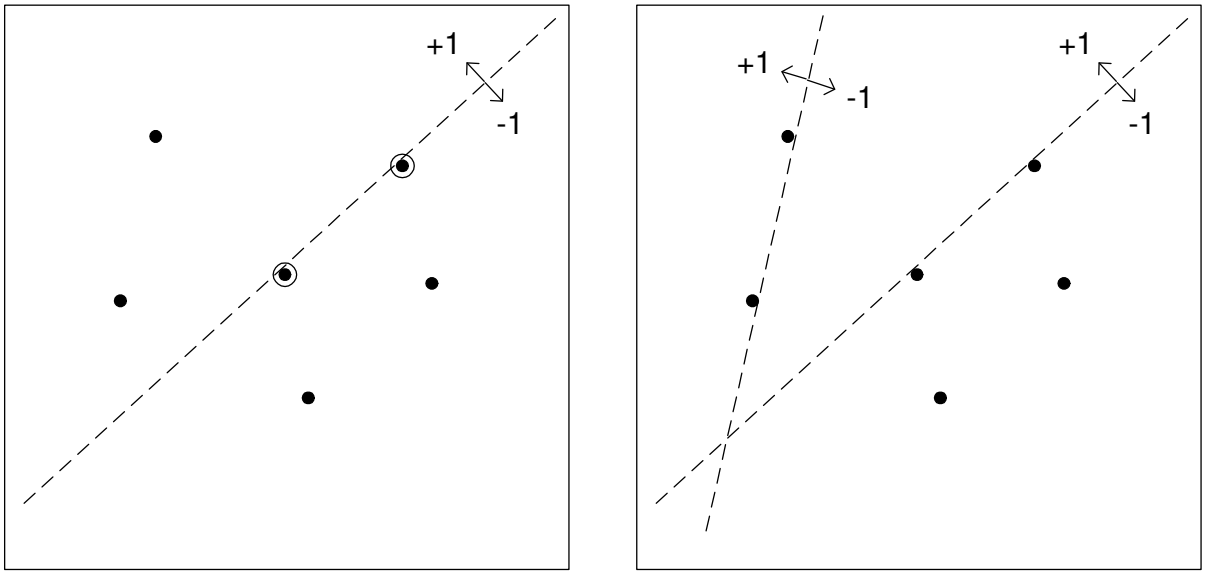
$$\min_{f \in \mathcal{F}} \left\{ \widehat{R}_n(f) + C(f) \right\}.$$

In the Bayesian setting, $p(f) \propto e^{-C(f)}$ can be interpreted as a prior probability density on $\mathcal{F}$, with more complex models being less probable and simpler models being more probable. In that sense, both the Bayesian and MDL approaches have a similar spirit.

### 1.2.3 Vapnik-Cervonenkis Dimension (Vapnik & Cervonenkis, 1971)

The Vapnik-Cervonenkis (VC) dimension measures the complexity of a class $\mathcal{F}$ relative to a random sample of $n$ training data. For example, take $\mathcal{F}$ to be all linear classifiers in 2-dimensional feature space. Clearly, the space of linear classifiers is infinite (there are an infinite number of lines which can be drawn in the plane). However, many of these linear classifiers would assign the same labels to the training data.

The number of unique labellings of the training data that can be achieved with linear classifiers is, in fact, finite. A line can be defined by picking *any* pair of training points, as illustrated in Figure 3(a). Two classifiers can be defined from each such line: one that outputs a label "1" for everything on or above the line, and another that outputs "0" for everything on or above. There exist $\binom{n}{2}$ such pairs of training points, and these define all possible unique labellings of the training data. Therefore, there are at most $2\binom{n}{2}$ unique linear classifiers for any random set of $n$ 2-dimensional features (the factor of 2 is due to the fact that for each linear classifier there are 2 possible assignments of the labelling).




(a) (b)

Figure 3: Fitting a linear classifier to 2-dimensional data. There are an infinite number of such classifiers. (a) We can generate a linear classifier by choosing two data points, drawing a line with both points on one side, and declaring all points on or above the line to be "+1" (or "−1") and all points below the line to be "−1" (or "+1"). From this perspective, we see that the two linear classifiers depicted in (b) are equivalent for this set of data points, and hence relative to the set of $n$ training data there are only on the order of $n^2$ unique linear classifiers.

Thus, instead of infinitely many linear classifiers, we realize that as far as a random sample of $n$ training data is concerned, there are at most

$$\begin{aligned} 2\binom{n}{2} &= \frac{2n!}{(n-2)!2!} \\ &= n(n-1) \end{aligned}$$

unique linear classifiers. That is, using linear classification rules, there are at most $n(n-1) \approx n^2$ unique label assignments for $n$ data points. If we like, we can encode each possibility with $\log_2 n(n-1) \approx 2\log_2 n$ bits. In $d$ dimensions there are $2\binom{n}{d}$ hyperplane classification rules which can be encoded in roughly $d\log_2 n$

bits. Roughly speaking, the number of bits required for encoding each model is the VC dimension. The remarkable aspect of the VC dimension is that it is often finite even when $\mathcal{F}$ is infinite (as in this example).

If $\mathcal{X}$ has $d$ dimensions in total, we might consider linear classifiers based on $1, 2, \ldots, d$ features at a time. Lower dimensional hyperplanes are less complex than higher dimensional ones. Suppose we set

$$\begin{aligned} \mathcal{F}_1 &= \text{linear classifiers using 1 feature} \\ \mathcal{F}_2 &= \text{linear classifiers using 2 features} \\ \ldots &\quad \text{and so on} \end{aligned}$$

These spaces have increasing VC dimensions, and we can try to balance the empirical risk and a cost function depending on the VC dimension. Such procedures are often referred to as *Structural Risk Minimization*. This gives you a glimpse of what the VC dimension is all about. In future lectures we will revisit this topic in greater detail.

## 1.3 Hold-out Methods

The basic idea of "hold-out" methods is to split the $n$ samples $D \equiv \{X_i, Y_i\}_{i=1}^n$ into a training set, $D_T$, and a test set, $D_V$.

$$D_T = \{X_i, Y_i\}_{i=1}^m, \qquad D_V = \{X_i, Y_i\}_{i=m+1}^n$$

Now, suppose we have a collection of different model spaces $\{\mathcal{F}_\lambda\}$ indexed by $\lambda \in \Lambda$ (e.g., $\mathcal{F}_\lambda$ is the set of polynomials of degree $d$, with $\lambda = d$), or suppose that we have a collection of complexity penalization criteria $L_\lambda(f)$ indexed by $\lambda$ (*e.g.*,let $L_\lambda(f) = \widehat{R}(f) + \lambda c(f)$, with $\lambda \in \mathbf{R}^+$). We can obtain candidate solutions using the training set as follows. Define

$$\widehat{R}_m(f) = \sum_{i=1}^m \ell(f(X_i), Y_i)$$

and take

$$\hat{f}_\lambda = \arg\min_{f \in \mathcal{F}_\lambda} \widehat{R}_m(f)$$

or

$$\hat{f}_\lambda = \arg\min_{f \in \mathcal{F}} \left\{ \widehat{R}_m(f) + \lambda c(f) \right\}$$

This provides us with a set of candidate solutions $\{\hat{f}_\lambda\}$. Then we can define the hold-out error estimate using the test set:

$$\widehat{R}_V(f) = \frac{1}{n-m+1} \sum_{i=m+1}^n \ell(f(X_i), Y_i),$$

and select the "best" model to be $\widehat{f} = \widehat{f}_{\widehat{\lambda}}$ where

$$\widehat{\lambda} = \arg\min_{\lambda} \widehat{R}_V \left( \hat{f}_\lambda \right)$$

This type of procedure has many nice theoretical guarantees, provided both the training and test set grow with $n$.

### 1.3.1 Leaving-one-out Cross-Validation (Wahba, 1971)

A very popular hold-out method is the so call "leaving-one-out cross-validation" studied in depth by Grace Wahba (UW-Madison, Statistics). For each $\lambda$ we compute

$$\hat{f}_\lambda^{(k)} = \arg\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{\substack{i=1\\ i\neq k}}^{n} \ell(f(X_i),Y_i) + \lambda C(f)$$

or

$$\hat{f}_\lambda^{(k)} = \arg\min_{f\in\mathcal{F}_\lambda} \frac{1}{n}\sum_{\substack{i=1\\ i\neq k}}^{n} \ell(f(X_i),Y_i).$$

Then we have cross-validation function

$$\begin{aligned} V(\lambda) &= \frac{1}{n}\sum_{k=1}^{n} \ell(f_\lambda^{(k)}(X_k),Y_k) \\ \lambda^* &= \arg\min_\lambda V(\lambda). \end{aligned}$$

# 2 Summary

To summarize, this lecture gave a brief and incomplete survey of different methods for dealing with the issues of overfitting and model selection. Given a set of training data, $D_n = \{X_i, Y_i\}_{i=1}^n$, our overall goal is to find

$$f^* = \arg\min_{f\in\mathcal{F}} R(f)$$

from some collection of functions, $\mathcal{F}$. Because we do not know the true distribution $P_{XY}$ underlying the data points $D_n$, it is difficult to get an exact handle on the risk, $R(f)$. If we only focus on minimizing the empirical risk $\widehat{R}(f)$ we end up overfitting to the training data. Two general approaches were presented.

1. In the first approach we consider an indexed collection of spaces $\{\mathcal{F}_\lambda\}_{\lambda\in\Lambda}$ such that the complexity of $\mathcal{F}_\lambda$ increases as $\lambda$ increases, and
$$\lim_{\lambda\to\infty} \mathcal{F}_\lambda = \mathcal{F}.$$
A solution is given by
$$\widehat{f}_{\lambda^*} = \arg\min_{f\in\mathcal{F}_{\lambda^*}} \widehat{R}_n(f)$$
where either $\lambda^*$ is a function which increases with $n$,
$$\lambda^* = \lambda(n),$$
or $\lambda^*$ is chosen by hold-out validation.

2. The alternative approach is to incorporate a penalty term into the risk minimization problem formulation. Here we consider an indexed collection of penalties $\{C_\lambda\}_{\lambda\in\Lambda}$ satisfying the following properties:
   (a) $C_\lambda : \mathcal{F} \to \mathbf{R}^+$;
   (b) For each $f \in \mathcal{F}$ and $\lambda_1 < \lambda_2$ we have $C_{\lambda_1}(f) \leq C_{\lambda_2}(f)$;
   (c) There exists $\lambda_0 \in \Lambda$ such that $C_{\lambda_0}(f) = 0$ for all $f \in \mathcal{F}$.

   In this formulation we find a solution
$$\widehat{f}_{\lambda^*} = \arg\min_{f\in\mathcal{F}} \widehat{R}_n(f) + C_{\lambda^*}(f),$$
where either $\lambda^* = \lambda(n)$, a function growing the number of data samples $n$, or $\lambda^*$ is selected by hold-out validation.

# 3 Consistency

If an estimator or classifier $\widehat{f}_{\lambda^*}$ satisfies

$$E\left[R(\widehat{f}_{\lambda^*})\right] \to \inf_{f \in \mathcal{F}} R(f) \qquad \text{as } n \to \infty,$$

then we say that $\widehat{f}_{\lambda^*}$ is $\mathcal{F}$-consistent with respect to the risk $R$. When the context is clear, we will simply say that $\widehat{f}$ is consistent.