

Regularization and Complexity

Lecturer: Rob Nowak

Scribe: Michael Rabbat

1 Competing Goals: The Bias-Variance Tradeoff

We ended the previous lecture with a brief discussion of overfitting. Recall that, given a set of n data points, D_n , our goal in solving the learning from data problem is to choose a function $f_n \in \mathcal{F}$ which minimizes the expected risk $E[R(f_n)]$, 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.

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

The *approximation error* term quantifies the performance hit incurred by imposing restrictions on \mathcal{F} , and the *estimation error* term expresses how well the chosen function f_n will perform in relation to the best possible f which could have been chosen from the class \mathcal{F} . This decomposition into expected 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 large¹ 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 we can make the estimation error very small.

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 solve the problem by minimizing the empirical risk:

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

If \mathcal{F} is very large then $\widehat{R}_n(f)$ can be made arbitrarily small and the resulting f_n can “overfit” to the data since $\widehat{R}_n(f)$ will not be a good indicator of $R(f_n)$.

We can't easily determine whether we are over or underfitting just by looking at the empirical risk.

2 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:

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

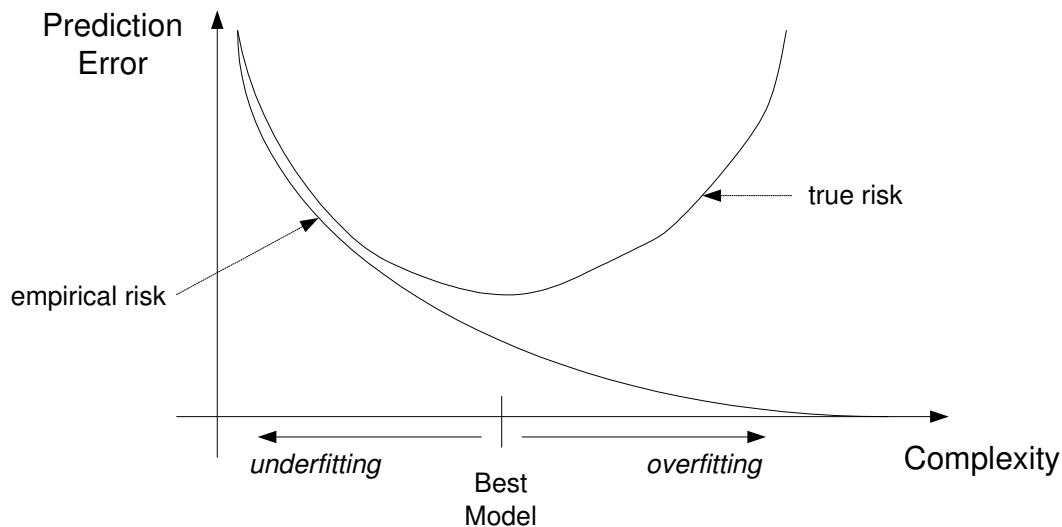


Figure 1: Illustration of overfitting to the data.

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

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

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 “curvier”, with the intuition being that the true function is probably smooth so a function which is very curvy 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.

2.1 Bayesian Methods (Bayes, 1764)

One can interpret the cost $C(f)$ as reflecting prior knowledge about which f 's 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 define $\mathbb{F} = \mathcal{F} \cup \mathcal{F}^c$ with \mathbb{F} being the space of all measurable functions, 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 .

2.2 Description Length Methods (Rissanen, 1978)

Express each f as a string of bits, and accordingly 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

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

As in the Bayesian approach, $p(f) \propto e^{-C(f)}$ 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.

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). A line can be defined by picking *any* pair of training points, as illustrated in Figure 2(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. Therefore, there are at most $2 \binom{n}{2}$ unique classifiers for any random set of n 2-dimensional features.

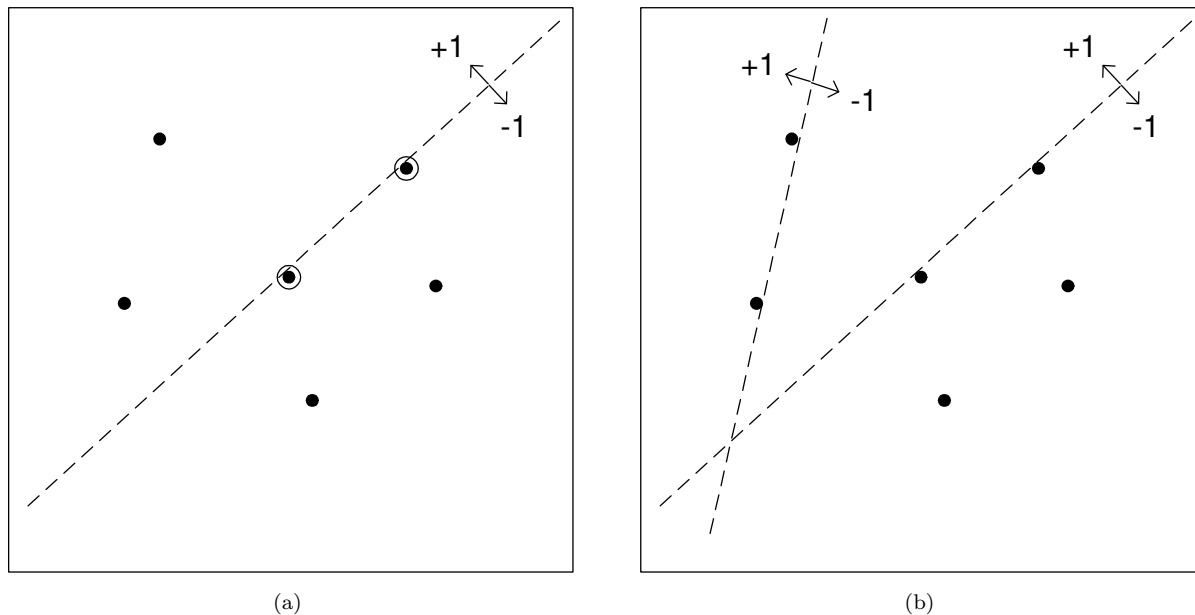


Figure 2: Fitting a linear classifier to 2-dimensional data. There are an infinite number of such classifiers. (a) We can generate a linear classifier by choosing two data points, drawing a line with both points on one side, and declaring all points on or above the line to be “+1” (or “-1”) and all points 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. 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.

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

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

This gives you a glimpse of what the VC dimension is all about. In future lectures we will revisit this topic in greater detail.

2.4 Validation Methods

Split the n training samples $D \equiv \{X_i, Y_i\}_{i=1}^n$ into a training set, D_T , and a validation set, D_V .

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

Now, suppose we have a collection of different model spaces $\{\mathcal{F}_\lambda\}$ indexed by $\lambda \in \Lambda$ (e.g., \mathcal{F}_λ is the set of polynomials of degree d , with $\lambda = d$), or suppose that we have a collection of criteria $L_\lambda(f)$ where λ is an index (e.g., let $L_\lambda(f) = \widehat{R}(f) + \lambda c(f)$, with $\lambda \in \mathbf{R}$). Let

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

and take

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

or

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

We will end up with a set of solutions $\{\hat{f}_\lambda\}$. Then we can take

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

and select the “best” model to be $\hat{f} = \hat{f}_{\hat{\lambda}}$ where

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

2.4.1 Leaving-one-out Cross-Validation (Wahba, 1971)

A particular validation method of interest is the so call “leaving-one-out cross-validation” studied by Grace Wahba here at the UW. For each λ 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

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

2.5 Method of Sieves (Grenander, 1981)

Let the class of candidate functions grow with n . That is, take

$$\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_n, \dots$$

where $|\mathcal{F}_i|$ grows as $i \rightarrow \infty$. 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

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

In the next lecture we will consider an example using the method of sieves.

3 Summary

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

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

for some class 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 $\hat{R}(f)$ 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}_λ increases as λ increases, and

$$\lim_{\lambda \rightarrow \infty} \mathcal{F}_\lambda = \mathcal{F}.$$

A solution is given by

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

where either λ^* is given by some function which increases with n ,

$$\lambda^* = \lambda(n),$$

or λ^* 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) $C_\lambda : \mathcal{F} \rightarrow \mathbf{R}^+$;
- (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

$$\hat{f}_{\lambda^*} = \arg \min_{f \in \mathcal{F}} \hat{R}_n(f) + C_{\lambda^*}(f),$$

where either $\lambda^* = \lambda(n)$, a function growing the number of data samples n , or λ^* is selected by hold-out validation.

4 Consistency

If an estimator or classifier \hat{f}_{λ^*} satisfies

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

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