Statistical Machine Learning Notes 10

Learning Theory

Instructor: Justin Domke

1 Introduction

Most of the methods we have talked about in the course have been introduced somewhat heuristically, in the sense that we have not rigorously proven that they actually work! Roughly speaking, in supervised learning we have taken the following strategy:

- Pick some class of functions $f(\mathbf{x})$ (decision trees, linear functions, etc.)
- Pick some loss function, measuring how we would like $f(\mathbf{x})$ to perform on test data.
- Fit $f(\mathbf{x})$ so that it has a good average loss on training data. (Perhaps using cross-validation to regularize.)

What is missing here is a proof that the performance on training data is indicative of performance on test data. We intuitively know that the more "powerful" the class of functions is, the more training data we will tend to need, but we have not made the definition of "powerful", nor this relationship precise.

In these notes we will study two of the most basic ways of characterizing the "power" of a set of functions. We will look at some rigorous bounds confirming and quantifying our above intuition– that is, for a specific set of functions, how much training data is needed to prove that the function will work well on test data?

2 Hoeffding's Inequality

The basic tool we will use to understand generalization is Hoeffding's inequality. This is a general result in probability theory. It is extremely widely used in machine learning theory. There are several equivalent forms of it, and it is worth understanding these in detail.

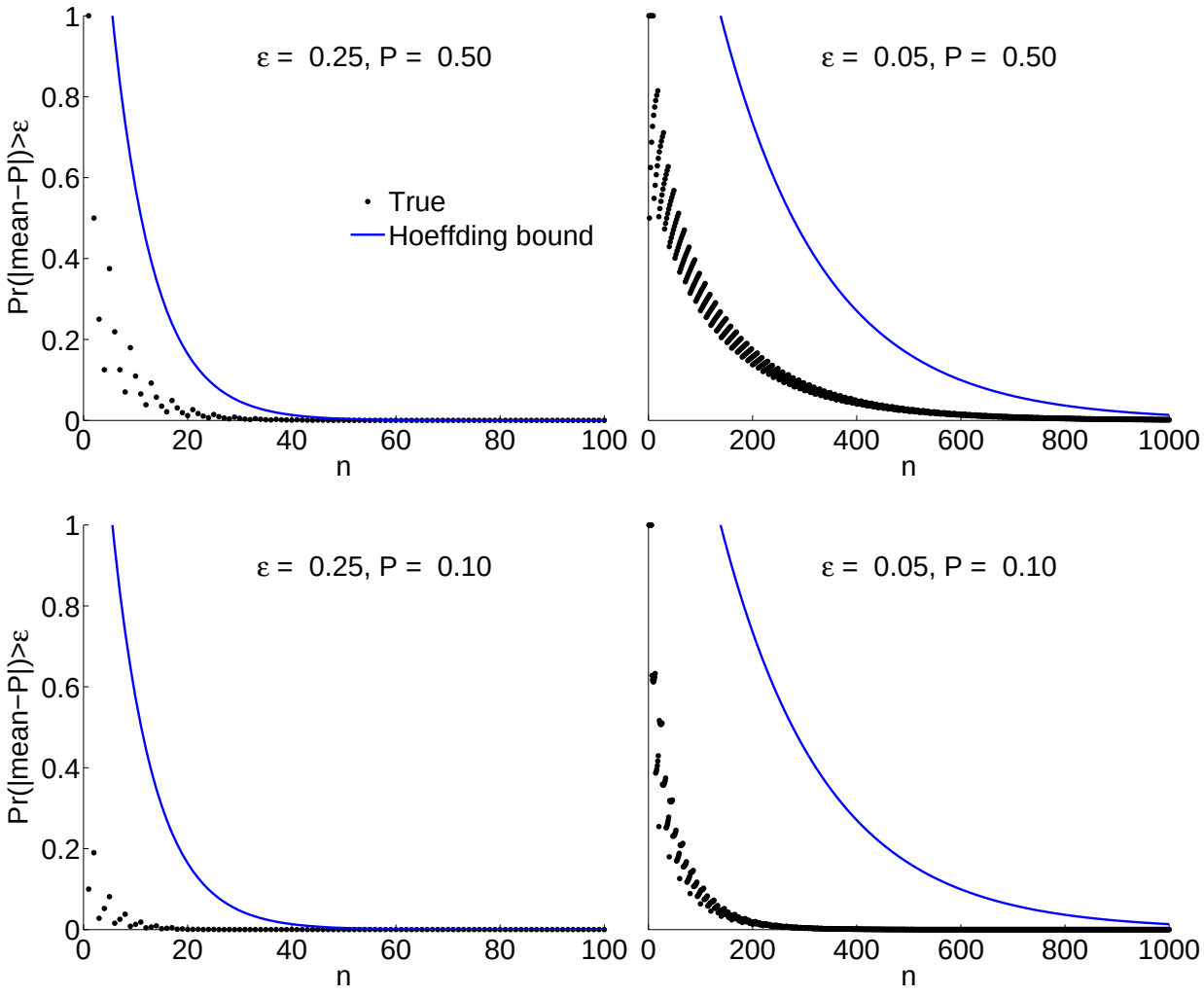
Theorem 1 (Hoeffding's Inequality)**.** *Let $Z_1, ..., Z_n$ be random independent, identically distributed random variables, such that $0 \leq Z_i \leq 1$. Then,*

$$Pr\big[|\frac{1}{n}\sum_{i=1}^{n} Z_i - E[Z]| > \epsilon\big] \leq \delta = 2\exp\big(-2n\epsilon^2\big)$$

The intuition for this result is very simple. We have a bunch of variables Z_i. We know that when we average a bunch of them up, we should usually get something close to the expected value. Hoeffding quantifies "usually" and "close" for us.

(Note that the general form of Hoeffding's inequality is for random variables in some range $a \leq Z_i \leq b$. As we will mostly be worrying about the 0-1 classification error, the above form is fine for our purposes. Note also that can also rescale your variables to lie between 0 and 1, and then apply the above proof.)

The following examples compare Hoeffding's inequality to the true probability of deviating from the mean by more than ϵ for binomial distributed variables, with $E[Z] = P$. For $P = \frac{1}{2}$ the bound is not bad. However, for $P = \frac{1}{10}$ it is not good at all. What is happening is that Hoeffding's inequality does not make use of any properties of the distribution, such as its mean or variance. In a way, this is great, since we can calculate it just from n and ϵ. The price we pay for this generality is that *some distributions* will converge to their means much faster than Hoeffding is capable of knowing.



You should ask yourself: how will these figures look with $P = .9$? Will Hoeffding be loose or tight?

Another form of Hoeffding's inequality of the following[1].

Theorem 2 (Hoeffding-2)**.** *Suppose we choose*

$$n \geq \frac{1}{2\epsilon^2} \log \frac{2}{\delta}. \tag{2.1}$$

Then, with probability at least $1 - \delta$*, the difference between the empirical mean* $\frac{1}{n}\sum_{i=1}^{n} Z_i$ *and the true mean* $E[Z]$ *is at most* ϵ*.*

This second form is very useful. To understand it, we can think of setting two "slop" parameters:

- The "accuracy" ϵ says how far we are willing to allow the empirical mean to be from the true mean.
- The "confidence" δ says what probability we are willing to allow of "failure". (That is, a deviation larger than ϵ)

If we choose these two parameters, Eq. 2.1 tells us how much it will "cost" in terms of samples.

Note that, informally, accuracy is expensive, while confidence is cheap. Explicitly, if we find n for some ϵ, δ, then we decide that we want 10 times more confidence. We can calculate that for $\delta' = \delta/10$, we will need

$$n' = \frac{1}{2}\Big(\frac{1}{\epsilon}\Big)^2 \log \frac{2 \cdot 10}{\delta} = n + \frac{1}{2}\Big(\frac{1}{\epsilon}\Big)^2 \log(10) = n + C(\epsilon)$$

[1] Want proof, do you? Suppose

$$n \geq \frac{1}{2\epsilon^2} \log \frac{2}{\delta}.$$

Then, by Hoeffding's inequality,

$$\begin{aligned} Pr[|\frac{1}{n}\sum_{i=1}^{n} Z_i - E[Z]| > \epsilon] &\leq 2\exp(-2n\epsilon^2) \\ &\leq 2\exp(-2\frac{1}{2\epsilon^2}\log\frac{2}{\delta}\epsilon^2). \\ &= 2\exp(-\log\frac{2}{\delta}). \\ &= \delta \end{aligned}$$

samples to achieve this. Thus, we can just add a *constant* number $C(\epsilon)$ of extra samples. If we would like 100 times more confidence, we can just add $2C(\epsilon)$ extra samples. Another way of looking at this is that $n \propto \log \frac{2}{\delta}$ or $\delta \propto \frac{2}{\exp(n)}$ Just to emphasize: this is *great*, this is the best we could imagine. We will see below that the "cheapness" of confidence turns out to be key to our goal of creating learning bounds.

On the other hand, accuracy is quite expensive. Suppose that we decide we want 10 times more accuracy. We can calculate that for $\epsilon' = \epsilon/10$, we will need $100n$ samples. An increase of a factor of 100! If we want 100 times more accuracy, we will need a factor of 10,000 times more samples. Another way of looking at this is that $\epsilon \propto \frac{1}{\sqrt{n}}$.

Yet another way of stating Hoeffding's inequality is

Theorem 3 (Hoeffding-3)**.** *If we draw n samples, then with probability at least $1-\delta$, the difference between the empirical mean $\frac{1}{n}\sum_{i=1}^{n} Z_i$ and the true mean $E[Z]$ is at most ϵ, where*

$$\epsilon \leq \sqrt{\frac{1}{2n}\log\frac{2}{\delta}}$$

With only this simple tool, we can actually derive quite a lot about learning theory.

3 Finite Hypothesis Spaces

How do we use this result for learning theory? The variables we are interested in bounding are the 0-1 *classification error rates* of classifers. Given some classifier g, we are interested in bounding how far the true error rate $R_{\text{true}}(g)$ is from the observed error rate on n samples, $R_n(g)$. (The notation $R(g)$ is, for the moment, banned.) Here that our bounds are on the $0-1$ classification error.

Given n training samples, we can state bounds on the difference of the observed and true error rates for any classifier g. Namely, using Hoeffding-3, with probability $1-\delta$,

$$|R_{\text{true}}(g) - R_n(g)| \leq \sqrt{\frac{1}{2n}\log\frac{2}{\delta}}.$$

Another way (Hoeffding-2) of stating this result is, if we want that $|R_{\text{true}}(g) - R_n(g)| \leq \epsilon$ with probability at least $1-\delta$, then we should pick

$$n \geq \frac{1}{2\epsilon^2}\log\frac{2}{\delta}. \tag{3.1}$$

Now, let's derive some learning bounds! Here, we will think of things rather more abstractly than we have before. We will think of having a *set* of possible classifiers $\mathcal{G}$. These could

be the set of all decision trees, the set of all linear functions, etc. For now, we assume that the set $\mathcal{G}$ is finite. Note that this is a very strong assumption! When we talked about linear functions, our classifiers were parameterized by some vector of weights $\mathbf{w}$. Since these are real numbers, this represents infinite number of classifiers. We could get a finite set by picking a finite number of weight vectors $\mathbf{w}$, and only considering those.

Now, the above bounds look very nice. We have made no assumptions on the true distribution or the form of the classifiers g, and so these bounds are almost universally applicable.

Unfortunately, they aren't useful. Let's start with a *bad* idea of how to apply them. (This does not work! Don't do it. You have been warned!)

Incorrect, wrong, broken algorithm:

- Input the set of classifiers $\mathcal{G}$.
- Draw $n \geq \frac{1}{2}\left(\frac{1}{\epsilon}\right)^2 \log \frac{2}{\delta}$ samples.
- Output $g^* = \arg\min_{g \in \mathcal{G}} R_n(g)$

We might be tempted to conclude that since $R_n(g)$ is close to $R_{\text{true}}(g)$ with high probability for all g, the output g^* should be close to the best class. Trouble is, it isn't true! One way of looking at the previous results is this: for any given classifier, there aren't too many "bad" training sets for which the empirical risk is far off from the true risk. However, different classifiers can have different bad training sets. If we have 100 classifiers, and each of them is inaccurate on 1% of training sets, it is possible that we always have at least one such that the empirical risk and true risk are far off.

(On the other hand, it is possible that the bad training sets *do* overlap. In practice, this does seem to happen to some degree, and is probably partially responsible for the fact that learning algorithms generalize better in practice than these type of bounds can prove.)

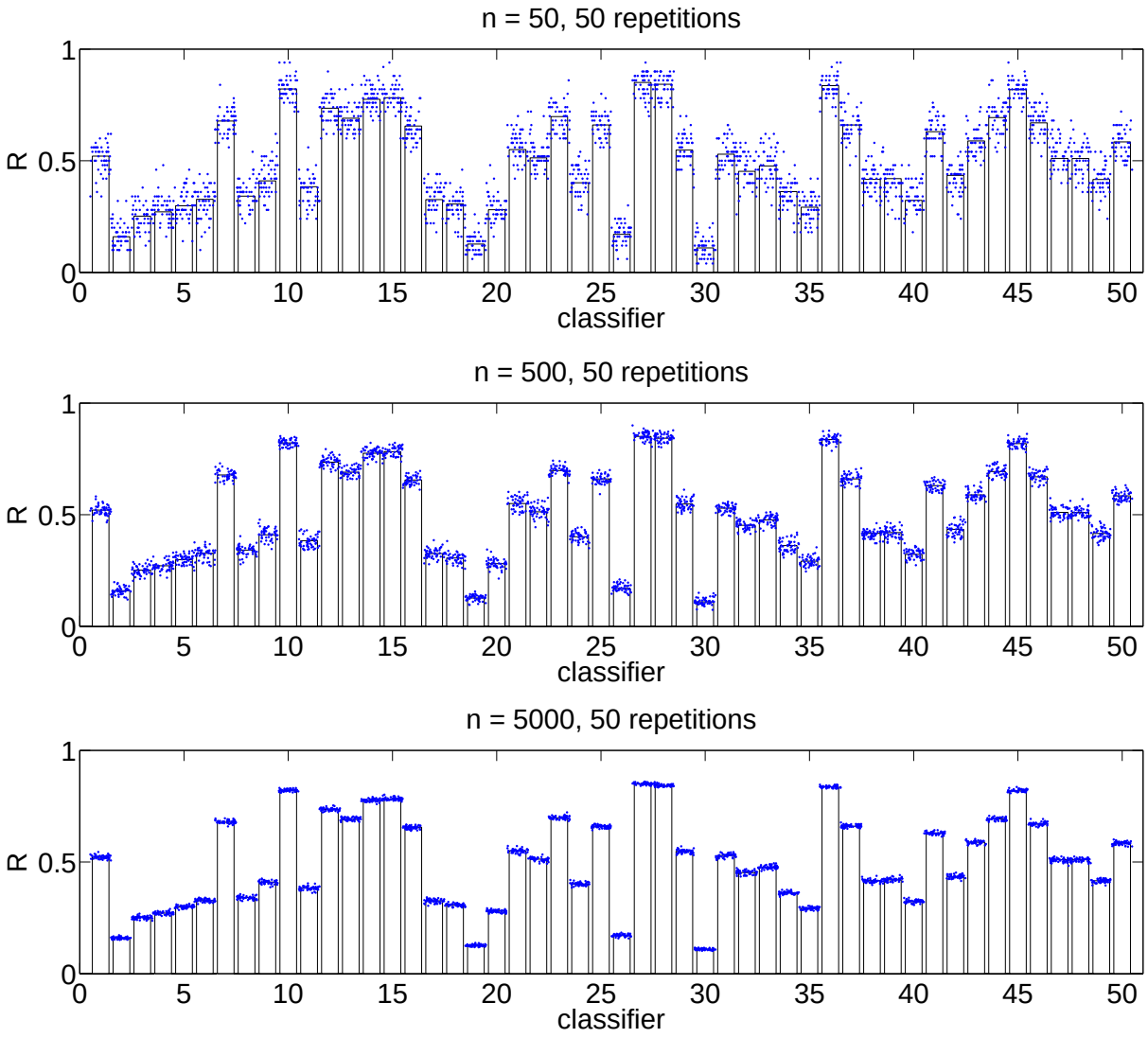
The way to combat this is to make the risk of each particular classifier being off *really* small. If we have 100 classifiers, and want only a 1% chance of failure, we limit the probability of failure of each to 0.01%. Or, if we want an overall probability of failure of δ, we make sure that each individual classifer can only fail with probability $\delta/\mathcal{G}$. Plugging this into our previous result, we have

Theorem 4. *With probability* $1 - \delta$*, for all* $g \in \mathcal{G}$ *simultaneously*

$$|R_{true}(g) - R_n(g)| \leq \sqrt{\frac{1}{2n} \log \frac{2|\mathcal{G}|}{\delta}}. \tag{3.2}$$

Notice that if the different classifiers g are similar to one another, this bound will be loose. In the limit that all g are identical, the right hand side should just be $\sqrt{\frac{1}{2n} \log \frac{2}{\delta}}$.

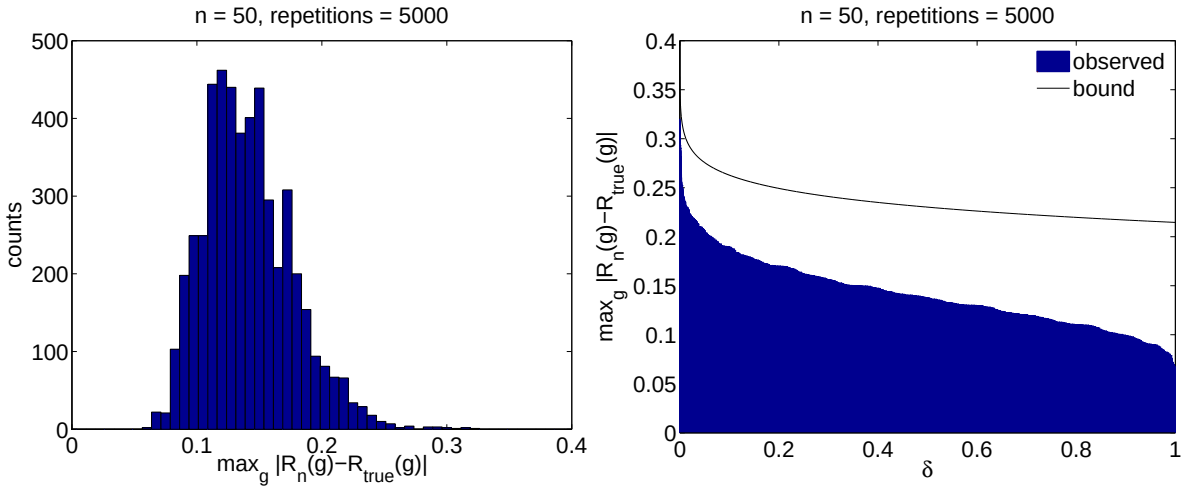
Let's test this theorem out with some experiments. In the following experiment, we pick 50 random linear classifiers in five dimensions, and calculate the true risk[2] for each (shown as a bar graph). The true clasification rule is also linear (not included in the set of 50 classifiers). Then, we repeat the following experiment: draw n samples, and calculate the empirical risk (shown as blue dots). We can clearly see that we estimate the risk more accurately with large n.



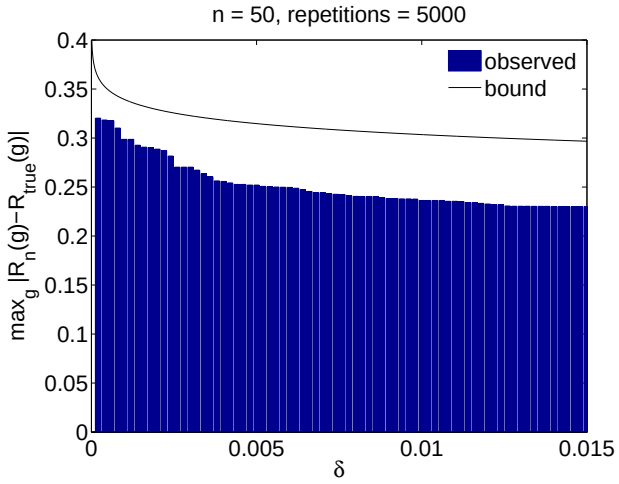
Now, lets see how our bound in Eq. 3.2 compares to reality. First of all, we compute a histogram of the maximum disparity between the empirical and true risk. (Below on left.) Next, we plot the observed disparities, in order. (Below on right, in blue.) This gives us

[2]To be perfectly honest, this was approximated on 10^6 random samples.

an estimate of probabilities. For example, the median observed disparity is about .138, telling us that with probability $\frac{1}{2}$ we see a maximum deviation of less than .138. Our bound, meanwhile, only guarantees that we should see a deviation of less than about .230.



Since in general we are interested in bounds we can state with high confidence, lets zoom in on the range of small δ. Here, we see that the bound performs relatively better.



4 Structural Risk Minimization

Lets recall the model selection problem. When doing supervised learning, we pick some class of functions $\mathcal{G}$, then pick the function g in that class with the the lowest empirical risk. The model selection problem is this: What set of function $\mathcal{G}$ should we use?

Notice that our bound in Eq. 5.1 exhibits a bias-variance tradeoff. If we pick a big set of

functions, some g might have a low risk. Thus, as $\mathcal{G}$ gets bigger, $\min_{g\in\mathcal{G}} R_{\text{true}}(g)$ will be non-increasing. (This means a decrease in bias.) On the other hand, the more functions we have, the more danger there is that one happens to score well on the training data. Thus, as $\mathcal{G}$ gets bigger, $2\sqrt{\frac{1}{2n}\log\frac{2|\mathcal{G}|}{\delta}}$ is increasing. (This means an increase in variance.)

In structural risk minimization, we have a sequence of function sets or "models", $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathcal{G}_3$, etc. We assume that they are strictly increasing, and so

$$\mathcal{G}_1 \subset \mathcal{G}_2 \subset \mathcal{G}_3 \subset ....$$

We would like to select $\mathcal{G}_i$ to trade-off between bias and variance. Now, we define

$$g_i^* = \arg\min_{g\in\mathcal{G}_i} R_n(g_i^*).$$

This is the function that minimizes the empirical risk under the model $\mathcal{G}_i$. This, our question is if we should output g_1^*, g_2^*, etc. We know that for more complex models, g_i^* will be selected from a larger set, and so could fit the true function better. On the other hand, we run a greater risk of overfitting.

Recall that we know, from Eq. 3.2, that for all $g \in \mathcal{G}_i$ simultaneously,

$$R_{\text{true}}(g) \leq R_n(g) + \sqrt{\frac{1}{2n}\log\frac{2|\mathcal{G}_i|}{\delta}}.$$

Consequently, this is also true for g_i^*. This gives us a bound on the true risk of each of the functions, in terms of only the empirical risk and the size of the model.

$$\boxed{R_{\text{true}}(g_i^*) \leq R_n(g_i^*) + \sqrt{\frac{1}{2n}\log\frac{2|\mathcal{G}_i|}{\delta}}.} \tag{4.1}$$

What we would like to to is minimize $R_{\text{true}}(g_i^*)$. Since we can't do that, the idea of strucural risk minimization is quite simple: minimize the bound!

Structural Risk Minimization using a Hoeffding bound

- For all i, compute $g_i^* = \arg\min_{g\in\mathcal{G}_i} R_n(g)$
- Pick i such that $R_n(g_i^*) + \sqrt{\frac{1}{2n}\log\frac{2|\mathcal{G}_i|}{\delta}}$ is minimized.
- Output g_i^*

This is an alternative to methods like cross-validation. Note that in practice the bound in Eq. 4.1 is often very loose. Thus, while SRM may be based on firmer theoretical foundations than cross-validation, this does not mean that it will *work better*.

5 More on Finite Hypothesis Spaces

We can also ask another question. Let g^* be the best classifier in $\mathcal{G}$.

$$g^* = \arg\min_{g \in \mathcal{G}} R_{\text{true}}(g)$$

We might try to bound the probability that empirical risk minimization picks g^*. This is hard to do, because if there is some other classifier that has a true risk very close to that of g^*, a huge number of samples could be required to distinguish them. On the other hand, we don't care too much– if we pick some classifier that has a true risk close to that of g^*, we should be happy with that. So, instead we will bound the difference of the true risk of the classifier picked by empirical risk minimization and the true risk of the optimal classifier.

Theorem 5. *With probability at least* $1 - \delta$*,*

$$R_{\text{true}}\big(\arg\min_{g \in \mathcal{G}} R_n(g)\big) \leq \min_{g \in \mathcal{G}} R_{\text{true}}(g) + 2\sqrt{\frac{1}{2n} \log \frac{2|\mathcal{G}|}{\delta}}. \tag{5.1}$$

What this says is this: Take the function g^* minimizing the empirical risk. Then, with high probability $(1 - \delta)$, the true risk of g^* will not be too much worse that the true risk of the best function. Put another way, empirical risk minimization *usually* picks a classifier with a true risk *close* to optimal, where "usually" is specified as δ and "close" is the constant on the right hand side.

You should prove this as an exercise. The basic idea is that, with high probability $R_n(g') \geq R_{\text{true}}(g') - \epsilon$, while simultaneously $R_n(g^*) \leq R_{\text{true}}(g^*) + \epsilon$. Since g' did best on the training data, we have $R_{\text{true}}(g') \leq R_n(g') + \epsilon \leq R_n(g^*) + \epsilon \leq R_{\text{true}}\,(g*) + 2\epsilon$.

We have one more theorem for finite hypothesis spaces.

Theorem 6. *If we set*

$$n \geq \frac{1}{2\epsilon^2} \log \frac{2|\mathcal{G}|}{\delta}, \tag{5.2}$$

then with probability $1 - \delta$*, for all* $g \in \mathcal{G}$ *simultaneously,*

$$|R_{\text{true}}(g) - R_n(g)| \leq \epsilon.$$

We can prove this by recalling (Eq. 3.1). This tells us that the above result must hold for each g independently with probability $\delta/|\mathcal{G}|$. Thus, it must hold for all of them with probability δ.

6 Infinite Spaces and VC Dimension

The big question we want to answer is, given a set of functions, how much data do we need to collect to fit the set of functions reliably. The above results suggest that for finite sets of functions, the amount of data is (at most) logarithmic in the size of the set. These results don't seem to apply to most of the classifiers we have discussed in this class, as they generally fit real numbers, and so involve an infinite set of possible functions. On the other hand, regardless of the fact that we use real numbers in analysing or algorithms, we use digital computers, which represent numbers only to a fixed precision[3]. So, for example, if we are fitting a linear model with 10 weights, on a computer that uses 32 bits to represent a float, we actually have a large but finite number of possible models: $2^{32\cdot 10}$. More generally, if we have P parameters, Eq. 5.2 suggests using

$$n \geq \frac{1}{2\epsilon^2} \log \frac{2 \cdot 2^{32P}}{\delta} = \frac{1}{2\epsilon^2}\big(\log \frac{2}{\delta} + 32P \log 2\big)$$

samples suffices. This is reassuringly intuitive: the number of samples required is linear in the number of free parameters. On the other hand, this isn't a particularly *tight* bound, and it is somehow distasteful to suddenly pull our finite precision representation of parameters out of a hat, when this assumption was not taken into account when deriving the other methods. (It seems to violate good taste to switch between real numbers and finite precision for analysis whenever one is more convenient.)

Theorem 7. *With probability* $1 - \delta$*, for all* $g \in \mathcal{G}$ *simultaneously,*

$$R_{true}(g) \leq R_n(g) + \sqrt{\frac{VC[\mathcal{G}]}{n}\big(\log \frac{n}{VC[\mathcal{G}]} + \log 2e\big) + \frac{1}{n} \log \frac{4}{\delta}}. \tag{6.1}$$

Where $VC[\mathcal{G}]$ is the VC-dimension of the set $\mathcal{G}$, which we will define presently.

Take some set of points $S = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_d\}$. We say that $\mathcal{G}$ "shatters" S if the points can be classified in all possibile ways by $\mathcal{G}$. Formally stated, $\mathcal{G}$ shatters S if

[3]Technically, most algorithms could be implement using an abitrary precision library, but don't think about that.

$$\forall y_i \in \{-1, +1\},\ \exists g \in \mathcal{G} : y_i = \operatorname{sign} g(\mathbf{x}_i).$$

The VC-dimension of $\mathcal{G}$ is defined to be the size of the *largest* set S that can be shattered by $\mathcal{G}$. Notice that we only need to be able to find *one* set of size d that can be shattered in order for the VC dimension to be at least d.

Examples:

- A finite set of classifiers, can only produce (at most) $|\mathcal{G}|$ possible different labelings of any group of points. Shattering a group of d points requires 2^d labelings, implying that thus, $2^{VC[\mathcal{G}]} \leq |\mathcal{G}|$, and so $VC[\mathcal{G}] \leq \log_2 |\mathcal{G}|$.

$$R_{\text{true}}(g) \leq R_n(g) + \sqrt{\frac{\log_2 |\mathcal{G}|}{n}\big(\log \frac{n}{\log_2 |\mathcal{G}|} + \log 2e\big) + \frac{1}{n}\log \frac{4}{\delta}}.$$

- VC-dimension of linear classifier in D dimensions is at least D. Choose set of points

$$\begin{aligned} \mathbf{x}_1 &= (1, 0, 0, ..., 0) \\ \mathbf{x}_2 &= (0, 1, 0, ..., 0) \\ \mathbf{x}_3 &= (0, 0, 1, ..., 0) \end{aligned}$$

  For this fixed set of points, we claim that for any set of labels $y_i \in \{-1, +1\}$ it is possible to find a vector of weights $\mathbf{w}$ such that $y_i = \operatorname{sign}(\mathbf{w} \cdot \mathbf{x}_i)$ for all i. (Actually, this is really easy! Just choose $w_i = y_i$.) In fact $VC[\mathcal{G}] = D$, though the upper-bound proof is somewhat harder.

- VC-dimension of SVM with polynomial kernel $k(\mathbf{x}, \mathbf{v}) = (\mathbf{x} \cdot \mathbf{v})^p$ is $\binom{D+p-1}{p}$ where D is the length of $\mathbf{x}$.

Just as with the Hoeffding bound above, we can use the the VC bound for structural risk minimization. The assumption is now, we have a sequence of functions $\mathcal{G}_1$, $\mathcal{G}_2$, etc., where the VC dimension of the sets is increasing.

Structural Risk Minimization with VC dimension.

- For all i, compute $g_i^* = \arg\min_{g \in \mathcal{G}_i} R_n(g)$
- Pick i such that $R_n(g_i^*) + \sqrt{\frac{VC[\mathcal{G}_i]}{n}\big(\log \frac{n}{VC[\mathcal{G}_i]} + \log 2e\big) + \frac{1}{n}\log \frac{4}{\delta}}$ is minimized.
- Output g_i^*

7 Discussion

The fundamental weakness of the above bounds is their looseness. In practice, the bound on the difference between the training risk and true risk in Eq. 6.1 is often hundreds of times higher than the true difference. There are many worst-case assumptions leading to the bound that are often not so bad in practice. Tightening the bounds remains an open area of research. On the other hand, the bound can sometimes work well in practice despite its looseness. The reason for this is that we are fundamentally interested in performing model selection, not bounding test errors. The model selected by structural risk minimization is sometimes quite good, despite the looseness of the bound.

Sources:

- A Tutorial on Support Vector Machines for Pattern Recognition, Burges, Data Mining and Knowledge Discovery, 1998
- Introduction to Statistical Learning Theory, Bousquet, Boucheron, Lugosi, Advanced Lectures on Machine Learning Lecture Notes in Artificial Intelligence, 2004