23 Learning Theory

LEARNING THEORY: WHAT IS GENERALIZATION?

[One thing humans do well is generalize. When you were a young child, you only had to see a few examples of cows before you learned to recognize cows, including cows you had never seen before. You didn't have to see every cow. You didn't even have to see $\log n$ of the cows.]

[Learning theory tries to explain how machine learning algorithms generalize, so they can classify data they've never seen before. It also tries to derive mathematically how much training data we need to generalize well. Learning theory starts with the observation that if we want to generalize, we must constrain what hypotheses we allow our learner to consider.]

A range space (aka set system) is a pair (P, H), where
P is set of all possible test/training points (can be infinite)
H is hypothesis class, a set of hypotheses (aka ranges, aka classifiers):
each hypothesis is a subset $h \subseteq P$ that specifies which points h predicts are in class C.
[So each hypothesis h is a 2-class classifier, and H is a set of sets of points.]

Examples:

1. Power set classifier: P is a set of k numbers; H is the power set of P, containing all 2^k subsets of P.
   e.g. $P = \{1, 2\}, H = \{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$
2. Linear classifier: $P = \mathbb{R}^d$; H is the set of all halfspaces; each halfspace has the form $\{x : w \cdot x \geq -\alpha\}$.
   [In this example, both P and H are infinite. In particular, H contains every possible halfspace—that is, every possible linear classifier in d dimensions.]

[The power set classifier sounds very powerful, because it can learn every possible hypothesis. But the reality is that it can't generalize at all. Imagine we have three training points and three test points in a row.]

? C C ? N ?

[The power set classifier can classify these three test points any way you like. Unfortunately, that means it has learned nothing about the test points from the training points. By contrast, the linear classifier can learn only two hypotheses that fit this training data. The leftmost test point must be classified class C, and the rightmost test point must be classified class Not-C. Only the test point in the middle can swing either way. So the linear classifier has a big advantage: it can generalize rapidly. That's also a big disadvantage if the data isn't close to linearly separable, but that's another story.]

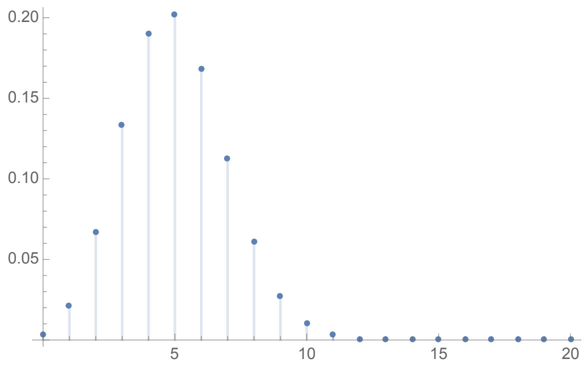
[Now we will investigate how well the training error predicts the test error, and how that differs for these two classifiers.]

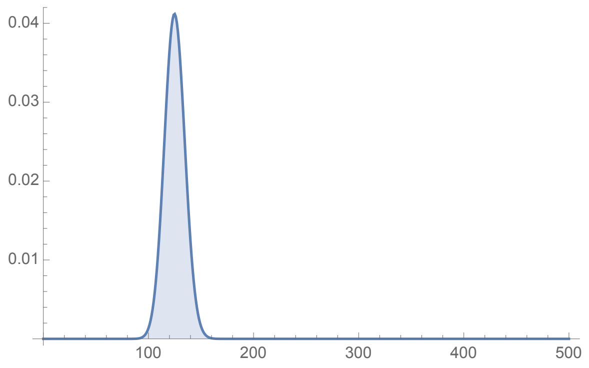
Suppose all training pts & test pts are drawn independently from same prob. distribution $\mathcal{D}$ defined on domain P. [$\mathcal{D}$ also determines each point's label. Classes C and Not-C may have overlapping distributions.]

Let $h \in H$ be a hypothesis [a classifier]. h predicts a pt x is in class C if $x \in h$.
The risk aka generalization error $R(h)$ of h is the probability that h misclassifies a random pt x drawn from $\mathcal{D}$—i.e. the prob. that $x \in$ C but $x \notin h$ or vice versa.
[Risk is almost the same as the test error. To be precise, the risk is the average test error for test points drawn from $\mathcal{D}$. Sometimes the test error is higher, sometimes lower, but on average it is $R(h)$. If you had an infinite amount of test data, the risk and the test error would be the same.]

Let $X \subseteq P$ be a set of n training pts drawn from $\mathcal{D}$
The empirical risk aka training error $\hat{R}(h)$ is % of X misclassified by h.
[This matches the definition of empirical risk I gave you in Lecture 12, if you use the 0-1 loss function.]

h misclassifies each training pt w/prob. $R(h)$, so total misclassified has a binomial distribution.
As $n \to \infty$, $\hat{R}(h)$ better approximates $R(h)$.





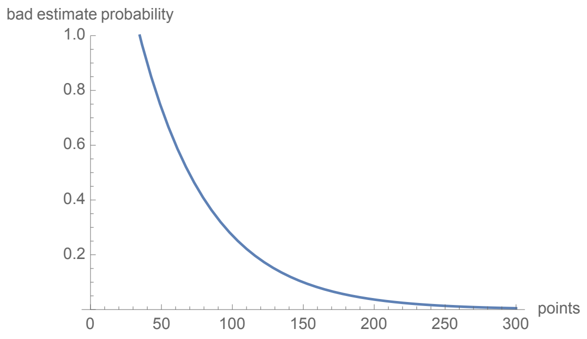
$\boxed{\text{binom20.pdf, binom500.pdf}}$ [Consider a hypothesis whose risk of misclassification is 25%. Plotted are distributions of the number of misclassified training points for 20 points and 500 points, respectively. For 20 points, the training error is not a reliable estimate of the risk: the hypothesis might get “lucky” with misleadingly low training error.]

[If we had infinite training data, this distribution would become infinitely narrow and the training error would always be close to the risk, just like the test error when we have infinite test data. But we can’t have infinite training data. So, how well does the training error approximate the risk?]

Hoeffding’s inequality tells us prob. of bad estimate:

$$\Pr(|\hat{R}(h) - R(h)| > \epsilon) \leq 2e^{-2\epsilon^2 n}.$$

[Hoeffding’s inequality is a standard result about how likely it is that a number drawn from a binomial distribution will be far from its mean. If n is big enough, then it’s very unlikely.]



$\boxed{\text{hoeffding.pdf}}$ [Hoeffding’s bound for the unambitious $\epsilon = 0.1$. It takes at least 200 training points to have some reasonable confidence of attaining that error bound.]

[One reason this matters is because we will try to choose the best hypothesis. If the training error is a bad estimate of the test error, we might choose a hypothesis we think is good but really isn’t. So we are happy to see that the likelihood of that decays exponentially in the amount of training data.]

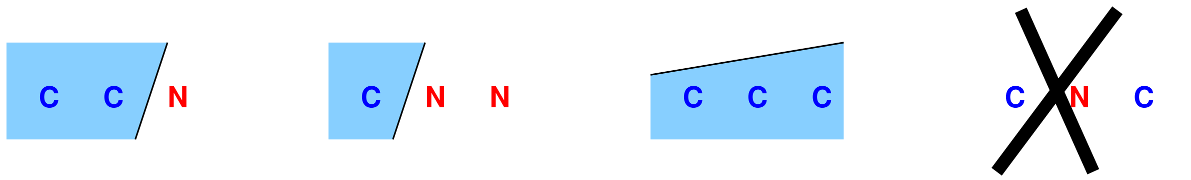
Idea for learning alg: choose $\hat{h} \in H$ that minimizes $\hat{R}(\hat{h})$! Empirical risk minimization.
[None of the classification algorithms we've studied actually do this, but only because it's computationally infeasible to pick the best hypothesis. Support vector machines can find a linear classifier with zero training error when the training data is linearly separable. When it isn't, SVMs try to find a linear classifier with low training error, but they don't generally find the one with minimum training error. That's NP-hard.]
[Nevertheless, for the sake of understanding learning theory, we're going to pretend that we have the computational power to try every hypothesis and pick the one with the lowest training error.]

Problem: if too many hypotheses, some h with high $R(h)$ will get lucky and have very low $\hat{R}(h)$!

[This brings us to a central idea of learning theory. You might think that the ideal learning algorithm would have the largest class of hypotheses, so it could find the perfect one to fit the data. But the reality is that you can have so many hypotheses that some of them just get lucky and score far lower training error than their actual risk. That's another way to understand what "overfitting" is.]
[More precisely, the problem isn't too *many* hypotheses. Usually we have infinitely many hypotheses, and that's okay. The problem is too many dichotomies.]

Dichotomies

A dichotomy of X is $X \cap h$, where $h \in H$.
[A dichotomy picks out the training points that h predicts are in class C. Think of each dichotomy as assigning each training point to class C or class Not-C.]



[Draw this by hand. $\boxed{\text{dichotomies.pdf}}$ Three examples of dichotomies for three points in a hypothesis class of linear classifiers, and one example (right) that is not a dichotomy.]

[For n training points, there could be up to 2^n dichotomies. The more dichotomies there are, the more likely it is that one of them will get lucky and have misleadingly low empirical risk.]

Extreme case: if H allows all 2^n possible dichotomies, $\hat{R}(\hat{h}) = 0$ even if every $h \in H$ has high risk.
[If our hypothesis class permits all 2^n possible assignments of the n training points to classes, then one of them will have zero training error. But that's true even if all of the hypotheses are terrible and have a large risk. Because the hypothesis class imposes no structure, we overfit the training points.]

Given Π dichotomies, Pr(at least one dichotomy has $|\hat{R} - R| > \epsilon) \leq \delta$, where $\delta = 2\Pi\, e^{-2\epsilon^2 n}$.
[Let's fix a value of δ and solve for ϵ.] Hence with prob. $\geq 1 - \delta$, for every $h \in H$,

$$|\hat{R}(h) - R(h)| \leq \epsilon = \sqrt{\frac{1}{2n} \ln \frac{2\Pi}{\delta}}.$$

[This tells us that the smaller we make Π, the number of possible dichotomies, and the larger we make n, the number of training points, the more likely it is that the training error will be an accurate measure of how well the classifer actually performs on test data.]

smaller Π or larger $n \Rightarrow$ training error probably closer to true risk (& test error).

[Smaller Π means we're less likely to overfit. We have less variance, but more bias. This doesn't mean the risk will be small. If our hypothesis class H doesn't fit the data well, both the training error and the test error will be large. In an ideal world, we want a hypothesis class that fits the data well, yet doesn't have many hypotheses.]

Let $h^* \in H$ minimize $R(h^*)$; "best" classifier.
[Remember we picked the classifier $\hat{h}$ that minimizes the empirical risk. We really want the classifier h^* that minimizes the actual risk, but we can't know what h^* is. But if Π is small and n is large, the hypothesis $\hat{h}$ we have chosen is probably nearly as good as h^*.]

With prob. $\geq 1 - \delta$, our chosen $\hat{h}$ has nearly optimal risk:

$$R(\hat{h}) \leq \hat{R}(\hat{h}) + \epsilon \leq \hat{R}(h^*) + \epsilon \leq R(h^*) + 2\epsilon, \qquad \epsilon = \sqrt{\frac{1}{2n} \ln \frac{2\Pi}{\delta}}.$$

[This is excellent news! It means that with enough training data and a limit on the number of dichotomies, empirical risk minimization usually chooses a classifier close to the best one in the hypothesis class.]

Choose a δ and an ϵ.
The <u>sample complexity</u> is the # of training pts needed to achieve this ϵ with prob. $\geq 1 - \delta$:

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

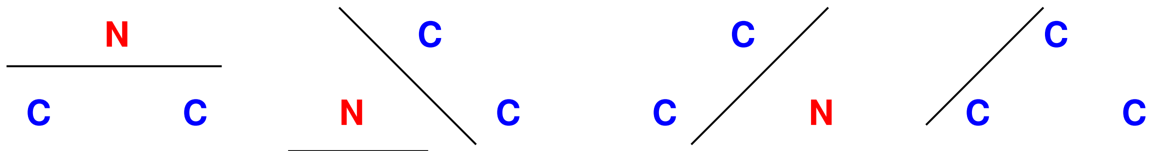
[If Π is small, we won't need too many samples to choose a good classifier. Unfortunately, if $\Pi = 2^n$ we lose, because this inequality says that n has to be bigger than n. So the power set classifier can't learn much or generalize at all. We need to severely reduce Π, the number of possible dichotomies. One way to do that is to use a linear classifier.]

The Shatter Function & Linear Classifiers

[How many ways can you divide n points into two classes with a hyperplane?]

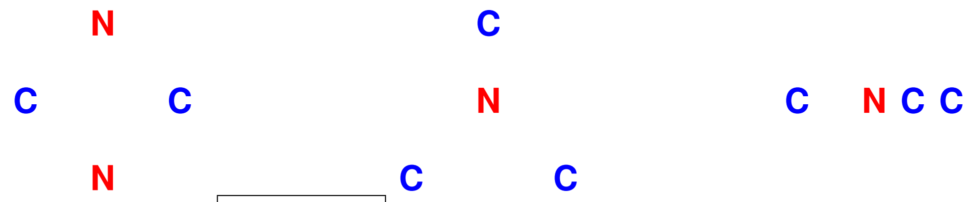
$$\begin{aligned}
\text{\# of dichotomies: } \Pi_H(X) &= |\{X \cap h : h \in H\}| && \in [1, 2^n] \text{ where } n = |X| \\
\underline{\text{shatter function}}\text{: } \Pi_H(n) &= \max_{|X|=n, X \subseteq P} \Pi_H(X) && \text{[The most dichotomies of any point set of size } n\text{]}
\end{aligned}$$

Example: Linear classifiers in plane. H = set of all halfplanes. $\Pi_H(3) = 8$:



[Draw this by hand. $\boxed{\text{shatter.pdf}}$ Linear classifiers can induce all eight dichotomies of these three points. The other four dichotomies are the complements of these four.]

$\Pi_H(4) = 14$:
[Instead of showing you all 14 dichotomies, let me show you dichotomies that halfplanes *cannot* learn, which illustrate why no four points have 16 dichotomies.]



[Draw this by hand. $\boxed{\text{unshatter.pdf}}$ Examples of dichotomies of four points in the plane that no linear classifier can induce.]

[This isn't a proof that 14 is the maximum, because we have to show that 16 is not possible for *any* four points in the plane. The standard proof uses a result called Radon's Theorem.]

Fact: for all range spaces, either $\Pi_H(n)$ is polynomial in n, or $\Pi_H(n) = 2^n \;\; \forall n \geq 0$.

[This is a surprising fact with deep implications. Imagine that you have m points, some of them training points and some of them test points. Either a range space permits every possible dichotomy of the points, and the training points don't help you classify the test points at all; or the range space permits only a polynomial subset of the 2^m possible dichotomies, so once you have labeled the training points, you have usually cut down the number of ways you can classify the test points dramatically. No shatter function ever occupies the no-man's-land between polynomial and 2^m.]

[For linear classifiers, we know exactly how many dichotomies there can be.]

Cover's Theorem [1965]: linear classifiers in $\mathbb{R}^d$ allow up to $\Pi_H(n) = 2\sum_{i=0}^{d}\binom{n-1}{i}$ dichotomies of n pts.

For $n \leq d+1$, $\Pi_H(n) = 2^n$.
For $n \geq d+1$, $\Pi_H(n) \leq 2\left(\frac{e(n-1)}{d}\right)^d$ [Observe that this is polynomial in n! With exponent d.]
and the sample complexity needed to achieve $R(\hat{h}) \leq \hat{R}(\hat{h}) + \epsilon \leq R(h^*) + 2\epsilon$ with prob. $\geq 1-\delta$ is

$$n \geq \frac{1}{2\epsilon^2}\left(d \ln \frac{n-1}{d} + d + \ln \frac{4}{\delta}\right).$$ [Observe that the logarithm turned the exponent d into a factor!]

Corollary: linear classifiers need only $n \in O(d)$ training pts
for training error to accurately predict risk/test error.

[In a d-dimensional feature space, we need more than d training points to train an accurate linear classifier. But it's reassuring to know that the number we need is linear in d. By contrast, if we have a classifier that permits all 2^n possible dichotomies however large n is, then no amount of training data will guarantee that the training error of the hypothesis we choose approximates the true risk.]

[The constant hidden in that big-O notation can be quite large. It could be 100 or more, depending on your choices of ϵ and δ. If you want a lot of confidence that you've chosen one of the best hypotheses, you have to pay for it with a large sample size.]

[This sample complexity applies even if you add polynomial features or other features, but you have to count the extra features in d. So the number of training points you need increases with the number of polynomial terms.]

VC Dimension

The Vapnik–Chervonenkis dimension of (P, H) is

$$\mathrm{VC}(H) = \max\{n : \Pi_H(n) = 2^n\}. \qquad \Leftarrow \text{Can be } \infty.$$

Say that H shatters a set X of n pts if $\Pi_H(X) = 2^n$.
$\mathrm{VC}(H)$ is size of largest X that H can shatter.
[This means that X is a point set for which all 2^n dichotomies are possible.]

[I told you earlier that if the shatter function isn't 2^n for all n, then it's a polynomial in n. The VC dimension is motivated by an observation that sometimes makes it easy to bound that polynomial.]

Theorem: $\Pi_H(n) \leq \sum_{i=0}^{\mathrm{VC}(H)} \binom{n}{i}$. Hence for $n \geq \mathrm{VC}(H)$, $\Pi_H(n) \leq \left(\frac{en}{\mathrm{VC}(H)}\right)^{\mathrm{VC}(H)}$.

[So the VC dimension is an upper bound on the exponent of the polynomial. This theorem is useful because often we can find an easy upper bound on the VC dimension. You just need to show that for some number n, no set of n points can have all 2^n dichotomies.]

Corollary: $O(\mathrm{VC}(H))$ training pts suffice for accuracy. [The hidden constant is much bigger than 1.]

[If the VC dimension is finite, it tells us how the sample complexity grows with the number of features. If the VC dimension is infinite, no amount of training data will make the classifier generalize well.]

Example: Linear classifiers in plane.
Recall $\Pi_H(3) = 8$: there exist 3 pts shattered by halfplanes.
But $\Pi_H(4) = 14$: no 4 pts are shattered.
Hence:

- $\mathrm{VC}(H) = 3$ [The VC dimension of halplanes is 3.]
- $\Pi_H(n) \leq \frac{e^3}{27} n^3$ [The shatter function is polynomial.]
- $O(1)$ sample complexity.

[The VC dimension doesn't always give us the tightest bound. In this example, the VC dimension promises that the number of ways halfplanes can classify the points is at worst cubic in n; but in reality, it's quadratic in n. In general, linear classifiers in d dimensions have VC dimension $d + 1$, which is one dimension looser than the exponent Cover proved. That's not a big deal, though, as the sample complexity and the accuracy bound are both based on the *logarithm* of the shatter function. So if we get the exponent wrong, it only changes a constant in the sample complexity.]

[The important thing is simply to show that there is some polynomial bound on the shatter function at all. VC dimension is not the only way to do that, but often it's the easiest.]

[The main point you should take from this lecture is that if you want to have generalization, you need to limit the expressiveness of your hypothesis class so that you limit the number of possible dichotomies of a point set. This may or may not increase the bias, but if you don't limit the number of dichotomies at all, the overfitting will be very bad. If you limit the hypothesis class, your artificial child will only need to look at $O(d)$ cows to learn the concept of cows. If you don't, your artificial child will need to look at every cow in the world, and every non-cow too.]