CS761 Spring 2013 Advanced Machine Learning

Statistical Learning Theory

Lecturer: Xiaojin Zhu *jerryzhu@cs.wisc.edu*

Consider a family of binary classifiers $G = \{g : X \mapsto \{-1, 1\}\}$. G can be either probabilistic models or not, such as decision trees, neural nets, SVMs, logic rules, the groundhog family of Punxsutawney Phil, a tank full of Paul the Octopus' relatives, etc. Each $g \in G$ predicts the label $y = g(x)$ from input x.

Importantly, assume an *unknown but fixed* joint distribution $p(x, y)$ from which the training and future test items are sampled. Now consider the 0-1 loss. This leads to the *risk* of g

$$R(g) = \mathbb{E}(g(x) \neq y).$$

★ *The expectation is over $(x, y) \sim p$. $g(x) \neq y$ takes value in $\{0, 1\}$. The "test set error" in practice is an unbiased estimate of the risk.*

Our ultimate goal is to pick $g \in G$ so that the risk is minimized. It is important to understand three fundamental limits on how low the risk can go:

1. Since $p(y \mid x)$ may not be "crisp", there is a Bayes risk lower bound on *any* classifier. Let $\eta(x) = \mathbb{E}(Y \mid X = x) = 2p(y = 1 \mid x) - 1$. Then the Bayes classifier is $B(x) = \text{sign}(\eta(x))$. It achieves the minimum risk over all measurable functions $R(B) = \inf_g R(g)$ (Note the inf is not restricted to G). This risk is known as the Bayes risk $R(B) = \mathbb{E}(\frac{1-|\eta(x)|}{2})$.

2. It is possible that $B \notin G$. For example, G may consists of all linear classifiers but B may have a nonlinear decision boundary. In this case, we settle for finding $g^* = \arg\inf_{g \in G} R(g)$. The gap $R(g^*) - R(B)$ is known as the *approximation error*, i.e., the error incurred from approximating the concept B with an incorrect set G.

3. We are only given a finite training set $(x_1, y_1) \ldots (x_n, y_n) \overset{iid}{\sim} p$, not p itself, to find g^*. In general, our learning algorithm will return some $\hat{g} \in G$ based on the training set. $\hat{g}$ usually does not coincide with g^*. This incurs another gap $R(\hat{g}) - R(g^*)$, which is known as the *estimation error*. Namely, the error stemming from estimating g^* with limited data. Note the estimation error is a random variable, while the approximation error is not.

With the definitions above, we have

$$R(\hat{g}) - R(B) = [R(\hat{g}) - R(g^*)] + [R(g^*) - R(B)] = \text{estimation error} + \text{approximation error}.$$

★ *This decomposition is similar to the bias-variance trade-off, with estimation error playing the role of variance and approximation error playing the role of bias.*

In particular, we will focus on the *empirical risk minimizer*:

$$\hat{g} = \text{argmin}_{g \in G} R_n(g)$$

where

$$R_n(g) = \frac{1}{n} \sum_{i=1}^{n} g(x_i) \neq y_i.$$

★ *Yes, empirical risk minimization is a fancy name for minimizing training error, i.e., overfitting. As every practitioner knows, we shouldn't use this $\hat{g}$. However, the analysis is particularly convenient. Actually, we are going to precisely quantify overfitting – that's the whole point of this lecture. Furthermore, the behavior of other $\hat{g}$'s, such as the regularized empirical risk minimizer, can be analyzed, too.*

Furthermore, for simplicity we will focus on bounding estimation error and disregard approximation error. That is, we want to have a statement that for "most" training sets sampled from p,

$$R(\hat{g}) \le R(g^*) + \text{func}(n, G).$$

This will be made precise soon. The key ingredient is a bound between the empirical risk and the true error $R_n(g) - R(g)$, to which we now turn to.

1 For a fixed g chosen before seeing training data

To start, let us consider any pre-specified $g \in G$ before seeing the training data. This is important: we cannot pick $g \in G$ using the training data, or what follows will not apply.

The strong law of large numbers states that $R_n(g) \to R(g)$ almost surely as $n \to \infty$. However, neither this law nor the central limit theorem quantifies the relation between $R_n(g)$ and $R(g)$ for a finite n. Instead, this is studied by the so-called concentration of measure. Hoeffding's inequality is one such concentration inequality.

Theorem 1 (Hoeffding) *Let $Z_1, \ldots, Z_n$ be independent with $P(Z_i \in [a, b]) = 1$ and the same mean μ. Then for all $\epsilon > 0$,*

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} Z_i - \mu\right| > \epsilon\right) \le 2e^{-\frac{2n\epsilon^2}{(b-a)^2}}. \tag{1}$$

In our problem, $Z_i = l(g(x_i), y_i)$ where $l(y', y) = [y' \ne y]$ is the 0-1 loss function. It is easy to see that $\frac{1}{n}\sum_{i=1}^{n} Z_i = R_n(g)$, and $\mu = R(g)$. Furthermore, $Z_i \in [0, 1]$ because of the 0-1 loss. Therefore, Hoeffding's inequality gives

$$P\left(|R_n(g) - R(g)| > \epsilon\right) \le 2e^{-2n\epsilon^2}. \tag{2}$$

In fact, there is a one-sided bound

$$P\left(R(g) - R_n(g) > \epsilon\right) \le e^{-2n\epsilon^2}. \tag{3}$$

Defining $\delta = e^{-2n\epsilon^2}$, we can rewrite

$$\epsilon = \sqrt{\frac{\log \frac{1}{\delta}}{2n}}. \tag{4}$$

This leads to the following statement:

Theorem 2 *For a fixed g, for any $\delta > 0$, with probability at least $1 - \delta$*

$$R(g) \le R_n(g) + \sqrt{\frac{\log \frac{1}{\delta}}{2n}}. \tag{5}$$

Two things should be noted. First, if g is selected using a training set, this bound does not apply (see next section). Second, the probability $1 - \delta$ is w.r.t. training set generation. Consider all training sets of size n. Some of such sets are "good" in the sense of (5), namely g's true error is not too far from its training error on the set. The theorem says that with probability at least $1 - \delta$, your randomly generated training set will be a good one.

★ *How tight is the Hoeffding bound? Consider $\delta = 0.05$, namely 95% of the training sets will satisfy the bound. For this δ, the deviation $\sqrt{\frac{\log \frac{1}{\delta}}{2n}}$ is 0.39 for $n = 10$ (this is pretty bad – $R_n(g)$ is not very indicative of $R(g)$; not very surprising though given the very small training set); 0.12 for $n = 100$; 0.04 for $n = 1000$, and 0.01 for $n = 10,000$ (need a lot of training data to achieve one percent). Keep in mind though, this does not apply to a learned (i.e., somehow picked using training data) classifier!*

2 Uniform deviation for finite G

"Learning" or "training" means we select $\hat{g} \in G$ using the training set. The simple bound (5) no longer apply because now $\hat{g}$ is selected based on the particular training set (you can show this with simulation). One way to proceed is to look at *uniform deviation*, namely bounding

$$\sup_{g \in G} R_(g) - R_n(g)$$

because any upper bound is also a bound on the trained classifier $\hat{g}$:

$$R(\hat{g}) - R_n(\hat{g}) \leq \sup_{g \in G} R_(g) - R_n(g).$$

Uniform deviation can be better understood if we look again at the meaning of δ for a fixed g. Let's explicitly define the "bad" training sets for g:

$$BAD_g = \{(x, y)_{1:n} : R(g) - R_n(g) > \epsilon\}.$$

Hoeffding's says that

$$P(\text{your training set} \in BAD_g) \leq e^{-2n\epsilon^2}.$$

Now, you have many $g \in G$, each has its own BAD_g (which in general may not be the same). Assume $|G|$ is finite for now (which is not true for things like all linear classifiers). We observe that the *union bound* states that

$$P(\text{your training set} \in \cup_{g \in G} BAD_g) \leq \sum_{g \in G} P(\text{your training set} \in BAD_g) = |G| e^{-2n\epsilon^2}.$$

Inverting it, we get the probability that your training set is good for all $g \in G$: With probability at least $1 - |G|e^{-2n\epsilon^2}$, $\sup_{g \in G} R(g) - R_n(g) \leq \epsilon$. Again, let $\delta = |G|e^{-2n\epsilon^2}$ and we arrive at a uniform deviation bound.

Theorem 3 *For any $\delta > 0$, with probability at least $1 - \delta$,*

$$\sup_{g \in G} R(g) - R_n(g) \leq \sqrt{\frac{\log |G| + \log(1/\delta)}{2n}}.$$

This bound holds "uniformly," i.e., for all $g \in G$. The price we have to pay is a loose bound with the additional $\log |G|$ term which stems from the union bound. This immediately leads to a bound on the particular learned classifier $\hat{g}$:

$$R_(\hat{g}) - R_n(\hat{g}) \leq \sqrt{\frac{\log |G| + \log(1/\delta)}{2n}}. \tag{6}$$

★ *Example. There has been 56 US presidential elections. There are $|G| = 3100$ US counties. Under great simplifications, view each county's voting outcome as a classifier g: election-index $\mapsto$ candidate. Suppose you looked for and found a county $\hat{g}$ that always correctly predicts the election outcome, namely $R_n(\hat{g}) = 0$. How well does this county predict future elections? With probability at least 0.95,*

$$R(\hat{g}) \leq 0 + \sqrt{\frac{\log(3100) + \log(1/0.05)}{2 \times 56}} = 0.31.$$

The bound is not that great.

3 Bounding the estimation error for finite G

So far we have been bounding $R(g) - R_n(g)$ for all $g \in G$. In the special case where our algorithm performs empirical risk minimization so that $\hat{g} = \text{argmin}_{g \in G} R_n(g)$, we can easily bound the estimation error. The key is $R_n(\hat{g}) \leq R_n(g^*)$.

$$
\begin{aligned}
& R(\hat{g}) - R(g^*) && (7)\\
= \;& R(\hat{g}) - R_n(\hat{g}) + R_n(\hat{g}) - R(g^*) && (8)\\
\leq \;& R(\hat{g}) - R_n(\hat{g}) + R_n(g^*) - R(g^*) && (9)\\
\leq \;& 2 \sup_{g \in G} |R(g) - R_n(g)| && (10)\\
= \;& 2\sqrt{\frac{\log |G| + \log(2/\delta)}{2n}}. && (11)
\end{aligned}
$$

4 Countably Infinite G with a prior distribution $q(g)$ (Occam's Razor bound)

Of course, in many cases G is infinite and the bounds in previous sections are vacuous. In the special case where G is countable and there is a prior distribution $q(g)$ (in the sense that q does not depend the training set), there is a simple extension to the bounds, which we state now. The more general case would need more advanced techniques such as the VC-dimension or the Rademacher complexity, which we discuss in later sections.

Where does the prior q come from? One possibility is a prefix code. That is, each $g \in G$ is encoded by a binary sequence, such that no g is a prefix of another g'. Then the code length $c(g)$ can be used to define a prior distribution $q(g) = e^{-c(g)\log 2}$ such that the shorter the code, the higher the probability. The Kraft inequality states that $\sum_{g \in G} q(g) \leq 1$.

Let us look at how the previous union bound fail on infinite G:

$$P(\text{your training set} \in \cup_{g \in G} BAD_g) \leq \sum_{g \in G} P(\text{your training set} \in BAD_g) = \sum_{g \in G} e^{-2n\epsilon^2} = |G|e^{-2n\epsilon^2} = \infty e^{-2n\epsilon^2}.$$

The idea is to replace

$$\sum_{g \in G} e^{-2n\epsilon^2}$$

with

$$\sum_{g \in G} q(g) e^{-2n\epsilon^2}$$

so that the sum is finite and bounded by $e^{-2n\epsilon^2}$. We can do this by defining

$$VeryBAD_g = \left\{ (x,y)_{1:n} : R(g) - R_n(g) > \sqrt{\epsilon^2 - \frac{\log q(g)}{2n}} \right\},$$

such that Hoeffding's inequality gives

$$P(\text{your training set} \in VeryBAD_g) \leq q(g) e^{-2n\epsilon^2} = e^{-2n\left(\epsilon^2 - \frac{\log q(g)}{2n}\right)}.$$

The union bound now gives the desired

$$P(\text{your training set} \in \cup_{g \in G} VeryBAD_g) \leq \sum_{g \in G} q(g) e^{-2n\epsilon^2} \leq e^{-2n\epsilon^2}.$$

Inverting, we get

Theorem 4 *For any $\delta > 0$, with probability at least $1 - \delta$,*

$$\forall g \in G, R(g) - R_n(g) \leq \sqrt{\frac{-\log q(g) + \log(1/\delta)}{2n}}.$$

If the learned model $\hat{g}$ happen to have high prior probability $q(\hat{g})$, its bound will be tight. In other words, good prior knowledge leads to tight bound. Keep in mind though q needs to be determined without using the training data. Also note that if G is finite and q is uniform, we recover Theorem 3.

5 Growth number

In general, however, the prior trick does not work. The key idea is that even an infinite G can only produce a finite number of classification patterns on any training set of size n, thus forming a finite number of equivalent classes. With proper care, things go back to the finite regime.

We first define the number of classification patterns. A single g produces an n-vector $(g(x_1) \neq y_1, \ldots g(x_n) \neq y_n)$ on the training data. The set of n-vectors produced by G

$$\{(g(x_1) \neq y_1, \ldots g(x_n) \neq y_n) : g \in G\}$$

has at most 2^n members. Clearly this set depends on the training data: in the (unlikely) event that $x_1 = \ldots = x_n, y_1 = \ldots = y_n$ the set can have at most 2 members. But let's think of the training data that produces the largest such set, and call the size of the set $S_G(n)$:

$$S_G(n) = \sup_{(x_1,y_1)\ldots(x_n,y_n)} \left|\{(g(x_1) \neq y_1, \ldots g(x_n) \neq y_n) : g \in G\}\right|. \tag{12}$$

$S_G(n)$ is known as the *growth number.*

Example 1 *Let $G = \{ax + b \geq 0 : a, b \in \mathbb{R}\}$ for $x \in \mathbb{R}$ (abuse of notation: the Boolean expression returns 1 or -1). G consists of 1D threshold classifiers. Then $S_G(1) = 2$ since we can find a classifier to correctly classify some (x, y), or misclassify it. $S_G(2) = 4$ since we can find four classifiers to produce all possible patterns on $(1, 1), (2, 1)$. $S_G(3) = 6$ since the following patterns are possible:*

```
1 1 1
0 1 1
0 0 1
0 0 0
1 0 0
1 1 0
```

while 1 0 1 and 0 1 0 are impossible. In general for this G, $S_G(n) = 2n$.

To use the growth number, we first introduce an interesting symmetrization lemma which turns a bound on infinite G into another bound on patterns on $2n$ items (i.e., finite!). We introduce a *ghost sample* of size n: $(x'_1, y'_1) \ldots (x'_n, y'_n) \overset{iid}{\sim} p$. The ghost sample is drawn from the same p as the training sample. The key is that the ghost and real training samples are independent. The ghost sample is purely a conceptual object – we don't really have to generate them. We define

$$R'_n(g) = \frac{1}{n}\sum_{i=1}^{n} g(x'_i) \neq y'_i$$

to be the ghost empirical risk. Like the empirical risk $R_n(g)$, $R'_n(g)$ is a random variable since it depends on the ghost sample.

Lemma 1 (Symmetrization) *For all* $\epsilon \geq \sqrt{\frac{2\log 2}{n}}$,

$$P\left(\sup_{g\in G} R(g) - R_n(g) > \epsilon\right) \leq 2P\left(\sup_{g\in G} R'_n(g) - R_n(g) > \frac{\epsilon}{2}\right)$$

Proof: Let the "worst difference classifier" be

$$g_w = \arg\sup_{g\in G} R(g) - R_n(g).$$

Note g_w is a random variable since it depends on the training sample. We will use 1_z to denote the Boolean function on z. Consider

$$1_{R(g_w)-R_n(g_w)>\epsilon} 1_{R(g_w)-R'_n(g_w)<\frac{\epsilon}{2}} \tag{13}$$
$$= 1_{(R(g_w)-R_n(g_w)>\epsilon)\wedge\left(R'_n(g_w)-R(g_w)>-\frac{\epsilon}{2}\right)} \tag{14}$$
$$\leq 1_{R'_n(g_w)-R_n(g_w)>\frac{\epsilon}{2}}. \tag{15}$$

To see the last step, note that $(x-y>a)\wedge(z-x>b)$ implies $z-y>a+b$, but not vice versa. To see this, let $a=b=0$. $z>y$ does not mean $z>x>y$ (which is $x>y\wedge z>x$). In other words, sometimes $1_{z-y>a+b}=1$ but $1_{x-y>a\wedge z-x>b}=0$.

Take the expectation w.r.t. the ghost sample,

$$1_{R(g_w)-R_n(g_w)>\epsilon} P'\left(R(g_w) - R'_n(g_w) < \frac{\epsilon}{2}\right) \tag{16}$$
$$\leq P'\left(R'_n(g_w) - R_n(g_w) > \frac{\epsilon}{2}\right). \tag{17}$$

Now, g_w is picked according to the real sample. From the perspective of P' it is just some fixed classifier. Hence the Hoeffding bound (1) applies:

$$P'\left(R(g_w) - R'_n(g_w) > \frac{\epsilon}{2}\right) \leq e^{-2n(\epsilon/2)^2}.$$

Due to our assumption $\epsilon \geq \sqrt{\frac{2\log 2}{n}}$, it is easy to show $e^{-2n(\epsilon/2)^2} \leq 1/2$. Taking the complement gives

$$P'\left(R(g_w) - R'_n(g_w) < \frac{\epsilon}{2}\right) \geq 1 - e^{-2n(\epsilon/2)^2} \geq 1/2.$$

This leads to

$$1_{R(g_w)-R_n(g_w)>\epsilon} \cdot \frac{1}{2} \leq P'\left(R'_n(g_w) - R_n(g_w) > \frac{\epsilon}{2}\right).$$

Since we picked g_w, the LHS can be rewritten as

$$1_{\sup_{g\in G} R(g)-R_n(g)>\epsilon}.$$

We also have

$$P'\left(R'_n(g_w) - R_n(g_w) > \frac{\epsilon}{2}\right) \leq P'\left(\sup_{g\in G} R'_n(g) - R_n(g) > \frac{\epsilon}{2}\right) = P\left(\sup_{g\in G} R'_n(g) - R_n(g) > \frac{\epsilon}{2}\right),$$

where the inequality comes from sup, and the equality is because of symmetry between ghost and real samples. These give

$$1_{\sup_{g\in G} R(g)-R_n(g)>\epsilon} \leq 2P\left(\sup_{g\in G} R'_n(g) - R_n(g) > \frac{\epsilon}{2}\right).$$

Finally, taking expectation w.r.t. the real sample gives the lemma. ■

Let

$$G(2n) = \{(g(x_1) \neq y_1, \dots g(x_n) \neq y_n, g(x'_1) \neq y'_1, \dots g(x'_n) \neq y'_n) : g \in G\}$$

be the set of $2n$-vectors produced by $g \in G$ on the real and ghost samples. By definition $|G(2n)| \leq S_G(2n)$. We now apply the union bound:

$$\begin{aligned} P\left(\sup_{g\in G} R(g) - R_n(g) > \epsilon\right) &\leq 2P\left(\sup_{g\in G} R'_n(g) - R_n(g) > \frac{\epsilon}{2}\right) && (18)\\ &= 2P\left(\max_{g\in G(2n)} R'_n(g) - R_n(g) > \frac{\epsilon}{2}\right) && (19)\\ &\leq 2|G(2n)|P\left(R'_n(g) - R_n(g) > \frac{\epsilon}{2}\right) && (20)\\ &\leq 2S_G(2n)P\left(R'_n(g) - R_n(g) > \frac{\epsilon}{2}\right). && (21)\end{aligned}$$

There is a version of the Hoeffding's inequality that bounds the difference between two samples:

$$P\left(R'_n(g) - R_n(g) > t\right) \leq e^{-nt^2/2}.$$

We arrive at an important result.

Theorem 5 (Vapnik and Chervonenkis) *Let G be a class of binary functions. For any $\epsilon \geq \sqrt{\frac{2\log 2}{n}}$,*

$$P\left(\sup_{g\in G} R(g) - R_n(g) > \epsilon\right) \leq 2S_G(2n)e^{-n\epsilon^2/8}$$

and hence, with probability at least $1-\delta$,

$$\sup_{g\in G} R(g) - R_n(g) \leq 2\sqrt{2\frac{\log S_G(2n) + \log\frac{2}{\delta}}{n}}.$$

Note: this is a tighter bound than the VC-dimension bound below.

6 VC dimension

Recall Example 1. In general for that G, $S_G(n) = 2n$. However, there is something fundamentally different for $n = 1$ and $n = 2$: our classifiers were able to produce all possible patterns. That is, we can write $S_G(n) = 2^n$ for $n = 1, 2$. In contrast, when $n \geq 3$ there were patterns we couldn't produce.

If $S_G(n) = 2^n$, there is a training set of size n where G can produce all patterns. We say G *shatters* that training set.

Definition 1 *The Vapnik-Chervonenkis (VC) dimension of G is the largest n such that $S_G(n) = 2^n$. Equivalently, the VC dimension is the size of the largest training set that G can shatter.*

Example 2 *Let G be all linear classifiers in $\mathbb{R}^d$. Then the VC dimension of G is $d+1$.*

Example 3 *The VC dimension is not simply measuring the number of parameters in G. The one-parameter family $G = \{sign(\sin(ax)) : a \in \mathbb{R}\}$ has infinite VC dimension.*

Here is the important but subtle relation between the growth function $S_G(n)$ and the VC dimension: $S_G(n)$ grows *exponentially* up to $n = VCdim(G)$, but for larger n it reduces to a much slower *polynomial* growth. Think of Example 1 when $n \geq 3$.

Theorem 6 (Sauer) *Let G has finite VC dimension h. Then for all n*

$$S_G(n) \le \sum_{i=0}^{h} \binom{n}{i}.$$

For all $n \ge h$

$$S_G(n) \le \left(\frac{en}{h}\right)^h.$$

Applying the last inequality on $S_G(2n)$ we immediately get the following bound, expressed with VC dimension:

Corollary 1 *Let G be a class of binary functions with VC dimension h. With probability at least $1-\delta$,*

$$\sup_{g \in G} R(g) - R_n(g) \le 2\sqrt{2\frac{h\log n + h\log\frac{2e}{h} + \log\frac{2}{\delta}}{n}}.$$

In this uniform deviation bound the important term is $\sqrt{\frac{h\log n}{n}}$.

7 VC Entropy

We briefly introduce a few other *capacity measures* and the associated bounds in the next few sections.

Let $G(xy_{1:n}) = \{(g(x_1) \ne y_1, \ldots, g(x_n) \ne y_n) : g \in G\}$ be the projection of G onto the specific training sample $(x,y)_{1:n}$. Previously we took the supreme over all training samples of size n to obtain the growth number. Hence, growth number is independent of the underlying joint distribution $p(x,y)$. We now introduce a capacity measure that depends on p.

Definition 2 (VC entropy) *The VC entropy is defined as*

$$H_G(n) = \log \mathbb{E}[|G(xy_{1:n})|].$$

Theorem 7 *For any $\delta > 0$, with probability at least $1-\delta$,*

$$\forall g \in G, R(g) \le R_n(g) + 2\sqrt{2\frac{H_G(2n) + \log 2/\delta}{n}}.$$

8 Covering Numbers

We define a random metric over G, based on the training sample:

$$d_n(g,g') = \frac{1}{n}\sum_{i=1}^{n} 1_{g(x_i) \ne g'(x_i)}.$$

Define the ball with radius ϵ:

$$B(g,\epsilon) = \{g' \in G : d_n(g,g') \le \epsilon\}.$$

We say that the set $g_1, \ldots, g_m$ is an ϵ-cover of G if

$$G \subset \cup_{i=1:m} B(g_i,\epsilon).$$

Definition 3 (Covering number) *The covering number of G at radius ϵ with respect to d_n, $N(G,\epsilon,n)$, is the minimum size of the ϵ-cover.*

Theorem 8 *For any $t > 0$,*

$$P\left(\exists g \in G : R(g) > R_n(g) + t\right) \le 8\mathbb{E}[N(G,t,n)]e^{-nt^2/128}.$$

9 Rademacher Complexity

Consider F to be a class of functions f such that $0 \leq f(z) \leq 1$. For binary classification, let $z = (x, y)$ and $f(z) = 1_{g(x) \neq y}$ which is a one-one mapping between f and g: f is the 0-1 loss of g. Let

$$Pf = \mathbb{E}(f(Z)) = R(g)$$

and

$$P_n f = \frac{1}{n} \sum_{i=1}^{n} f(Z_i) = R_n(g).$$

Random variables $\sigma_1, \ldots, \sigma_n$ are called Rademacher random variables if they are iid and $P(\sigma_i = 1) = P(\sigma_i = -1) = \frac{1}{2}$.

Definition 4 (Rademacher complexity) *The Rademacher complexity of F is*

$$Rad_n(F) = \mathbb{E}\left(\sup_{f \in F} \left| \frac{1}{n} \sum_{i=1}^{n} \sigma_i f(Z_i) \right| \right)$$

where the expectation is over σ and Z.

The empirical Rademacher complexity of F is

$$Rad_n(F, Z^n) = \mathbb{E}_\sigma \left(\sup_{f \in F} \left| \frac{1}{n} \sum_{i=1}^{n} \sigma_i f(Z_i) \right| \right)$$

where $Z^n = (Z_1, \ldots, Z_n)$ is the training sample, and the expectation is over σ only.

Theorem 9 *With probability at least $1 - \delta$,*

$$\sup_{f \in F} |P_n f - Pf| \leq 2Rad_n(F) + \sqrt{\frac{\log(2/\delta)}{2n}}$$

and

$$\sup_{f \in F} |P_n f - Pf| \leq 2Rad_n(F, Z^n) + \sqrt{\frac{4\log(2/\delta)}{n}}.$$

References

[1] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In O. Bousquet, U. von Luxburg, and G. Rsch, editors, *Advanced Lectures on Machine Learning*, pages 169–207. Springer, 2004.