5.1 Review: Hoeffding’s inequality (simplified)

If \( x_1, \ldots, x_n \in [0, 1] \) are independent random variables, then Hoeffding’s inequality states that

\[
\Pr \left( \frac{1}{n} \sum_{i=1}^{n} x_i - E \left( \frac{1}{n} \sum_{i=1}^{n} x_i \right) \geq t \right) \leq \exp(-2t^2n)
\]

The left hand side dies off extremely quickly as \( t \) increases. Note that if we want to get a bound in terms of the absolute value of the deviation then we get a probability bound that increases by 2. We can also solve for \( t \) in terms of a given probability \( \delta \):

\[
\delta \geq 2 \exp(-2t^2n) \\
\mapsto \log \left( \frac{2}{\delta} \right) \leq 2t^2 n \\
\mapsto t \geq \sqrt{\frac{\log(2/\delta)}{2n}}
\]

**Fact 5.1.** With probability \( 1 - \delta \) we have

\[
\left\| \frac{1}{n} \sum_{i} x_i - E \left( \frac{1}{n} \sum_{i} x_i \right) \right\| \leq \sqrt{\frac{\log(2/\delta)}{2n}}
\]

5.2 One more deviation bound

We want to ensure by taking enough random samples that some event doesn’t occur “often”, that is with probability less than \( \epsilon \). Formally, let \( x_i \in \{0, 1\} \) with \( \Pr(x_i = 1) \geq \epsilon \). What is the probability that \( \sum_{i=1}^{n} x_i = 0 \)? We know that

\[
\prod_{i=1}^{n} (\Pr(x_i = 0)) \leq (1 - \epsilon)^n = \exp(n \log(1 - \epsilon))
\]

Since log is a concave function, \( \log(1 + x) \leq x \) for any \( x \in \mathbb{R} \). So \( \exp(n \log(1 - \epsilon)) \leq e^{-nx} \).

**Fact 5.2.** If \( n \geq \frac{\log(1/\delta)}{x} \) then with probability \( 1 - \delta \), \( \sum_{i=1}^{n} x_i \neq 0 \).

Note that since \( x^2 < x \) for small, positive values of \( x \), this is a tighter lower bound on \( n \) than the one given by Hoeffding’s inequality for small \( \epsilon \).
5.3 Sketch of a typical machine learning problem and support vector machines

In a linear classification problem, we are given data \((x_1, y_1), \ldots, (x_n, y_n)\) independently and identically from a distribution \(D\). Here \(x_i \in \mathbb{R}^d\) and \(y_i \in \{-1, 1\}\). We want to find \(w \in \mathbb{R}^d\), a weight coefficient vector such that \(\Pr(\text{sgn}(w^T x) \neq y)\) is small for all future \((x, y) \sim D\). One way to find \(w\) is by solving the following maximization problem:

\[
\arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \max(0, 1 - y_i (w^T x_i)) + \lambda \|w\|^2
\]

where \(\lambda \in \mathbb{R}\) is a chosen parameter. The function within the arg min term is the support vector machine’s loss function, called the hinge loss.

**Definition 5.3** (Training Error). The training error, written as \(\text{err}_n(w)\), is

\[
\frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[\text{sgn}(w^T x_i) \neq y_i]
\]

The hinge loss function is an approximation of the training error. Using the hinge loss function or otherwise, pick some \(w \in \mathbb{R}^d\) such that \(\text{err}_n(w) \leq \epsilon\). How do we know that the model is “good”? First, a definition:

**Definition 5.4** (Ideal Test Error). The test error, written as \(\text{err}(w)\), is

\[
\mathbb{E}(\mathbb{1}[\text{sgn}(w^T x) \neq y]) = \Pr((w^T x)y \leq 0)
\]

where the expectation and probability are taken over distribution \(D\), \((x, y) \sim D\).

A good model has small ideal test error. An erroneous approach is as follows. Pick \(\hat{w}_n = \arg\min_{w \in \mathbb{R}^d} \mathbb{1}[(w^T x_i)y_i \leq 0]\).

Apply Hoeffding’s inequality:

\[
I_i = \mathbb{1}[(w^T x_i)y \leq 0]
\]

\[
|\text{err}_n(w) - \text{err}(w)| = \frac{1}{n} \sum_{i=1}^{n} I_i - \mathbb{E}(I) \leq \sqrt{\frac{\log(2/\delta)}{2n}}
\]

with probability \(1 - \delta\). This argument is false because \(I_i\) are intentionally correlated to fit the data. They are no longer independent.

5.4 PAC-Learning: “Probably Approximately Correct”

Key pieces:

- \(X\) input space
- Output space \(Y = \{0, 1\}\)
- Concept class \(C\), a set of function families taking \(X\) to \(Y\).

Here \(C\) can be viewed as part of \(P(X)\), the power set of \(X\).

**Definition 5.5.** A learning instance consists of:

- A distribution \(D \in \Delta(X)\).
• A target concept $c \in \mathbb{C}$.

Our goal is to have an algorithm $A$ that maps a collection of learning instances to a hypothesis $h : X \to Y$, hopefully with $\Pr_{x \sim D}(h(x) \neq c(x)) \leq \epsilon$.

**Definition 5.6 (Risk).** Given $D \in \Delta(X)$ and target $c \in \mathbb{C}$, the risk of $h$, a function from $X$ to $Y$, is

$$R(h) = \mathbb{E}[\mathbb{1}(h(x) \neq c(x))] = \Pr_{x \sim D}(h(x) \neq c(x))$$

where $R$ depends on $D$ and $c$. This is also known as the generalization error.

**Definition 5.7 (Empirical Risk).** The empirical risk on $x_1, \ldots, x_n$ is defined as

$$\hat{R}(h) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[h(x_i) \neq c(x_i)]$$