11.1 Empirical Risk Minimization

In many machine learning tasks, we have data $Z$ from some distribution $p$ and the task is to minimize the risk:

$$R(f) = E_{Z \sim p}[\ell(f(Z), Z)]$$  \hspace{1cm} (11.1)

where $\ell$ is a loss function of interest, e.g., in classification $Z = (X, Y)$ and we use 0/1 loss $\ell(f(Z), Z) = 1_{f(X) \neq Y}$, in regression $Z = (X, Y)$ and we use squared error $\ell(f(Z), Z) = (f(X) - Y)^2$, and in density estimation $Z = X$ and we use negative log likelihood loss $\ell(f(Z), Z) = -\log f(X)$. We are interested in finding the optimal predictor

$$f^* = \arg \min_f R(f)$$

In practice, we compute the empirical risk:

$$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^n \ell(f(X_i), Y_i)$$  \hspace{1cm} (11.2)

The minimizer of the empirical risk will overfit the data (e.g., a function that is zero everywhere except at $X_i$ it takes the value $Y_i$). Hence, we choose the $\hat{f}$ that minimizes the empirical risk over some class $\mathcal{F}$, such as parametric models, histogram classifiers, decision trees or linear/polynomials functions, etc.

$$\hat{f}_{\text{ERM}} = \arg \min_{f \in \mathcal{F}} \hat{R}(f)$$  \hspace{1cm} (11.3)

To justify this empirical risk minimization (ERM) method, we need to know how similar the $R(f)$ and $\hat{R}(f)$ are. For bounded loss function $\ell \in [0, 1]$, we can apply Hoeffding's concentration inequality which states that for a bounded random variable $W \in [0, 1]$

$$P\left(|\frac{1}{n} \sum_{i=1}^n W_i - W| \geq \epsilon\right) \leq 2e^{-2n\epsilon^2}$$

Equivalently, if $\ell$ is bounded (e.g., 0/1 loss function in classification), then with probability at least $1 - \delta(f)$, we have the following bound for a given $f \in \mathcal{F}$:

$$R(f) \leq \hat{R}(f) + \sqrt{\frac{\log \frac{2}{\delta(f)}}{2n}}$$  \hspace{1cm} (11.4)
When $\mathcal{F}$ is finite, we can take $\delta(f) = \frac{1}{\left|\mathcal{F}\right|}$. Then, with probability $\geq 1 - \delta$,

$$R(f) \leq \hat{R}(f) + \sqrt{\frac{\log |\mathcal{F}| + \log \frac{2}{\delta}}{2n}}, \forall f \in \mathcal{F} \quad (11.5)$$

where $\log |\mathcal{F}|$ can be considered as the number of bits needed to encode $\mathcal{F}$. However, we can obtain a tighter bound using prefix codes. This would also be helpful when dealing with countably infinite classes $\mathcal{F}$. Later, we will see how to handle continuous classes.

When $\mathcal{F}$ is finite or countably infinite, we can use the prefix code. Let $c(f)$ denote the prefix code length for encoding $f$. A more complex model needs longer code to encode, we will see some examples in a little bit. According to Kraft’s inequality,

$$\sum_{f \in \mathcal{F}} D^{-c(f)} \leq 1 \quad (11.6)$$

when using $D$-ary code. We can take $\delta(f) = \delta D^{-c(f)}$. Then we get the bound, with probability $\geq 1 - \delta$, for all $f \in \mathcal{F}$,

$$R(f) \leq \hat{R}(f) + \sqrt{\frac{c(f) + \log \frac{2}{\delta}}{2n}} \quad (11.7)$$

Here log would be to base $D$ (we will drop the $D$ in the notation to keep things simple). It implies the empirical risk is closer to true risk when $f$ is simpler (needs fewer bits to encode).

To bound the risk of the Empirical Risk Minimizer $\hat{f}^{ERM}$, notice that with probability $\geq 1 - \delta$

$$R(\hat{f}^{ERM}) \leq \hat{R}(\hat{f}^{ERM}) + \sqrt{\frac{c(\hat{f}^{ERM}) + \log \frac{2}{\delta}}{2n}} \quad (11.8)$$

$$\leq \hat{R}(f) + \sqrt{\frac{c(\hat{f}^{ERM}) + \log \frac{2}{\delta}}{2n}} \quad (11.9)$$

for all $f \in \mathcal{F}$.

Now let $\Omega$ denote the event on which we have bounded the deviation of true and empirical risks for all $f \in \mathcal{F}$, which means $P(\Omega) \geq 1 - \delta$. Also, let $\Omega^c$ denote the complement set of $\Omega$, then the expectation of $R(\hat{f}^{ERM})$ is also bounded as follows.

$$E[R(\hat{f}^{ERM})] \leq E[R(\hat{f}^{ERM})|\Omega]P(\Omega) + E[R(\hat{f}^{ERM})|\Omega^c]P(\Omega^c) \quad (11.10)$$

Since

$$P(\Omega) \leq 1$$

$$P(\Omega^c) \leq \delta$$

and

$$E[R(\hat{f}^{ERM})|\Omega^c] \leq 1 \text{ (loss function is bounded by 1)}$$

we have

$$E[R(\hat{f}^{ERM})] \leq E[R(\hat{f}^{ERM})|\Omega] + \delta \quad (11.11)$$

$$\leq R(f) + E[\sqrt{\frac{c(\hat{f}^{ERM}) + \log \frac{2}{\delta}}{2n}}] + \delta \quad (11.12)$$

for all $f \in \mathcal{F}$. Thus, we get

$$E[\hat{R}(f)^{ERM}] - R^* \leq \min_{f \in \mathcal{F}} R(f) - R^* + E[\sqrt{\frac{c(\hat{f}^{ERM}) + \log \frac{2}{\delta}}{2n}}] + \delta \quad (11.13)$$
11.2 Complexity Regularized Empirical Risk Minimization aka Structural Risk Minimization

To achieve better estimation of the true risk, we should minimize both the empirical risk and complexity, instead of only minimizing the empirical risk.

\[ \hat{f}_{SRM} = \arg \min_{f \in F} \{ \hat{R}(f) + \epsilon(f) \} \]  

(11.14)

where \( \epsilon(f) = \sqrt{\frac{\alpha(f) + \log \frac{2}{\delta}}{2n}} \). With probability \( \geq 1 - \delta \), we have the following bound on \( R(\hat{f}) \)

\[ R(\hat{f}_{SRM}) \leq \hat{R}(\hat{f}_{SRM}) + \epsilon(\hat{f}_{SRM}) \]  

(11.15)

\[ \leq \hat{R}(f) + \epsilon(f), \forall f \in \mathcal{F} \]  

(11.16)

Proceeding as before, we get

\[ E[R(\hat{f}_{SRM})] \leq R(f) + \epsilon(f) + \delta \]  

(11.17)

Since this is true for all \( f \in \mathcal{F} \), we get the following which states that the complexity penalized ERM balances both the risk and complexity.

\[ E[R(\hat{f}_{SRM})] - R^* \leq \min_{f \in \mathcal{F}} \{ R(f) - R^* + \epsilon(f) \} + \delta \]  

(11.18)

for any \( \delta \in (0, 1) \). Here \( R(f) - R^* \) is the approximation error which tells us how well a function approximates the optimal predictor, and \( \epsilon(f) \) the estimation error which tells us how the true and empirical risks deviate for \( f \). Typically, the simpler \( f \) is, the smaller the estimation error but the larger the approximation error. The complexity penalized ERM picks a function that balances these two errors.

In comparison, the ERM bound looks like

\[ E[R(\hat{f}_{ERM})] - R^* \leq \min_{f \in \mathcal{F}} \{ R(f) - R^* \} + E[\epsilon(\hat{f}_{ERM})] + \delta \]  

(11.19)

which is typically worse since the predictor does not balance approximation and estimation error.

Next, we will construct prefix codes for some classification models and apply the resulting bounds to 0/1 loss (which is bounded by 1).

11.3 Histogram Classifiers

The histogram classifier, shown in Figure 11.2, is similar to histogram density estimation (which you might be more familiar with). It divides the domain into bins. For the points inside each bin, it assigns a label. For example, it can use the majority vote of the data points to decide the label in each bin.

Let \( \mathcal{F}_m \) denote the class of histogram classifiers with \( m \) bins. Suppose there are two classes 0 and 1. Then \( |\mathcal{F}_m| = 2^m \). To encode \( 2^m \) different classifiers, we need \( \log 2^m = m \) bits using uniform coding - notice that this is a prefix code. In addition, we need to encode the integer \( m \) denoting the histogram resolution, which needs \( \log m \) bits. Appending these bits to the uniform code still leads to a prefix code. Therefore, we need \( c(f) = O(m) \) bits to encode \( f \in \mathcal{F}_m \).

\[ \text{We will consider the countably infinite class } \mathcal{F} = \bigcup_m \mathcal{F}_m. \]

\[ ^1 \text{Though a better code can result in tighter bounds, such an encoding is rate optimal (gets best possible dependence of error on number of samples) and we will only care about right order of number of bits and resulting error.} \]
Figure 11.1: Histogram Classifier

Figure 11.2: Tree Classifier
Plugging this in the earlier expression, we have the complexity penalized ERM or SRM classifier.

\[
\hat{f}^{SRM} = \arg \min_{f \in \mathcal{F}} \{ \hat{R}(f) + \sqrt{\frac{O(m) + \log \frac{2}{\delta}}{2n}} \} = \arg \min_{m} \{ \min_{f \in \mathcal{F}_m} \hat{R}(f) + \sqrt{\frac{O(m) + \log \frac{2}{\delta}}{2n}} \}
\] (11.20)

The error bound is

\[
E[R(\hat{f}^{SRM})] \leq \min_{m} \{ \min_{f \in \mathcal{F}_m} R(f) + \sqrt{\frac{O(m) + \log \frac{2}{\delta}}{2n}} \} + \delta
\] (11.21)

Thus, the complexity penalized procedure also performs model selection (pick the best \(m\)) for histogram classifiers automatically.

### 11.4 Decision Trees Classifiers

Decision trees classifiers are related to histogram classifiers. Rather than only using uniform bin size in histogram classifiers, decision trees classifiers allow different bin sizes.

For example, the axis parallel tree splits the space in the axis parallel way - alternating between vertical and horizontal splits. Here we only consider the dyadic split, which means the split can only happen at the middle value of a bin. Figure 11.4 shows an example of recursive dyadic partition/tree growing.

Now we need to encode such dyadic decision trees classifiers. In specific, we need to encode the structure of the tree and the label on the leaves. Let \(\mathcal{F}_K\) denotes the class of decision trees with \(K\) leaves. We will do SRM over the countably infinite class \(\bigcup K \mathcal{F}_K\).
For encoding tree structure, we can assign 0 for internal nodes and 1 for leaves. Then we traverse the tree top-to-bottom and in left-to-right order to produce the code. An example is shown in Figure 11.4. So the number of bits needed to encode the tree structure is equal to the number of nodes. For a tree with \( K \) leaves, there are \( 2K - 1 \) nodes. Hence we need \( 2K - 1 \) bits to encode the tree structure. Notice that this is a prefix code. In order to encode the labels on the \( K \) leaves, we can use \( K \) bits and append it to the tree structure code, still resulting in a prefix code. In sum, we need \( 3K - 1 \) bits to encode decision tree classifiers.

Then the complexity penalized ERM of decision trees is

\[
\hat{f}_{SRM} = \min_{f \in F} \left\{ \hat{R}(f) + \sqrt{\frac{3K - 1 + \log \frac{2}{3}}{2n}} \right\} = \min_k \left\{ \min_{f \in \mathcal{F}_k} \hat{R}(f) + \sqrt{\frac{3K - 1 + \log \frac{2}{3}}{2n}} \right\}
\]

The error bound is

\[
E[R(\hat{f}_{SRM})] \leq \min_k \left\{ \min_{f \in \mathcal{F}_k} R(f) + \sqrt{\frac{3K - 1 + \log \frac{2}{3}}{2n}} \right\} + \delta
\]

Thus, the complexity penalized procedure also performs model selection (pick the best \( K \)) for decision tree classifiers automatically.

### 11.5 Comparison of Histogram and Decision Trees Classifiers

We consider a bounded domain \([0, 1]^d\). Let assume the true decision boundary \( b(x) \) is a function of \( d - 1 \) out of the \( d \) coordinates and \( b(x) \) is Lipschitz where \( x \in [0, 1]^{d-1} \). It means \( |b(x) - b(x')| \leq L||x - x'|| \) for some constant \( L > 0 \). For simplicity, let's consider \( d = 2 \).

With \( d = 2 \), in order to get the same approximation error as a \( m \) bin histogram, for a 1-dimensional decision boundary, a decision tree needs \( K = \tilde{O}(\sqrt{m}) \) bins. Here we give a brief proof.

For histogram, the most number of bins that the boundary can intersect is \( L\sqrt{m} \) since there are \( \sqrt{m} \) bins along any coordinate. Let's now think of the smallest that \( R(f) - R^* = P(f(X) \neq f(X)) \) can be using a \( m \)-bin histogram. The best histogram classifier with \( m \) bins can incur at most \( O(1) \) error in the bins which intersect the boundary and the probability of a test point falling in one such bin is \( 1/m \). Therefore,

\[
\inf_{f \in \mathcal{F}_m} R(f) - R^* \leq L\sqrt{m} \frac{1}{m} O(1) = O\left(\frac{1}{\sqrt{m}}\right)
\]

So for histogram, the excess risk bound scales as (only keeping correct order dependence on \( m \))

\[
\min_m \frac{1}{\sqrt{m}} + \sqrt{\frac{m + \log \frac{2}{3}}{2n}} \approx n^{-\frac{1}{4}}
\]

where the optimum number of bins balancing approximation and estimation error is \( m \approx \sqrt{n} \).

For decision tree with same resolution finest leaves i.e., the smallest leaves have size same as a histogram bin with \( m \) bins, the most number of finest resolution leaves that can intersect the boundary is \( L\sqrt{m} \). Now there are other leaves that may not intersect the boundary, but it can be shown that the total number of leaves of such a decision tree \( K \leq 8L\sqrt{m} \). (A crude way to see a slightly looser bound is to realize that for

\[\text{Though a better code can result in tighter bounds, such an encoding is rate optimal (gets best possible dependence of error on number of samples) and we will only care about right order of number of bits and resulting error.}\]

\[\text{\tilde{O} absorbs log dependence in addition to constant factor dependence}\]
every leaf that intersects the boundary, there are 4 leaves at each level that may need to be present in the
tree, giving us $4L\sqrt{m}\log m$ bound on $K$ as there are $\log m$ levels in a tree with finest resolution leaves of
volume $1/m$. This can be improved by realizing that some of the leaves intersecting the boundary need to
be co-located and hence share sibling leaves at higher levels. We skip the details.) So we have $K = \tilde{O}(\sqrt{m})$.
Therefore,
\[ \inf_{f \in \mathcal{F}_K} R(f) - R^* = L\sqrt{m}\frac{1}{m}O(1) = O\left(\frac{1}{\sqrt{m}}\right) = \tilde{O}\left(\frac{1}{K}\right) \quad (11.25) \]

For decision trees, the excess risk bound scales as (keeping correct order dependence on $K$ only)
\[ \min_K \frac{1}{K} + \sqrt{\frac{K + \log \frac{2}{3}}{2n}} \asymp n^{-\frac{1}{3}} \]

where the optimum number of leaves balancing approximation and estimation error is $K \asymp n^{1/3}$.

Thus, decision tree classifiers have error that converges faster with number of samples than histogram
classifiers for well-behaved (d-1 dimensional) boundaries, which is usually the case. However, if the decision
boundary is very complicated, essentially passing through all histogram bins, then the decision tree classifier
with same approximation properties will need $K \asymp m$. As is intuitive, in this case, the decision tree classifiers
are no better than histogram classifiers.

Remark: It turns out that the minimax rate of error convergence for $d-1$ dimensional boundaries is $n^{-1/d}$
(for $d = 2$ this is $n^{-1/2}$) and decision trees are actually capable of achieving that fast rate if the complexity of
a tree is encoded not just in terms of the number of leaves, but also takes into account the volume/resolution
of the leaves. If interested, see [http://papers.nips.cc/paper/2364-near-minimax-optimal-classification-with-
dyadic-classification-trees.pdf](http://papers.nips.cc/paper/2364-near-minimax-optimal-classification-with-
dyadic-classification-trees.pdf)