STAT 460/560 Class 13: Empirical Risk Minimization and Cross-Validation

Ben Bloem-Reddy

Reading: Chapter 13.6-7, [Was04], Chapter 7 [HTF09].

1. Prediction and empirical risk minimization

When the task is parameter estimation, risk is typically a purely theoretical construct: the risk usually depends on an unknown parameter and therefore cannot be estimated from data. However, consider the problem of predicting Y from X with a function $\hat{Y} = f(X)$. The risk is

$$R(P, f) = \mathbb{E}_P[L(Y, f(X))] = \int L(y, f(x)) dP(x, y) \; . \tag{13.1}$$

Activity 13.1. Suppose we have $\mathbb{R}$-valued pairs $((X_1, Y_1), \ldots, (X_n, Y_n)) \sim_{\text{IID}} P$, and we're using squared error loss,

$$L(Y, f(X)) = \|f(X) - Y\|^2$$

Show that the function that achieves the smallest risk is

$$f(x) = \mathbb{E}_P[Y | X = x] \; . \tag{13.2}$$

Exercise 13.1. Suppose that $Y \in \{0, 1\}$, $((X_1, Y_1), \ldots, (X_n, Y_n)) \sim_{\text{IID}} P$, and we're using 0-1 loss,

$$L(Y, f(X)) = \mathbf{I}\{Y \neq f(X)\} \; . \tag{13.3}$$

Show that the function that achieves the smallest risk is

$$f(x) = \begin{cases} 1 & \text{if } P(Y = 1 | X = x) \geq \frac{1}{2} \\ 0 & \text{o.w.} \end{cases} \tag{13.4}$$

This is called the **Bayes classifier**.

We can use the risk in a couple of ways. First, we might be interested in calculating the risk of a fixed prediction function f. The risk depends on the unknown distribution P, so we need some ways to estimate it. The risk can be estimated by the **empirical risk**

$$\hat{R}_n(f) := R(\hat{P}_n, f) = \frac{1}{n} \sum_{i=1}^{n} L(Y_i, f(X_i)) \; . \tag{13.5}$$

Minimizing this type of objective as a way to estimate f from data is called **empirical risk minimization**:

$$\hat{f}_n = \arg\min_{f \in \mathcal{F}} \hat{R}_n(f) \; .$$

Setting some $\mathcal{F}$ (e.g., a neural network) is the default procedure for most prediction-oriented machine learning methods (though its implementation for different models and massive datasets is fairly complicated).

Having used a *training set* of observations $\mathcal{T}_n := ((X_1, Y_1), \ldots, (X_n, Y_n)) \sim_{\text{IID}} P$ to estimate $\hat{f}$, subsequently estimating the risk can be somewhat tricky. We might consider using the empirical risk. However, as we will show in a moment, this uses the training data twice (once to estimate $\hat{f}_n$ and once to estimate $R(P, \hat{f}_n)$), leading to risk estimates that are biased downward.

Alternatively, for a fixed set of observations $\mathcal{T}_n$ used to estimate $\hat{f}_n$, we might ask how our model generalizes to new (i.e., independent) data, $(X^*, Y^*) \sim P$. This is the **generalization risk** (or generalization error),

$$R(P, \hat{f}|\mathcal{T}_n) = \mathbb{E}_P[L(Y^*, \hat{f}_n(X^*)) \mid \mathcal{T}_n] \; . \tag{13.6}$$

(There's a subtle but important difference in notation between $\hat{f}$ and $\hat{f}_n$. The first can be thought of as a process or algorithm for estimating the function, but that does not depend on any particular data. The second is actually a function of the training data.) The difference between the generalization risk and the risk in (13.1) is that the generalization risk depends on the training data $\mathcal{T}_n$, and the risk further averages over the training data:

$$R(P, \hat{f}) = \mathbb{E}[R(P, \hat{f}|\mathcal{T}_n)] \; ,$$

where the expectation is over $\mathcal{T}_n$.

How do generalization risk and empirical risk compare? Let's analyze squared error loss by conditioning on the values of the X_i's, and consider using $\hat{f}$ to predict new independent outcomes, Y_i^*.

Activity 13.2. Let (X_i^*, Y_i^*) be a new observation, conditioned such that $X_i^* = X_i = x_i$. Show that

$$\mathbb{E}[(\hat{f}_n(X_i) - Y_i)^2 \mid X_i^* = X_i = x_i] - \mathbb{E}[(\hat{f}_n(X_i^*) - Y_i^*)^2 \mid X_i^* = X_i = x_i] = -2\text{Cov}[\hat{f}_n(X_i), Y_i \mid X_i = x_i] \; .$$

Hence,

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[L(Y_i, \hat{f}(X_i))|X_i] - \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[L(Y_i^*, \hat{f}(X_i^*))|X_i, X_i^* = X_i] = -\frac{2}{n}\sum_{i=1}^{n} \text{Cov}[\hat{f}_n(X_i), Y_i \mid X_i]$$

and

$$\mathbb{E}[\hat{R}_n(\hat{f}_n)] - R(P, \hat{f}) = -\frac{2}{n}\sum_{i=1}^{n} \mathbb{E}\left[\text{Cov}[\hat{f}_n(X_i), Y_i \mid X_i]\right] \tag{13.7}$$

The punchline is that the empirical risk is a downward-biased estimator of the risk, and the amount of bias is increasing in the covariance between each Y_i and $\hat{f}_n(X_i)$. The correlation here comes through the fact that Y_i is in the training data that was used to estimate $\hat{f}_n$. The more flexible our model is—the closer it is to being able to perfectly predict each training observation—the more the empirical risk underestimates the risk.

To sum up: The empirical risk tends to underestimate the risk because it's using the data twice—once to estimate $\hat{f}_n$ by minimizing empirical risk, then again to estimate the risk. As a rule of thumb, the more flexible (or complex) a model is, the greater the risk of *over-fitting*, which means that the model fits the training data well but does not generalize well.[1]

2. Cross-validation

There are various methods for trying to get around the bias of the empirical risk. Wasserman [Was04] has a few (AIC, BIC, Mallow's C_p) that use heuristics or additional assumptions to estimate the covariance term, which then generalize less heuristically (but less specifically) into strategies that add a model complexity penalty to the loss function. We will return to this idea later in the term.

[1] There is a big open caveat to this rule of thumb when it comes to so-called *over-parameterized* models, i.e., models with more parameters than training data observations. This is the realm in which many deep learning models operate, and understanding the statistical behavior of such models is an active area of research.

A general approach to unbiased risk estimation that trades computation and data for assumptions is **data-splitting**. The basic idea is to set aside part of your data, often called **test data**, so that your full data is split into training and test sets, $\mathcal{T}_n$ and $\mathcal{U}_m = ((X_1^*, Y_1^*), \ldots, (X_m^*, Y_m^*))$, respectively. Using the training set $\mathcal{T}_n$ to estimate $\hat{f}_n$, the test set is used to estimate the generalization error,

$$R(\hat{P}_m, \hat{f}|\mathcal{T}_n) = \frac{1}{m}\sum_{j=1}^{m} L(Y_j^*, \hat{f}_n(X_j^*)) \; . \tag{13.8}$$

If the test/train split is random, it's not hard to see that this is an unbiased estimator of the risk.

Exercise 13.2. Show that the estimator in (13.8) is an unbiased estimator of the risk.

There's a trade-off here: ultimately, we're splitting the data to estimate two expectations: the training data is used to estimate the empirical risk as a way to estimate $\hat{f}_n$; the testing data is used to estimate the risk of $\hat{f}_n$. If we set n to be too small, we will under-fit, yielding a model with large prediction bias. On the other hand, if we set m too small, we will get a high-variance estimate of the risk. For a data set of fixed size N, how do we choose n and m subject to $n + m = N$?

One extreme is to set $m = 1$, so that almost all of the observations go into estimating $\hat{f}$. The resulting generalization risk estimator will have high variance. A method for reducing that variance, at the cost of incurring some bias in the risk estimation, is **leave-one-out cross validation** (LOOCV). The basic idea is to fit N copies of the model, each time leaving out one observation. For example, suppose we leave out (X_i, Y_i), estimate $\hat{f}_{(i)}$ on the remaining observations, and then predict $\hat{Y}_{(i)} = \hat{f}_{(i)}(X_i)$. The LOOCV error is

$$\hat{R}_{\text{LOOCV}}(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n} L(Y_i, \hat{Y}_{(i)}) \; . \tag{13.9}$$

In general, LOOCV is computationally expensive; a less extreme approach is **k-fold cross-validation** (kCV), in which the observations are randomly subsetted into k roughly equal size "folds", then each fold is held out as the test (or validation) set. For example, if $k = 5$, five different risk estimates are averaged to obtain a final risk estimate. Section 7.10 of [HTF09] has a nice discussion of practical issues related to using cross-validation for model selection.

Exercise 13.3. Implement k-fold cross-validation using a prediction method of your choice (e.g., linear or logistic regression, random forest, neural network).

References

[HTF09] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer New York, 2009. URL: https://hastie.su.domains/ElemStatLearn/.

[Was04] L. Wasserman. *All of Statistics*. Springer New York, 2004.