

LECTURE 1: INTRODUCTION TO SUPERVISED LEARNING

In this lecture we formulate the basic (supervised) learning problem and introduce several key concepts including loss function, risk and error decomposition.

1 Basic Concepts

We use \mathcal{X} and \mathcal{Y} to denote the *input space* and the *output space*, where typically we have $\mathcal{X} = \mathbb{R}^p$. A joint probability distribution on $\mathcal{X} \times \mathcal{Y}$ is denoted as $P_{X,Y}$. Let (X, Y) be a pair of random variables distributed according to $P_{X,Y}$. We also use P_X and $P_{Y|X}$ to denote the marginal distribution of X and the conditional distribution of Y given X .

Let $\mathcal{D}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ be an i.i.d. random sample from $P_{X,Y}$. The goal of *supervised learning* is to find a mapping $h : \mathcal{X} \mapsto \mathcal{Y}$ based on \mathcal{D}_n so that $h(X)$ is a good approximation of Y . When $\mathcal{Y} = \mathbb{R}$ the learning problem is often called *regression* and when $\mathcal{Y} = \{0, 1\}$ or $\{-1, 1\}$ it is often called (binary) *classification*.

The dataset \mathcal{D}_n is often called the *training set* (or *training data*), and it is important since the distribution $P_{X,Y}$ is usually unknown. A *learning algorithm* is a procedure \mathcal{A} which takes the training set \mathcal{D}_n and produces a predictor $\hat{h} = \mathcal{A}(\mathcal{D}_n)$ as the output. Typically the learning algorithm will search over a space of functions \mathcal{H} , which we call the *hypothesis space*.

2 Loss Function and Risk

A *loss function* is a mapping $\ell : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}^+$ (sometimes $\mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}^+$). For example, in binary classification the 0/1 loss function $\ell(y, p) = I(y \neq p)$ is often used and in regression the *squared error loss* function $\ell(y, p) = (y - p)^2$ is often used. Other loss functions include the following: *absolute loss*, *Huber loss*, ϵ -*insensitive loss*, *hinge loss*, *logistic loss*, *exponential loss*, *modified least squares loss*, etc. They will be discussed later in more details.

The performance of a predictor $h : \mathcal{X} \mapsto \mathcal{Y}$ is measured by the expected loss, a.k.a. the *risk* or *generalization error*:

$$R(h) := \mathbb{E}_{X,Y}[\ell(Y, h(X))],$$

where the expectation is taken with respect to the distribution $P_{X,Y}$. Since in practice we estimate \hat{h} based on the training set \mathcal{D}_n , we have $R(\hat{h})$ itself a random variable. Thus we may also use the quantity $R(\mathcal{A}) = \mathbb{E}_{\mathcal{D}_n}[R(\hat{h})]$ to characterize the generalization performance of the learning algorithm \mathcal{A} , which is also called the *expected risk* of the learning algorithm.

The risk is an important measure of the goodness of the predictor $h(\cdot)$ since it tells how it performs on average in terms of the loss function $\ell(\cdot, \cdot)$. The minimum risk is defined as

$$R^* = \inf_h R(h)$$

where the infimum is often taken with respect to all measurable functions. The performance of a given predictor/estimator can be evaluated by how close $R(h)$ is to R^* . Minimization of the risk is non-trivial because the underlying distribution $P_{X,Y}$ is in general unknown, and the training data \mathcal{D}_n only gives us an incomplete knowledge of $P_{X,Y}$ in practice.

2.1 Binary Classification

For classification problem, a predictor h is also called a *classifier*, and the loss function for binary classification is often taken to be the 0/1 loss. In this case, we have

$$R(h) = \mathbb{E}_{X,Y}[\ell(Y, f(X))] = \mathbb{E}_{X,Y}[I(Y \neq f(X))] = P(f(X) \neq Y).$$

And the infimum risk R^* is also known as the *Bayes risk*.

The following result shows that the *Bayes classifier*, which is defined as

$$h^*(x) = \begin{cases} 1, & P(Y = 1|X = x) \geq 1/2 \\ 0, & \text{otherwise} \end{cases},$$

can achieve the Bayes risk.

Theorem 1-1. For any classifier h we have $R(h) \geq R(h^*)$, i.e. $R(h^*) = R^*$.

Proof.

We will show that $P(h(X) \neq Y) \geq P(h^*(X) \neq Y)$. For any $X = x$, we have

$$\begin{aligned} P(h(X) \neq Y|X = x) &= 1 - P(h(X) = Y|X = x) \\ &= 1 - (P(h(X) = 1, Y = 1|X = x) + P(h(X) = 0, Y = 0|X = x)) \\ &= 1 - (\mathbb{E}[I(h(X) = 1)I(Y = 1)|X = x] + \mathbb{E}[I(h(X) = 0)I(Y = 0)|X = x]) \\ &= 1 - (I(h(x) = 1)\mathbb{E}[I(Y = 1)|X = x] + I(h(x) = 0)\mathbb{E}[I(Y = 0)|X = x]) \\ &= 1 - I(h(x) = 1)P(Y = 1|X = x) - I(h(x) = 0)P(Y = 0|X = x). \end{aligned}$$

Letting $\eta(x) := P(Y = 1|X = x)$, we have for any $X = x$,

$$\begin{aligned} &P(h(X) \neq Y|X = x) - P(h^*(X) \neq Y|X = x) \\ &= P(h^*(X) = Y|X = x) - I(h(x) = 1)\eta(x) - I(h(x) = 0)(1 - \eta(x)) \\ &= \eta(x)[I(h^*(x) = 1) - I(h(x) = 1)] + (1 - \eta(x))[I(h^*(x) = 0) - I(h(x) = 0)] \\ &= (2\eta(x) - 1)[I(h^*(x) = 1) - I(h(x) = 1)] \\ &\geq 0 \end{aligned}$$

where the last inequality is true by the definition of $h^*(x)$. The result follows by intergrating both sides with respect to x .

□

2.2 Regression

In regression we typically have $\mathcal{X} = \mathbb{R}^p$ and $\mathcal{Y} = \mathbb{R}$. And the risk is often measured by the squared error loss, $\ell(p, y) = (p - y)^2$. The following result shows that for squared error regression, the optimal predictor is the conditional mean function $\mathbb{E}[Y|X = x]$.

Theorem 1-2. Suppose the loss function $\ell(\cdot, \cdot)$ is the squared error loss. Let $h^*(x) = \mathbb{E}[Y|X = x]$, then we have $R(h^*) = R^*$.

The proof will be left as an exercise. Thus regression with squared error can be thought as trying to estimate the conditional mean function. How about regression with its risk defined by the absolute error loss function?

3 Approximation Error vs. Estimation Error

Suppose that the learning algorithm chooses the predictor from the hypothesis space \mathcal{H} , and define

$$h^* = \operatorname{arg\,inf}_{h \in \mathcal{H}} R(h),$$

i.e. h^* is the best predictor among \mathcal{H} ¹. Then the *excess risk* of the output \hat{h}_n of the learning algorithm is defined and can be decomposed as follows²:

$$R(\hat{h}_n) - R^* = \underbrace{\left(R(h^*) - R^* \right)}_{\text{approximation error}} + \underbrace{\left(R(\hat{h}_n) - R(h^*) \right)}_{\text{estimation error}}$$

Such a decomposition reflects a trade-off similar to the bias-variance tradeoff (maybe slightly more general). The *approximation error* is deterministic and is caused by the restriction of using \mathcal{H} . The *estimation error* is caused by the usage of a finite sample that cannot completely represent the underlying distribution.

The approximation error term behaves like a bias square term, and the estimation error behaves like the variance term in standard statistical estimation problems. Similar to the bias-variance trade-off, there is also a trade-off between the approximation error and the estimation error. Basically if \mathcal{H} is large then we have a small approximation error but a relatively large estimation error and vice versa.

¹Sometimes h^* is defined as the approximately best predictor chosen by the estimation procedure if infinite amount of samples are given. In that case, the approximation error can be caused by both the function class \mathcal{H} and some intrinsic bias of the estimation procedure.

²Sometimes the decomposition is done for $\mathbb{E}_{\mathcal{D}_n}[R(\hat{h}_n)] - R^*$.