6.7900 Fall 2024: Lecture Notes 1

Revision: 9/19/24 2:48PM

There was a lot of logistical information which you can find at `gradml.mit.edu`.

1 What is machine learning?

Broadly, machine learning is a set of methods for *using data* to *become better at a task.*

Problem formalization

- Input space : $\mathcal{X}$ is often (but not necessarily) a vector space, often $\mathbb{R}^d$
- Output space : $\mathcal{Y}$ is a fixed finite discrete set, for *classification* problems; it could be a continuous set, such as $\mathbb{R}^d$ in a *regression* problem
- Training set : $\{(x^{(n)}, y^{(n)})\}_{n=1}^N$: set of N pairs of values, where $x^{(n)} \in \mathcal{X}$ and $y^{(n)} \in \mathcal{Y}$
- Decision rule : a function $h : \mathcal{X} \to \mathcal{Y}$.

Called h for "hypothesis."

- Loss function : $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. This function is potentially *asymmetric*, so that $L(a, g)$ is the loss (badness) of predicting or guessing g when the actual label is a.

Example problem: spam detection Our *data* is a set of pairs, of an email message and a binary value describing whether that message is spam. Our *task* is to predict whether future email messages are or are not spam. We represent each email message as a vector of *features*, $x^{(n)}$, for example with each $x_i^{(n)}$ having value 1 if some word i occurs in the nth message and 0 otherwise. The associated *label* $y^{(n)} \in \{0, 1\}$ indicates not-spam or spam.

What should our loss function be? Maybe something like:

$$L(a,g) = \begin{cases} 0 & \text{if } a = g \\ 1 & \text{if } a = 1 \text{ and } g = 0 \\ 15 & \text{if } a = 0 \text{ and } g = 1 \end{cases}$$

This encodes the idea that it's worse to say a message is spam when it is not (because we might miss an important message), than to say it is not spam when it is (because it is a mild annoyance).

Supervised learning problem: Given training set $\{(x^{(n)}, y^{(n)})\}_{n=1}^{N}$, find a decision rule $h : \mathcal{X} \to \mathcal{Y}$ that *performs well*.

To "perform well" is to make predictions on as-yet-unseen inputs that have low loss. That is, that $L(y^{(N+1)}, h(x^{(N+1)}))$ is low. But we don't know what these inputs will be! Probability to the rescue!

Probability is the language of uncertainty.

Assume training data and future data are drawn from some distribution $p(x, y)$. Now we can define *risk* of h to be the expected loss of using h to make a prediction on a new example:

$$\text{risk}(h) = \mathbb{E}[L(Y, h(X))]$$

Exercise: Show that, if $\{(X^{(N+m)}, Y^{(N+m)})\}_{m=1}^{M}$ are independent and identically distributed, the average expected loss of using h to make predictions on the next M points is equal to $\text{risk}(h)$.

Optimal h for classification We'd love to pick h to minimize risk $\mathbb{E}[L(Y, h(X))]$. If we know $p(X, Y)$, then we can!

Proposition: Consider K-class classification, in which $\mathcal{Y} = \{1, \ldots K\}$, and suppose X is a discrete random variable, Y is a random variable with domain $\mathcal{Y}$, and we know $p(x, y)$. Then the rule

$$h(x) = \arg\min_k \sum_{j=1}^{K} L(j,k) p(y = j \mid x)$$

minimizes $\mathbb{E}[L(Y, h(X))]$.

Proof. We start by expanding and rearranging terms in the risk definition:

$$\begin{aligned}
\mathbb{E}[L(Y, h(X))] &= \sum_{x,y \in \mathcal{X} \times \mathcal{Y}} L(y, h(x)) p(x, y) \\
&= \sum_{j=1}^{K} \sum_{x \in \mathcal{X}} L(j, h(x)) p(x, y = j) \\
&= \sum_{j=1}^{K} \sum_{x \in \mathcal{X}} L(j, h(x)) p(y = j \mid x) p(x) \\
&= \sum_{x \in \mathcal{X}} p(x) \sum_{j=1}^{K} L(j, h(x)) p(y = j \mid x)
\end{aligned}$$

Now observe that this sum over x can be minimized by independently minimizing the j for each x, and that $h(x) = \arg\min_k \sum_{j=1}^{K} L(j, k) p(y = j \mid x)$ is exactly the h that does so. $\square$

Exercise: State and prove a similar result for continuous $\mathcal{X}$.

Two-class decision rule So, if we have two classes 0 and 1, and known $p(x, y)$, what is the form of the optimal h? (Remember that the loss function is $L(a, g)$ where a is the true value of Y and g is the "guess" we are proposing to make.)

$$h(x) = \begin{cases} 0 & \text{if } \mathbb{E}[L(Y, 0)] < \mathbb{E}[L(Y, 1)] \\ 1 & \text{otherwise} \end{cases}$$

which can be rewritten as

$$\begin{aligned}
h(x) &= \begin{cases} 0 & \text{if } L(0,0)p(X = x, Y = 0) + L(1,0)p(X = x, Y = 1) < L(0,1)p(X = x, Y = 0) + L(1,1)p(X = x, Y = 1) \\ 1 & \text{otherwise} \end{cases} \\
&= \begin{cases} 0 & \text{if } L(0,0)p(Y = 0 \mid X = x) + L(1,0)p(Y = 1 \mid X = x) < L(0,1)p(Y = 0 \mid X = x) + L(1,1)p(Y = 1 \mid X = x) \\ 1 & \text{otherwise} \end{cases} \\
&= \begin{cases} 0 & \text{if } L(1,0)p(Y = 1 \mid X = x) < L(0,1)p(Y = 0 \mid X = x) \\ 1 & \text{otherwise} \end{cases}
\end{aligned}$$

Exercise: Why was it okay to turn $p(X = x, Y = 1)$ into $p(Y = 1 \mid X = x)$ in the derivation above?

Exercise: What happened to $L(0, 0)$ and $L(1, 1)$? Where would they go in the expression above? Why was it okay to omit them?

Optimal h for regression Now let's consider a regression problem where $\mathcal{Y} = \mathbb{R}$. If we know $p(x, y)$ we can also determine the optimal (minimum risk) h!

Proposition: Assume $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = \mathbb{R}$, and we know $p(x, y)$. If we are using *squared loss* $L(a, g) = (a - g)^2$, then the decision rule

$$h(x) = \mathbb{E}[Y \mid X = x]$$

minimizes $\mathbb{E}[L(Y, h(X))]$.

Exercise: Complete the proof! It may be useful to make this (common, but counter-intuitive) move at some point:

$$y - h(x) = y - \mathbb{E}[Y \mid X = x] + \mathbb{E}[Y \mid X = x] - h(x)$$

Pretty cool! It says that the best prediction, under squared loss, is the conditional mean of Y given $X = x$.

Major obstacle We don't know $p(x, y)$!

But we do have training data! So, let's use it in the process of picking h. It can only be helpful if it is related, somehow, to the data that we are going to have to make predictions about in the future. A typical, strong assumption is that *all data*, including training data and future data, are independent and identically distributed.

Given this relationship between past and future data, we might think about approximating the *risk* by the *empirical risk* (average loss on the training data), observing that

$$\mathbb{E}[L(Y, h(X))] \approx \frac{1}{N} \sum_{n=1}^{N} L(y^{(n)}, h(x^{(n)}))$$

Exercise: Will this be the key to solving all our problems in machine learning?