Supervised Learning

Supervised learning is the macIhine learning setup that involves a dataset organized into pairs of x and y values: $D_n = \{(x^1, y^1), ..., (x^n, y^n)\}$. Some mapping must be learned between the x and y values so that given a new x encountered in the future, a computer can accurately predict the corresponding y. For example, x values might be a patient's vital signs and y values might be whether or not that patient is having a heart attack.

x values are vectors in d dimensions: $x^{(i)} \in \mathbb{R}^d$. The set y values belong to can change depending on what problem is being approached. However, when the goal is to classify x values (whether or not a patient's vital signs indicate a heart attack or not), the set y belongs to should contain discrete numbers. In particular, if there are only two categories x can be classified as, then one is working with binary classification: $y^{(i)} \in \{+1, -1\}$.

The relationship between x and y values that a computer learns is called a hypothesis. It is some function that takes in an x and returns a y. Hypothesis also have parameters Θ, but they will be elaborated on later. A hypothesis is written as $y = h(x; \Theta)$.

How does one know that the prediction a hypothesis makes is accurate though? Loss functions are the answer. They are written as $L(g, a)$. The loss function takes in a hypothesis's guess g. g is basically just the value the hypothesis predicts given some x it encounters. a is the correct value that the hypothesis should return. The loss function returns how sad one should be that g was guessed when a was the actual answer. Because being sad is not fun, lower loss is better. A plethora of different loss functions exist.

- squared loss: $L(g, a) = (g - a)^2$
- linear loss: $L(g, a) = |g - a|$
- 0-1 loss:

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

Now, how can one measure how well a hypothesis works? To begin, the hypothesis should work well on the data it was trained on. The training set error allows measures the average loss of the hypothesis on the training data:

$E_n(h) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)}), y^{(i)})$. The hypothesis can be likened to a student in a calculus class though. Just memorizing homework answers does not guarantee the student full marks on the final exam. Think of the training set like a bunch of homework problems. The student should be able to generalize from practice and apply skills to test questions on final exam day. To really understand how well a hypothesis will perform on new data, one should save out a portion of the training data and call it the testing set. The test error is the average of the hypothesis's losses on the test set data: $E(h) = \frac{1}{n'}\sum_{i=n+1}^{n+n'} L(h(x^{(i)}), y^{(i)})$.

The computer learns hypotheses from datasets by using learning algorithms. However, learning algorithms will be explained more extensively later.