CS 598: Theoretical Machine Learning

Lecturer: Pranjal Awasthi
Scribe: Neelesh Kumar

Lecture #1
9/5/2017

1 Supervised Learning

An informal definition of supervised learning is as follows: we are given training data as input of the form: (x_1, y_1), (x_2, y_2),....., (x_m, y_m). Each x_i is a point in an instance space $\mathcal{X}$, and each $y_i = f(x_i)$ where f is the function we are interested in learning. The goal is to use the above training set to learn or approximate f. In other words, the goal of function approximation (or learning a function) is to use the training set to output $\hat{f}$ such that $\hat{f}$ is close to f, and therefore makes small errors on x.

1.1 Difference between Optimization and Machine Learning

Optimization only deals with input elements, i.e. the goal is to optimize over the input set. For example: Sorting takes a list of n elements as input and produces a sorted list of n elements as output. We are concerned only about the n elements present in the list, and not with any element present outside of the list. In contrast, the goal of machine learning is to optimize not only over the input set, but also to do well on new and unseen examples. This is called the problem of generalization.

1.2 Formalizing Generalization

The idea of generalization is to produce a function $\hat{f}$ that does as well as f (underlying function) on new data. For this to be true, there has to be some relationship between training data and test data, i.e. training and test data points should be "similar". We cannot expect any algorithm to do well if such a relationship doesn't exist. One way to enforce relationship is via probability distribution.

Let X be an instance space, i.e. space from which data is chosen (e.g. the space of all possible images). To build the training set, we assume that there is an unknown distribution D over X and each example x_i is drawn independently from X according to D. We generate m points in the same fashion to construct a training data set of size m. For testing, we draw new examples according to the same distribution D.

1.3 Notion of Error

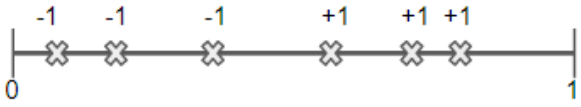
To evaluate the performance of the function $\hat{f}$ that is output by the learning algorithm, we define error as the probability that $\hat{f}$ makes mistakes on test data points. Mathematically, $err(\hat{f}) = Pr_{x\sim D}(\hat{f}(x) \neq f(x))$. Error is the fraction of the instance space where $\hat{f}$ makes a mistake. If D is uniform distribution, then error is the fraction of points on which $\hat{f}$ makes a mistake.

1.4 Formal definition of Supervised Learning

We can now formally define supervised learning as follows:
Input: (x_1, y_1), (x_2, y_2),....., (x_m, y_m), where $x_i \sim D$ and $y_i = f(x_i)$. $\epsilon > 0$. D is unknown to the algorithm.
Output: $\hat{f} : x \to \{+1, -1\}$ (for two class classification).
Goal: $err(\hat{f}) < \epsilon$; where $err(\hat{f})$=$Pr_{x \sim D}(\hat{f}(x) \neq f(x))$

2 Introduction to Probably Approximately Correct (PAC) Learning

To motivate the concept of PAC learning, we consider the following example of supervised learning:
Let the instance space X be a line, and the target function f be a threshold function, i.e. $f(x) = +1$ if $x > \theta$ and $f(x) = -1$ otherwise. The distribution D is a uniform distribution over the set $[0, 1]$. Note that D is unknown to the algorithm. We are given finite samples and the goal is to learn the threshold θ with error less than or equal to some small given constant ϵ.



Under such a scenario, we can make the following claims:
Claim 1: Using finite examples, it is not possible to achieve $\epsilon = 0$. Any function that the learning algorithm outputs is going to be **approximately correct**. But hopefully, we can control the error.

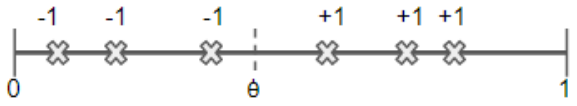
To achieve $\epsilon = 0$, all the points along the line which forms the instance space need to be included in training. This is so because if some part of the line is not included in training, there will always be some uncertainty in the prediction for the points along the missed part.
Claim 2: It is not possible for the learning algorithm to always achieve error ϵ. Any function that the learning algorithm outputs is **probably correct**.

Since the training points are drawn from distribution D, there is always a small (yet existent!) possibility that the training set is really bad, and not representative of the underlying distribution. This leads to some error due to uncertainty in the training set. Note that none of these subtleties are present in traditional optimization problems.
We can now formally define PAC learning as follows:

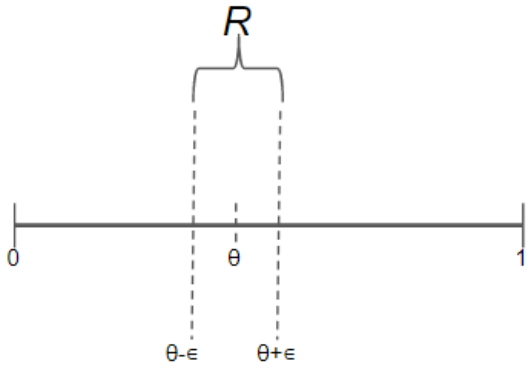
Definition 1 (PAC Learning)**.** *An algorithm A PAC learns unknown f if:* $\forall D$ *over* X, $\exists \epsilon > 0$, $\exists \delta > 0$, *A uses a training set S of finite number of examples and outputs* $\hat{f}$ *with probability,* $p \geq 1 - \delta$ *such that* $err(\hat{f}) \leq \epsilon$.

The challenge with the above definition is that it doesn't say anything about the number of training examples that the algorithm needs (the only condition is that it should be finite). It also doesn't say anything about the nature of the algorithm (in terms of running time and space).
We will now describe an algorithm that PAC learns the threshold function for the problem described at the beginning of the section. An approach to solving a learning problem is

to address the algorithmic and statistical questions. The algorithmic question refers to the learned function that makes prediction on new data points. The statistical question is what the size of training set should be in order to achieve PAC guarantee. The algorithm is as follows:
Scan the training examples from left to right and put a threshold $\hat{\theta}$ anywhere in the region where there is a change in label.



Next, we address the statistical question of the size of training set. Let the size of training set be m. Note that if $\hat{\theta}$ lies in the region R, then we can be confident that $err(\hat{f}) \leq \epsilon$.



To achieve this, the training set must contain at least 2 examples from the region R, one belonging to each class. Also, note that the probability of a single training data point not belonging to the region $[\theta, \theta + \epsilon]$ is at most $1 - \epsilon$. Similarly, the probability of a single training data point not belonging to the region $[\theta - \epsilon, \theta]$ is also at most $1 - \epsilon$. Since each point is independently drawn, the probability that we don't see at least one example from both the regions is upper bounded by $2(1 - \epsilon)^m$. In order to achieve PAC guarantee, the following must hold true:

$$
\begin{aligned}
2(1-\epsilon)^m &\leq \delta \\
\Rightarrow (1-\epsilon)^m &\leq e^{-\epsilon m} \leq \delta \\
\Rightarrow m &\leq \frac{1}{\epsilon} log(\frac{2}{\delta})
\end{aligned}
$$

3 References

- Chapters 2 and 3 from the book by Shai Shalev-Shwartz and Shai Ben-David. See online copy here: `http://www.cs.huji.ac.il/shais/UnderstandingMachineLearning/understandingmachine-learning-theory-algorithms.pdf`