Probably Approximately Correct Learning

Mengxi Wu (mengxiwu@usc.edu)

1 Notation

The notations we will use in this report are as follows.

- X: Instance space
- Y: Output space, such as $\{+1, -1\}$
- C: Function space, a family of functions $X \to Y$
- D: Unknown distribution over X
- f: Unknown target function $X \to Y$, taken from C
- h: The hypothesis function $X \to Y$ that the learning algorithm selects from a hypothesis class H
- S: The set of n training examples drawn from D, labeled with f
- $err_D(h)$: The true error of any hypothesis h
- $err_S(h)$: The empirical error or training error of h

2 Probably Approximately Correct Learning

Limitation of Learning. First, We cannot expect a learner to learn target function f exactly. It is natural to misclassify uncommon examples that do not include in the training set. Second, We cannot always expect to learn a close approximation to f. The training set can be not representative. Here we introduce the errors of learning.

True Error. The true error of the hypothesis h with respect to the target function f and the distribution D is the probability over x drawn from D that $h(x)$ and $f(x)$ differ.

$$err_D(h) = \mathrm{P}_D[h(x) \neq f(x)] \tag{2.1}$$

Empirical Error. The empirical error of the hypothesis h with respect to the target function f and the set S is the probability over x drawn from S that $h(x)$ and $f(x)$ differ.

$$err_S(h) = \mathrm{P}_S[h(x) \neq f(x)] = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}[f(x^i) \neq h(x^i)] \tag{2.2}$$

Probably Approximately Correctness (PAC). A good learner is that with a high probability it will learn a close approximation to the target function f. This concept relies on the Consistent Distribution Assumption: there is one probability distribution D that governs both training and testing samples.

Definition 2.1 (PAC Learnability)**.** *For a function space C, we say C is PAC-learnable if there exists an algorithm L with access to a query function, such that*
for every $f \in C$,
for any given probability distribution D,
for any given $\epsilon \in [0, \frac{1}{2}]$,

for any given $\delta \in [0, 1)$,
L on input ϵ, δ, and any $\{x^i\}_{i=1}^n$ sampled independently from D can output a hypothesis h such that

$$\mathrm{P}[err_D(h) \leq \epsilon] \geq 1 - \delta$$

Definition 2.2 (Efficient PAC Learnability)**.** *We say C is efficiently PAC learnable, if C is PAC learnable and the learning algorithm L can produce the hypothesis h in time polynomial in $\frac{1}{\epsilon}$, $\frac{1}{\delta}$, n, and size(H). Two Requirements:*

- *Polynomial sample complexity. It governs if there is enough information in training samples to distinguish a hypothesis h that approximate f.*
- *Polynomial time complexity. We also call it computational complexit. It tells if there is an efficient algorithm that can process the training samples and produce a good hypothesis h.*

Theorem 2.1. *The probability that there exists a bad hypothesis $h \in H$ that is consistent with n training samples and satisfies $err_D(h) > \epsilon$ is less then $|H|(1-\epsilon)^n$.*

Proof. Let h be a bad hypothesis. The probability that h is consistent with one training sample is less than $1 - \epsilon$. Since the n samples are independently drawn from D, the probability that h is consistent with n examples is less than $(1-\epsilon)^n$.

$$\mathrm{P}[\text{Any one of the } h \in H \text{ is consistent with } n \text{ samples}] \leq \sum_{h \in H} (1-\epsilon)^n = |H|(1-\epsilon)^n$$

We want our learning algorithm L to learn a good hypothesis h with training samples. Thus, we want the probability that L produces a bad hypothesis is smaller than a given very small number say δ. Given Theorem 2.1, we set $|H|(1-\epsilon)^n$ to be strictly smaller than δ and with the fact that $e^{-x} > 1 - x$,

$$|H|(1-\epsilon)^n < |H|e^{-\epsilon n} < \delta$$

$$\ln(|H|) - n\epsilon < \ln(\delta)$$

$$\ln(|H|) - \ln(\delta) < n\epsilon$$

$$n > \frac{1}{\epsilon}(\ln(|H|) + n\ln(\frac{1}{\delta})) \tag{2.3}$$

Eq.2.3 shows that if n is large enough, with high probability, a hypothesis h that is consistent with training samples can be learned to approximate f very well. From other pescpective, if we have small hypothesis space, we do not have to learn with too many training samples. However, here is a trade-off of the hypothesis space. If the space is small, then it generalizes well, but it may not be expressive enough.

References

[1] Matt Gormley. PAC Learning. link.

[2] Adam Klivans. Computational Learning Theory. link.

[3] Jeremy Kun. Probably Approximately Correct: a Formal Theory of Learning. link.

[4] Vivek Srikumar. Computational Learning Theory: Probably Approximately Correct (PAC) Learning. link.

[5] Ben Zhou and C. Cervantes. Computational Learning Theory. link.