Machine Learning Theory
Lecture 23: Boosting

Nicholas Harvey

November 29, 2018

"None of these things is any good on its own" my grandmother said. "It's only when you put them all together that they begin to make a little sense."

"The Witches", Roald Dahl

1 Motivation

Recall the definition of PAC-learnability.

Definition 1.1. $\mathcal{H}$ is PAC-learnable if there exists an algorithm $\mathcal{A}$ and a function $m(\epsilon, \delta)$ such that, for *every* distribution $\mathcal{D}$, for *every* $\epsilon > 0$, and for *every* $\delta > 0$, if $\mathcal{A}$ is given $m(\epsilon, \delta)$ i.i.d. samples S from $\mathcal{D}$, then it outputs a hypothesis h_S such that

$$\Pr_S\left[\, L_{\mathcal{D}}(h_S) \leq \epsilon \,\right] \;\geq\; 1 - \delta.$$

This condition involves *three* universal quantifiers, so may be quite difficult to satisfy!

Interestingly, the $\forall \mathcal{D}$ quantifier turns out to be the most important one. We have shown in Assignment 2, Question 4, that the $\forall \delta$ quantifier can be satisfied for free: as long as Definition 1.1 holds with δ fixed at $1/2$, it is possible to "amplify" the probability of success using validation.

The $\forall \epsilon$ quantifier can (in some sense) be satisfied "for free", using the technique of ***Boosting***. That is, if Definition 1.1 holds for ϵ bounded away from $1/2$, then Boosting shows that a "majority vote" of several classifiers from $\mathcal{H}$ can achieve arbitrarily small ϵ. However $\mathcal{H}$ is not necessarily closed under taking majority votes, so the hypothesis produced by Boosting might not belong to $\mathcal{H}$.

Weak vs Strong Learnability. Let us now introduce some definitions to more precise the goal of Boosting.

- A ***γ-weak learner*** with sample complexity $m(\delta)$ is an efficient algorithm $\mathcal{A}$ that, given δ and any distribution $\mathcal{D}$ over $\mathcal{X}$, draws $m(\delta)$ samples from $\mathcal{D}$ then returns a hypothesis $h \in \mathcal{H}$ satisfying $\ell^{01}_{\mathcal{D}}(h) \leq 1/2 - \gamma$ with probability at least $1 - \delta$.

- A ***strong learner*** with sample complexity $m(\epsilon, \delta)$ is an efficient algorithm $\mathcal{A}^*$ that, given ε, δ and any distribution $\mathcal{D}$ over $\mathcal{X}$, draws $m(\varepsilon, \delta)$ samples from $\mathcal{D}$ then returns a hypothesis $h \in \mathcal{H}$ satisfying $\ell^{01}_{\mathcal{D}}(h) \leq \varepsilon$ with probability at least $1 - \delta$.

Let $\mathcal{H}^{(T)}$ be set of hypotheses that are the majority vote of T classifiers in $\mathcal{H}$. We will show that, given oracle access to a weak learner $\mathcal{A}$, Boosting can:

- Produce a hypothesis $h^* \in \mathcal{H}^{(T)}$ with *training error* that tends to 0 as T increases.
- Produce a hypothesis $h^* \in \mathcal{H}^{(T)}$ with *test error* that tends to 0 as T increases, and if the number of samples is sufficiently large as a function of T. Thus, Boosting gives a strong learner for $\bigcup_{T>0} \mathcal{H}^{(T)}$. (This argument is a bit convoluted as it requires bounding the VC-dimension of $\mathcal{H}^{(T)}$ — see Section 2.1.)

The Boosting Idea. First draw a sufficiently large sample sequence

$$S \;=\; \big((X_1, Y_1), ..., (X_n, Y_n)\big)$$

i.i.d. from $\mathcal{D}$. Recall that the training error is defined as the expected loss of the uniform distribution over S.

The main idea of boosting is to repeatedly apply the weak learner to many different distributions over the sample S, generating many hypotheses $h_1, ..., h_T$. The final classifier h^* will be the majority vote over these hypotheses. The conclusion will be that h^* has low error with respect to (the uniform distribution over) the sample, i.e., for any desired $\hat{\epsilon} > 0$, we can achieve $\ell^{01}_S(h^*) \leq \hat{\epsilon}$ if T is sufficiently large as a function of $\hat{\epsilon}$.

The non-obvious idea is how to create these distributions. Somewhat counterintuitively, Boosting will

- *decrease* the probability for samples that are classified *correctly*, because the existing hypothesis already does a good job.
- *increase* the probability for samples that are classified *incorrectly*, in order to incentivize the next hypothesis to classify it correctly.

This perhaps makes sense at an informal level, but implementing it formally seems technically daunting. Fortunately, we do not need to reinvent the wheel: we have already seen a powerful framework for reweighting objects based on past performance, namely Randomized Weighted Majority.

RWM for Optimization. Our discussion of RWM was motivated by the scenario of prediction with expert advice. The RWM machinery is actually very powerful, and can be used to solve all sorts of constrained optimization problems. To do so, there are two main tasks:

- **Determine the experts.** Intuitively, the "experts" correspond to some "constraints" that we wish to satisfy. In the Boosting context, the experts are the samples, each of which imposes a contraint of being correctly classified. RWM will maintain a distribution over the samples indicating the "importance" of future hypotheses correctly classifying each sample.

- **Determine the adversarial costs.** Implement an "oracle" that knows the current distribution and behaves in an adversarial way to maximize the algorithm's cost.

  In the Boosting context, we wish to decrease the probability for samples that are correctly classified by the current hypothesis h_i. In the RWM framework, an expert's probability is decreased if it has large cost. So a sample should have large cost if it is classified correctly. The adversary's goal is to maximize the algorithm's expected cost, so the adversary should choose h_i to classify many samples correctly (according to distribution x_i). This can be accomplished by the weak learner.

2 Formal discussion

Each "expert" corresponds to a sample in S. (There are n experts and n samples.) RWM maintains a distribution x_i over the experts, which is corresponds to a distribution over the samples. Given such a distribution x_i, the "oracle" will use the weak learner to produce a hypothesis h_i satisfying $\ell^{01}_{x_i}(h_i) \leq 1/2 - \gamma$. This hypothesis determines the costs incurred by each expert.

Recall that RWM increases the weight of each expert that had low cost. We want to increase the probability (in the distribution x_{i+1}) for samples incorrectly classified by h_i, so we (counterintuitively)

- give cost 0 to the samples that are classified *incorrectly.*
- give cost 1 to the samples that are classified *correctly.*

In other words, we will define the cost vector $c_i \in \{0,1\}^n$ by

$$\begin{aligned} c_{i,j} &= 1_{h_i(X_j)=Y_j} = 1 - \ell^{01}(h_i,(X_j,Y_j)) \\ \Longrightarrow \qquad c_i^{\mathsf{T}} x_i &= \sum_{i=1}^{n} x_{i,j}\big(1 - \ell^{01}(h_i,(X_j,Y_j))\big) = 1 - \ell^{01}_{x_i}(h_i) \geq 1/2 + \gamma, \end{aligned}$$

since the weak learner guarantees $\ell^{01}_{x_i}(h_i) \leq 1/2 - \gamma$. Thus the total cost of the RWM algorithm is

$$A := \sum_{i=1}^{T} c_i^{\mathsf{T}} x_i \geq (1/2+\gamma)T.$$

Number of misclassified samples. Now our main goal is to analyze the number of misclassified samples. Observe that the j^{th} sample is misclassified by h^* if it is misclassified by a majority of $h_1, ..., h_T$ (due to the definition of h^*). In notation,

$$h^*(X_j) \neq Y_j \qquad \Longrightarrow \qquad |\{\, i \,:\, h_i(X_j) = Y_j \,\}| \leq T/2.$$

However, by the definition of the costs, we have

$$\sum_{i=1}^{T} c_{i,j} = |\{\, i \,:\, h_i(X_j) = Y_j \,\}|. \tag{2.1}$$

So we would like to bound the number of experts j whose total cost $\sum_{i=1}^{T} c_{i,j} \leq T/2$. This can be done using the result of Assignment 4, Question 1.

Lemma 2.1. Suppose the RWM algorithm is executed with parameters η and ϵ, where $\epsilon = 1 - e^{-\eta} \leq 1/2$. At time T, the number of experts with total cost at most C is at most

$$n \exp\big((\epsilon + \epsilon^2)C - \epsilon A\big).$$

Applying Lemma 2.1 with $C = T/2$ and $\epsilon = \gamma$, the number of experts with total cost at most $T/2$ is at most

$$n \exp\big((\epsilon + \epsilon^2)C - \epsilon A\big) \;=\; n \exp\big((\gamma + \gamma^2)T/2 - \gamma(1/2 + \gamma)T\big) \;=\; n \exp\big(-\gamma^2 T/2\big) \;=\; n\hat{\epsilon},$$

if we choose the number of iterations to be $T := 2\ln(1/\hat{\epsilon})/\gamma^2$.

To conclude, the number of misclassified samples is at most the number of experts of cost at most $T/2$, which is at most $n\hat{\epsilon}$. Thus

$$\ell_S^{01}(h^*) \;=\; \frac{1}{n} \cdot (\#\text{ samples incorrectly classified by } h^*) \;\leq\; \frac{1}{n}(n\hat{\epsilon}) \;=\; \hat{\epsilon}.$$

Success probability. Note that this argument required T calls to the weak learner. Our assumption on the weak learner only guarantees that it will succeed with probability at most δ. Let us choose $\delta = \hat{\delta}/T$. Thus, by a union bound, the overall failure probability of the Boosting algorithm is at most $\delta T = \hat{\delta}$.

Summary. We have shown that, with black-box access to a γ-weak learner with success probability $\hat{\delta}/T$, running the Boosting algorithm for $2\ln(1/\hat{\epsilon})/\gamma^2$ iterations produces a hypothesis h^* with training error $\ell_S^{01}(h^*) \leq \hat{\epsilon}$ with probability at least $1 - \hat{\delta}$. This hypothesis h^* does not lie in $\mathcal{H}$, but is a (weighted) majority vote of T hypotheses from $\mathcal{H}$, so $h^* \in \mathcal{H}^{(T)}$

2.1 Test error

So far we have only shown that h^* has low *training error* (i.e., error with respect to the uniform distribution on the sample S). We can also show that h^* has low *test error* by a uniform convergence argument for $\mathcal{H}^{(T)}$, the class of hypotheses that are a (weighted) majority vote of hypotheses in $\mathcal{H}$.

Suppose that $\mathcal{H}^{(T)}$ satisfies the uniform convergence property with sample complexity $m(\varepsilon, \delta)$. We draw a sample $S = (X_1, Y_1), ..., (X_n, Y_n)$ from $\mathcal{D}$ with $n = m(\varepsilon/2, \delta)$. With probability at least $1 - \delta$, we have

$$|\ell_S^{01}(h) - \ell_{\mathcal{D}}^{01}(h)| \;\leq\; \varepsilon \qquad \forall h \in \mathcal{H}^{(T)}.$$

It follows that $\ell_{\mathcal{D}}^{01}(h^*) \leq \ell_S^{01}(h) + \varepsilon \leq 2\varepsilon$.

To show that $\mathcal{H}^{(T)}$ has the uniform convergence property, it suffices to bound its VC-dimension. It is shown in [1, Lemma 10.3] that $\text{VCdim}(\mathcal{H}^{(T)}) = \tilde{O}(T \cdot \text{VCdim}(\mathcal{H}))$.

References

[1] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.