ELEG/CISC 867: Advanced Machine Learning **Spring 2019**

Lecture 8: Boosting

Lecturer: Xiugang Wu *03/26/2019, 03/28/2019*

Boosting is a meta-algorithm that uses a generalization of linear predictors to address two major issues in machine learning: bias-complexity tradeoff and computational complexity.

- In boosting paradigm, the learning starts with a basic class (that may have a large approximation error), and as it progresses the class grows richer. This allows the learner to have smooth control over the bias-complexity tradeoff.
- Boosting provides a tool for aggregating *weak* hypotheses to approximate gradually good predictors for larger, and harder to learn, classes. This addresses the computational complexity of learning.

This lecture will introduce a popular boosting algorithm, AdaBoost (Adaptive Boosting). AdaBoost outputs a hypothesis that is a linear combination of simple hypotheses, and enables us to control the tradeoff between approximation and estimation errors by varying a single parameter. Throughout this lecture, we assume that the realizability assumption holds.

1 Weak Learnability

First recall the definition of PAC learnability in the realizable case. A hypothesis class, $\mathcal{H}$, is PAC learnable in the realizable case if there exist $n_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm A with the following property: For every $\epsilon, \delta \in (0,1)$, for every distribution P over $\mathcal{X}$, and for every labeling function $f : \mathcal{X} \to \{\pm 1\}$, if the realizable assumption holds with respect to $\mathcal{H}, P, f$, then when running the learning algorithm A on $n \geq n_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples Z^n generated by P and labeled by f, the algorithm returns a hypothesis $A(Z^n) \in \mathcal{H}$ such that

$$P^n(L(A(Z^n), P, f) \leq \epsilon) \geq 1 - \delta. \tag{1}$$

By the fundamental theorem of learning, $\mathcal{H}$ is learnable iff its VC dimension is finite, and any learnable class can be learned using any ERM algorithm.

The above definition of PAC learning captures the statistical aspect of learning, but ignores the computational complexity. Indeed, it can be shown that implementing an ERM learner can be computationally hard (even in the realizable case); see Chapter 8 in the textbook. This motivates us to trade computational hardness with the requirement for accuracy. Given a distribution P and a target labeling function f, maybe there exists an efficiently computable learning algorithm whose error is just slightly better than a random guess? This idea is formalized in the following definition.

Definition 1.1 *A hypothesis class, $\mathcal{H}$, is γ-weak-learnable in the realizable case if there exist $n_{\mathcal{H}} : (0,1) \to \mathbb{N}$ and a learning algorithm A such that for any $\delta \in (0,1)$, any distribution P over $\mathcal{X}$, and any labeling function $f : \mathcal{X} \to \{\pm 1\}$, if the realizable assumption holds with respect to $\mathcal{H}, P, f$, then*

$$P^n(L(A(Z^n), P, f) \leq 1/2 - \gamma) \geq 1 - \delta, \tag{2}$$

whenever $n \geq n_{\mathcal{H}}(\delta)$, and in this case the algorithm A is said to be a γ-weak-learner for $\mathcal{H}$.

The difference between the above weak learning and the aforementioned strong learning lies in the different accuracy requirements in (1) and (2): strong learnability requires arbitrarily small error while weak learnability only requires error slightly smaller than 1/2, i.e. slightly better than a random guess. Let's discuss the impact of adopting weak learning as compared to strong learning:

- **Statistical Perspective:** The quantitative version of the fundamental theorem of learning (Theorem 6.8 in the textbook) tells us that the sample complexity of PAC learning satisfies $n_{\mathcal{H}}(\epsilon, \delta) \geq C_1 \frac{d+\log(1/\delta)}{\epsilon}$ for some constant C_1; applying this with $\epsilon = 1/2 - \gamma$, we see that if $d = \infty$ then $\mathcal{H}$ is not γ-weak-learnable. Therefore, weak learnability is also characterized by the VC dimension, and is (qualitatively) as hard as strong learnability if we ignore computational complexity and only consider sample complexity.
- **Computational Perspective:** When we do consider computational complexity, however, the potential advantage of weak learning is that there may exist an algorithm that satisfies the requirements of weak learning and can be implemented efficiently. For example, to construct a weak learner for $\mathcal{H}$, we may consider applying ERM with respect to some "simple" base hypothesis class $\mathcal{B}$, where $\mathcal{B}$ is such that i) $\text{ERM}_{\mathcal{B}}$ is efficiently implementable and ii) for any distribution consistent with $\mathcal{H}$, $\text{ERM}_{\mathcal{B}}$ hypothesis has error of at most $1/2 - \gamma$.

Assume the existence of an *efficient* weak learner. A natural question is then: Can we boost it into an *efficient* strong learner? We will introduce the AdaBoost algorithm and show that this is indeed possible. But before that, let us use an example to demonstrate that one can indeed build an efficient weak learner using a base hypothesis class $\mathcal{B}$.

1.1 Weak Learning of 3-Piece Classifiers Using Decision Stumps

Consider a binary classification problem, where $\mathcal{X} = \mathbb{R}$ and $\mathcal{Y} = \{-1, +1\}$. Let $\mathcal{H}$ be the class of 3-piece classifiers:

$$\mathcal{H} = \{h_{\theta_1,\theta_2,b} : \theta_1, \theta_2 \in \mathbb{R}, \theta_1 < \theta_2, b \in \{-1, +1\}\}$$

where $h_{\theta_1,\theta_2,b}$ is defined as

$$h_{\theta_1,\theta_2,b}(x) = \begin{cases} +b & x < \theta_1 \text{ or } x > \theta_2 \\ -b & x \in [\theta_1, \theta_2] \end{cases}.$$

Let $\mathcal{B}$ be the class of decision stumps:

$$\mathcal{B} = \{x \mapsto \text{sgn}(x - \theta) \cdot b : \theta \in \mathbb{R}, b \in \{-1, +1\}\}.$$

We can show that $\text{ERM}_{\mathcal{B}}$ is a γ-weak-learner for $\mathcal{H}$ with $\gamma = 1/12$. To see this, first note that for every distribution P consistent with $\mathcal{H}$, there exists a decision stump $h \in \mathcal{B}$ with $L(h, P) \leq 1/3$. Since the VC dimension of decision stumps is 2, from Theorem 6.8 in the textbook we have that if the sample size $n = \Omega(\frac{2+\log(1/\delta)}{\epsilon^2})$, then with probability at least $1 - \delta$, $\text{ERM}_{\mathcal{B}}$ returns a hypothesis with error at most $1/3 + \epsilon$. Setting $\epsilon = 1/12$, we have that if $n = \Omega(144 \log(1/\delta))$, then with probability at least $1 - \delta$, the error of $\text{ERM}_{\mathcal{B}}$ hypothesis is at most $1/3 + 1/12 = 1/2 - 1/12$. Furthermore, it can easily shown that $\text{ERM}_{\mathcal{B}}$ can be implemented efficiently in time $O(p(n))$.

2 AdaBoost

AdaBoost is a meta-algorithm that has access to a weak learner and finds a hypothesis with low empirical risk.

- The input of the AdaBoost algorithm is a set of training examples $z^n = \{(x_i, y_i)\}_{i=1}^n$, where $y_i = f(x_i), \forall i \in [1:n]$ for some labelling function f.
- The boosting process proceeds in a sequence of consecutive rounds. At round t, the booster has a distribution $\mathbf{D}^{(t)}$ on $[1:n]$ from the $(n-1)$-dimensional probability simplex, i.e. $\mathbf{D}^{(t)}$ is such that i) $\mathbf{D}^{(t)}(i) \geq 0, \forall i \in [1:n]$, and ii) $\sum_{i\in[1:n]} \mathbf{D}^{(t)}(i) = 1$. The booster passes the distribution $\mathbf{D}^{(t)}$ and training sample z^n to the weak learner.
- The weak learner returns a weak hypothesis h_t, whose error ϵ_t is less than $1/2-\gamma$ with probability $1-\delta$; here the error ϵ_t is calculated according to the distribution $P_n^{(t)}$ on $\mathcal{X}$ that is induced by $(\mathbf{D}^{(t)}, z^n)$, i.e.,

$$P_n^{(t)}(x) = \sum_{i=1}^n \mathbf{D}^{(t)}(i) \cdot \mathbb{I}(x_i = x), \forall x \in \mathcal{X},$$

  and

$$\epsilon_t \triangleq L(h_t, P_n^{(t)}, f) = \sum_{i=1}^n \mathbf{D}^{(t)}(i) \cdot \mathbb{I}(h_t(x_i) \neq y_i).$$

- Then AdaBoost assigns a weight w_t for h_t, i.e. $w_t = \frac{1}{2}\log(1/\epsilon_t - 1)$; the higher error of h_t the smaller weight h_t gets, and vice versa.
- At the end of the round t, AdaBoost updates the distribution $\mathbf{D}^{(t)}$ to $\mathbf{D}^{(t+1)}$ so that examples on which h_t errs will get a higher probability mass while examples on which h_t is correct will get a lower probability mass; this allows the weak learner to focus on the problematic examples in the next round.
- Finally, AdaBoost outputs a strong classifier that is based on a weighted sum of all the weak hypotheses.

The pseudocode of AdaBoost is in the following.

Algorithm 1 AdaBoost

input: Training sample $z^n = \{(x_i, y_i)\}_{i=1}^n$, weak learner WL, number of rounds T
initialize: $\mathbf{D}^{(1)} \leftarrow (\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$
for $t = 1, \ldots, T$ **do**
  $h_t \leftarrow \mathrm{WL}(\mathbf{D}^{(t)}, z^n)$
  $\epsilon_t \leftarrow \sum_{i=1}^n \mathbf{D}^{(t)}(i) \cdot \mathbb{I}(h_t(x_i) \neq y_i)$
  $w_t \leftarrow \frac{1}{2}\log(1/\epsilon_t - 1)$
  $\mathbf{D}^{(t+1)}(i) \leftarrow \frac{\mathbf{D}^{(t)}(i)e^{-w_t y_i h_t(x_i)}}{\sum_{j=1}^n \mathbf{D}^{(t)}(j)e^{-w_t y_j h_t(x_j)}}, \forall i \in [1:n]$
end for
output $h_s(x) = \mathrm{sgn}(\sum_{t=1}^T w_t h_t(x))$

The following theorem shows that the training error of the output strong hypothesis h_s decreases exponentially fast with the number T of boosting rounds.

Theorem 2.1 *Let z^n be the training set, and assume that at each iteration t of AdaBoost, the weak learner returns a hypothesis h_t for which $\epsilon_t \leq 1/2 - \gamma$. Then the training error of the output hypothesis h_s of AdaBoost after T iterations is at most*

$$L(h_s, z^n) = \frac{1}{n}\sum_{i=1}^n \mathbb{I}(h_s(x_i) \neq y_i) \leq e^{-2\gamma^2 T}.$$

Proof: For each $t \in [1:T]$, let $f_t = \sum_{p \le t} w_p h_p$ so that the output of AdaBoost is $\operatorname{sgn} \circ f_T$, and let $f_0 = 0$. For each $t \in [0:T]$, denote

$$Z_t = \frac{1}{n} \sum_{i=1}^{n} e^{-y_i f_t(x_i)},$$

so that the training error of $\operatorname{sgn} \circ f_t$ is upper bounded by Z_t, i.e.,

$$L(\operatorname{sgn} \circ f_t, z^n) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(\operatorname{sgn} \circ f_t(x_i) \neq y_i) \le \frac{1}{n} \sum_{i=1}^{n} e^{-y_i f_t(x_i)} = Z_t.$$

To prove the theorem it suffices to show $Z_T \le e^{-2\gamma^2 T}$.

Now write

$$Z_T = \frac{Z_T}{Z_{T-1}} \cdot \frac{Z_{T-1}}{Z_{T-2}} \cdots \frac{Z_2}{Z_1} \cdot \frac{Z_1}{Z_0}; \tag{3}$$

we will show

$$\frac{Z_t}{Z_{t-1}} \le e^{-2\gamma^2}, \forall t \in [1:T]. \tag{4}$$

For this, first note that by induction we have

$$\mathbf{D}^{(t)}(i) = \frac{e^{-y_i f_{t-1}(x_i)}}{\sum_{j=1}^{n} e^{-y_j f_{t-1}(x_j)}}.$$

Hence for any $t \in [1:T]$,

$$\begin{aligned}
\frac{Z_t}{Z_{t-1}} &= \frac{\sum_{i=1}^{n} e^{-y_i f_t(x_i)}}{\sum_{i=1}^{n} e^{-y_i f_{t-1}(x_i)}} \\
&= \frac{\sum_{i=1}^{n} e^{-y_i f_{t-1}(x_i)} \cdot e^{-y_i w_t h_t(x_i)}}{\sum_{i=1}^{n} e^{-y_i f_{t-1}(x_i)}} \\
&= \sum_{i=1}^{n} \frac{e^{-y_i f_{t-1}(x_i)}}{\sum_{i=1}^{n} e^{-y_i f_{t-1}(x_i)}} \cdot e^{-y_i w_t h_t(x_i)} \\
&= \sum_{i=1}^{n} \mathbf{D}^{(t)}(i) \cdot e^{-y_i w_t h_t(x_i)} \\
&= \sum_{i: y_i h_t(x_i) = 1} \mathbf{D}^{(t)}(i) \cdot e^{-w_t} + \sum_{i: y_i h_t(x_i) = -1} \mathbf{D}^{(t)}(i) \cdot e^{w_t} \\
&= (1 - \epsilon_t) \frac{1}{\sqrt{1/\epsilon_t - 1}} + \epsilon_t \sqrt{1/\epsilon_t - 1} \\
&= 2\sqrt{\epsilon_t (1 - \epsilon_t)}
\end{aligned}$$

which, for $\epsilon_t \le 1/2 - \gamma$, is upper bounded by

$$2\sqrt{\epsilon_t(1-\epsilon_t)} \le 2\sqrt{(1/2-\gamma)(1/2+\gamma)} = \sqrt{1-4\gamma^2} \le e^{-2\gamma^2}.$$

This proves the theorem. □

2.1 Discussion

- Computational Complexity: Each iteration of AdaBoost involves a single call to the weak learner and $O(n)$ operations; therefore, AdaBoost is efficient if the weak learner can be implemented efficiently (c.f. the case of ERM with respect to decision stumps).
- Failure Probability of Boosting: According to the definition of a weak learner, it can fail with probability δ. Using the union bound, the probability that the weak learner will not fail at any of the iterations is at least $1 - \delta T$. One can make δT very small without affecting the sample complexity too much since the dependence of the sample complexity on δ can always be logarithmic in $1/\delta$.
- Generalization Error: The above theorem shows that the training error of AdaBoost output hypothesis goes to zero as T grows. But what about the true error? Note that the output of AdaBoost is in fact a composition of a halfspace over the prediction of T weak hypotheses; we can show that if the weak hypotheses come from a base class of low VC dimension, then the generalization error is small, i.e. the true error is close to the training error, for AdaBoost output hypothesis.

3 Linear Combinations of Base Hypotheses

As mentioned before, a popular approach for constructing a weak learner is to apply ERM with respect to a base hypothesis class, say $\mathcal{B}$. In this case, the output of AdaBoost is from the hypothesis class of composing a halfspace over $\mathcal{B}^T$:

$$\begin{aligned}\mathcal{L}(\mathcal{B}, T) &= \left\{ x \mapsto \text{sgn}(\sum_{t=1}^{T} w_t h_t(x)) : \mathbf{w} \in \mathbb{R}^T; h_t \in \mathcal{B}, \forall t \in [1:T] \right\} \\ &= \left\{ x \mapsto \text{sgn}(\mathbf{w}^T \psi(x)) : \mathbf{w} \in \mathbb{R}^T; \psi(x) = (h_1(x), h_2(x), \ldots, h_T(x)), \text{ where } h_t \in \mathcal{B}, \forall t \in [1:T] \right\}.\end{aligned}$$

3.1 Bias-Complexity Tradeoff: Estimation Error vs. Approximation Error

- It can be shown that up to logarithmic factors, the VC dimension of $\mathcal{L}(\mathcal{B}, T)$ is upper bounded by T times the VC dimension of $\mathcal{B}$, i.e., VC-d$(\mathcal{L}(\mathcal{B}, T)) = \tilde{O}(\text{VC-d}(\mathcal{B}) \cdot T)$. Therefore, the estimation error of AdaBoost grows linearly with T, and will be small if $\mathcal{B}$ has a low VC dimension.
- On the other hand, the training error of AdaBoost decreases with T. In fact, it turns that T can be used to decrease the approximation error of $\mathcal{L}(\mathcal{B}, T)$, or in other words, to increase the expressive power of $\mathcal{L}(\mathcal{B}, T)$. See the example in Section 10.3 of the textbook, where it is shown that the class of halfspaces over T decision stumps yields all the T-piece classifier.
- Therefore, T is a parameter that an control the bias-complexity tradeoff. Enlarging T yields a more expressive hypothesis class but on the other hand might increase the estimation error.