Lecture 6, November 29, 2016

Machine Learning Theory

Boosting

Ruth Urner

1 Recap

We have seen that, for classes $\mathcal{H}$ of bounded VC-dimension, we get the following guarantee on the generalization error of an ERM classifier. There exits a constant C, such that (with high probability over the the draw of the sample S), we have

$$L(\hat{h}_n) \leq L(h^*) \ + \ C\sqrt{\frac{\mathrm{VC}(\mathcal{H}) + \ln 1/\delta}{n}} \tag{1}$$

We see that the generalization error of an empirical risk minimizer decomposes into two terms: $L(h^*) = \inf_{h\in\mathcal{H}} L(h)$ is called the *approximation error* of the hypothesis class $\mathcal{H}$, and it measures how suitable for the problem at hand the class that we chose was. The other part, $C\sqrt{\frac{\mathrm{VC}(\mathcal{H})+\ln 1/\delta}{n}}$, is called the *estimation error*. It goes down with the size n of the data sample that the learner saw. In fact, we saw in Lecture 3 that this term is a bound on how much the empirical risk estimates deviate from the true risks of the functions in our class $\mathcal{H}$. There, we used that inequality

$$L(\hat{h}_n) - L(h^*) \ \leq \ L(\hat{h}_n) - L_n(\hat{h}_n) + L_n(h^*) - L(h^*) \ \leq \ 2\max_{h\in\mathcal{H}} |L_n(h) - L(h)| \tag{2}$$

and then showed that

$$\max_{h\in\mathcal{H}} |L_n(h) - L(h)| \ \leq \ C\sqrt{\frac{\mathrm{VC}(\mathcal{H}) + \ln 1/\delta}{n}} \tag{3}$$

for a suitable constant C. The estimation error quantifies how good an indicator of the true risks the empirical risk estimates of the functions in out class are. Remarkably, for classes of bounded VC-dimension, we get a *uniform* bound bound on how much true and empirical risks may deviate, that is, the above bounds holds *simultaneously* for all functions h in the class $\mathcal{H}$ (as is implied by the max in the above inequality).

We see that we actually proved something stronger than what was needed for equation 1. We actually proved in Lecture 3 that classes of bounded VC-dimension enjoy the so-called *uniform convergence property*.

Definition 1. *Let $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ be a hypothesis class. We say that $\mathcal{H}$* has the uniform convergence property *if the following holds:*

For all $\epsilon, \delta > 0$, there exists a $n(\epsilon, \delta) \in \mathbb{N}$ such that for all distributions P over $\mathcal{X} \times \{0, 1\}$ and all $n \geq n(\epsilon, \delta)$, we have

$$\mathbb{P}_{S \sim P^n} \left[\max_{h \in \mathcal{H}} \{|L(h) - L_n(h)|\} \leq \epsilon \right] \;\geq\; 1 - \delta$$

Equation 2 shows how the uniform convergence property for a class implies Equation 1. It also shows that uniform convergence of the class implies learnability (in the sense of Definition 1 in Lecture 2) with any ERM learner. Therefore, we know that a binary hypothesis class enjoys the uniform convergence property if and only if it has bounded VC-dimension.

It is easy to see that uniform convergence, in form of Equation 3, implies the following bound on the true risk of the functions in the class. There exits a constant C, such that (with high probability over the the draw of the sample S), we have for all functions $h \in \mathcal{H}$

$$L(h) \leq L_n(h) \;+\; C\sqrt{\frac{\mathrm{VC}(\mathcal{H}) + \ln 1/\delta}{n}}, \tag{4}$$

That is, the true risk of any function in the class is bounded by its empirical risk (which is an observable quantity; it can be computed from the sample for any function h) plus the estimation error. In particular, for an empirical risk minimizer, we then also get

$$L(\hat{h}_n) \leq L_n(\hat{h}_n) \;+\; C\sqrt{\frac{\mathrm{VC}(\mathcal{H}) + \ln 1/\delta}{n}}. \tag{5}$$

Questions:

- How does the size of the hypothesis class affect the tradeoff of Equation 1? What happens if the class grows to approximation error and estimation error?
- What happens if $\mathcal{H}$ is the class of all binary functions over domain $\mathcal{X}$?

2 Boosting

Before introducing the general boosting paradigm, we need several ingredients.

2.1 Weak learning

Boosting relies on the idea that a certain weaker notion of learnability than Definition 1 in Lecture 2 may be easier to achieve. Namely, instead of requiring that an learning algorithm can achieve arbitrarily small generalization error (or approximate the approximation error of the class arbitrarily well), *weak learning* only requires that a learner can do somewhat better than random guessing.

Definition 2. *Let $\gamma > 0$ and let $\mathcal{H}$ be some hypothesis class. A learner $\mathcal{A}$ is a γ-weak learner for the class $\mathcal{H}$ if, for all $\delta > 0$ there is a sample size $m(\delta)$ such that for all distributions P over $\mathcal{X} \times \{0, 1\}$ that are realizable by $\mathcal{H}$, we have for all $m \geq m(\delta)$*

$$\mathbb{P}_{S \sim P^m}[L(\mathcal{A}(S)) \leq 1/2 - \gamma] \;\geq\; 1 - \delta$$

That is, a weak leaner is required to do better than random guessing by some fixed margin γ. The idea of boosting is to use such a weak learner repeatedly and combine the resulting predictors in to a majority vote predictor that can then be shown to do significantly better than its "weak" components.

2.2 Decision stumps

A class that is often used to get a weak learner in boosting are Decision Stumps. We will see below that for this class an ERM rule is efficiently implementable, which additionally makes this class computationally attractive.

Decision Stumps in one dimension are defined as follows:

$$\mathcal{H}_{1-\text{dec}-\text{st}} = \{h_{a,c} \ : \ a \in \mathbb{R}, c \in \{+1, -1\}\}$$

with

$$h_{a,c}(x) = \begin{cases} c & \text{if } x \leq a \\ -c & \text{if } x > a. \end{cases}$$

We will now show that ERM with respect to the class of one dimensional decision stumps provides a weak learner for the class of signed intervals. The class of signed intervals is defined as follows:

$$\mathcal{H}_{\text{s}-\text{int}} = \{h_{a,b,c} \ : \ a, b \in \mathbb{R}, c \in \{+1, -1\}\}$$

with

$$h_{a,b,c}(x) = \begin{cases} c & \text{if } x \in [a, b] \\ -c & \text{else .} \end{cases}$$

Note that for any distribution P over $\mathbb{R}$ that is realizable by the class $\mathcal{H}_{\text{s}-\text{int}}$, there exists a function $h \in \mathcal{H}_{1-\text{dec}-\text{st}}$ that has loss less than $1/3$ with respect to P. To see this, let $h_{a,b,c}$ be the function that realizes P, that is, has zero loss with respect to P. Then one of the areas $\{x \in \mathbb{R} \ : \ x \leq a\}$, $\{x \in \mathbb{R} \ : \ a \leq x \leq b\}$ or $\{x \in \mathbb{R} \ : \ b \leq x\}$ has to have probability weight at most $1/3$. It is easy to see that there is a decision stump that agrees with $h_{a,b,c}$ on the other two areas and therefore has loss at most $1/3$.

Thus, the approximation error of $\mathcal{H}_{1-\text{dec}-\text{st}}$ for distributions realizable by $\mathcal{H}_{\text{s}-\text{int}}$ is always at most $1/3$. Now, if we we learn (by ERM) the class $\mathcal{H}_{1-\text{dec}-\text{st}}$ up to estimation error $1/12$ we get

$$L(\hat{h}_n) \leq L(h^*) + 1/12 \leq 1/3 + 1/12 \leq 1/2 - 1/12$$

Note that guaranteeing estimation error of at most $1/12$ is possible by ERM with

$$m(\delta) = C\frac{\text{VC}(\mathcal{H}_{1-\text{dec}-\text{st}}) + \ln(1/\delta)}{(1/12)^2}$$

many samples according to our introductory discussion. Thus, we have seen that outputting an ERM decision stump yields a γ weak learner for the class of signed intervals with $\gamma = 1/12$.

Next, we will argue that an ERM rule can be efficiently implemented for decision stumps in any dimension. General decision stumps in $\mathbb{R}^d$ are defined as follows:

$$\mathcal{H}_{\text{dec}-\text{st}} = \{h_{a,c,i} \ : \ a \in \mathbb{R}, c \in \{+1, -1\}, i \in [d]\}$$

with

$$h_{a,c,i}(\boldsymbol{x}) = \begin{cases} c & \text{if } x_i \leq a \\ -c & \text{if } x_i > a. \end{cases}$$

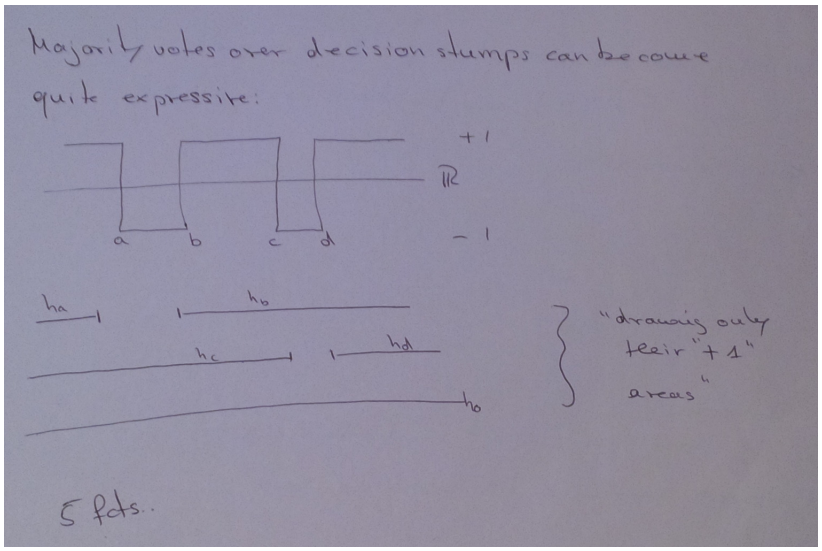
Finding general decision stump of minimal empirical error can be done by d linear scans over over the data set of size n (one for each dimension, keeping track of the current loss minimizer). Overall, this takes computation time $O(dn)$. For more details on this, see also Section 10.1.1 in the textbook [1].

2.3 Weighted Majority Votes

Boosting outputs a classifier that is a weighted majority vote over a set of classifiers that the weak learner output. Recall that, for some class $\mathcal{H}$ weighted T-majority votes over $\mathcal{H}$ are defined as follows:

$$\mathrm{MV}(\mathcal{H}, T) \;=\; \left\{ h \in \{0,1\}^{\mathcal{X}} \;:\; \exists h_1, \ldots h_T \in \mathcal{H}, \boldsymbol{w} \in \mathbb{R}^T \text{ such that } h(x) = \operatorname{sign}\left(\sum_{i+1}^{T} w_i h_i(x) \right) \right\}$$

Clearly, the original class $\mathcal{H}$ is included in the class of T majority votes over $\mathcal{H}$ for any positive integer T. Taking weighted majority votes over some class can yield a significantly larger (and hence more expressive) class of predictors. For example, any function over $\mathbb{R}$ that switches between intervals of value $+1$ and -1 can be realized by a majority votes over signed, one-dimensional decisions stumps. See figure below for an illustrative example.



On the other hand, we have shown in the tutorials, that the VC-dimension of weighted T-majority votes is bounded as

$$\mathrm{MV}(\mathcal{H}, T) \;\leq\; T(\mathrm{VC}(\mathcal{H}) + 1)(3\log(\mathrm{VC}(\mathcal{H}) + 1) + 2) = O(T(\mathrm{VC}(\mathcal{H}))\log(T(\mathrm{VC}(\mathcal{H}))))$$

That is, even though, the class becomes more expressive, we can still control its complexity in terms of the VC-dimension of the original class and the number T of classifiers used in the majority vote. Thus, there is hope that we can control the generalization properties of an algorithm that outputs weighted majority votes, such as boosting.

2.4 Adaboost

We are now ready to describe the Adaboost algorithm. Adaboost proceeds in rounds. It is given an (initially unweighted) sample $S = ((x_1, y_1), \ldots, (x_n, y_n))$ and a hypothesis class $\mathcal{H}$. Further, it is assumed that Adaboost has access to a weak learner for $\mathcal{H}$. We denote this weak learner by $\mathrm{WL}_{\mathcal{H}}$.

In each round, the weak learner is invoked on a weighed version of the sample S. We denote the weight of sample (x_i, y_i) in round t by D_i^t. Note that the samples S with weights D_i^t can be viewed as a distribution D^t over the sample points. The loss of a function h with respect to this distribution is defined as

$$L_{D^t}(h) = \sum_{i=1}^{n} D_i^t \mathbf{1}[h(x_i) \neq y_i]$$

The weak learner is called with respect to this distribution and outputs a predictor h_t in round t.

Then, before the next round, the points on which h_t errs receive heavier weights under D^{t+1} whereas the points on which h_t is correct, will receive lighter weights. Thus, in the next round, the weak learner will output a hypothesis h_{t+1} that "focuses on getting the previous mistakes corrected".

Formally, Adaboost can be stated as follows:

Algorithm 1 Adaboost

Input $S = ((x_1, y_1), \ldots, (x_n, y_n))$, number of rounds T, weak learner for $\mathcal{H}$
Initialize: $D_i^1 = \frac{1}{n}$ for all i
for $t = 1, \ldots, T$ **do**
  $h_t = \mathrm{WL}_{\mathcal{H}}(D^t, S)$
  $\epsilon_t = \sum_{i=1}^{n} D_i^t \mathbf{1}[h(x_i) \neq y_i]$
  $w_t = \frac{1}{2} \log\left(\frac{1}{\epsilon_t} - 1\right)$
  $D_i^{t+1} = \frac{D_i^t \mathrm{e}^{-(w_t y_i h_t(x_i))}}{\sum_{j=1}^{t} D_j^t \mathrm{e}^{-(w_t y_j h_t(x_j))}}$
end for
return $h_{\mathrm{ada}} = \mathrm{sign}\left(\sum_{t=1}^{T} w_t h_t\right)$

We now show that, under the assumption that the weak learner is successful (as in Definition 2), the empirical loss of the combined weighted majority vote decreases exponentially fast with the number of rounds in Adaboost (that is, with the number of predictors that go in to the majority vote).

Theorem 1. *Assume that at each round of Adaboost we have* $\epsilon_t \leq \frac{1}{2} - \gamma$ *for some* $\gamma > 0$. *Then, the training error of the output of Adaboost is bounded by*

$$L_n(h_{ada}) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}[h(x_i) \neq y_i] \leq \mathrm{e}^{-2\gamma^2 T}$$

Proof. The proof can be found as proof of Theorem 10.2 in [1]. □

2.5 Discussion

Given some base hypothesis class $\mathcal{H}$, Adaboost outputs a function from the class of weighted majority votes $\mathrm{MV}(\mathcal{H}, T)$. We thus, get the following generalization bound for the output classier of Adaboost:

$$L(h_{ada}) \leq L_n(h_{ada}) \; + \; C\sqrt{\frac{\mathrm{VC}(\mathrm{MV}(\mathcal{H}, T)) + \ln 1/\delta}{n}}, \tag{6}$$

(see Recap Section). According to Theorem 1, the empirical loss $L(h_{ada})$ decreases exponentially fast with T. That is, if $T \geq C \log(n)$ (for some constant C), then the empirical loss becomes less than $1/n$, which means that all training examples are classified correctly. Meanwhile the estimation error part in the above bound becomes

$$C\sqrt{\frac{\mathrm{VC}(\mathrm{MV}(\mathcal{H}, \log(n))) + \ln 1/\delta}{n}},$$

which is roughly

$$C\sqrt{\frac{\log(n)(\mathrm{VC}(\mathcal{H}))\log(\log(n)(\mathrm{VC}(\mathcal{H}))) + \ln 1/\delta}{n}},$$

which is then a bound on the generalization error of the output of Adaboost and will decrease to 0 for large enough n.

However, note that this consideration is based on the assumptions of Theorem 1, that is, it is based on the existence of a weak learner. Since we can not expect that, for any hypothesis class and any distribution, taking weighted majority votes as in boosting will eventually lead to realizability, this assumption and so the above analysis should be taken with a grain of salt. Nevertheless, boosting has proven itself to be very successful in practice. Thus, in practice, we observe, that we don't need the full strength of weak learnability as in Definition 2 for boosting to be a successful algorithm.

References

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