1 Lecture 8: 2017.03.20: Boosting

Recall the PAC-learning scheme:

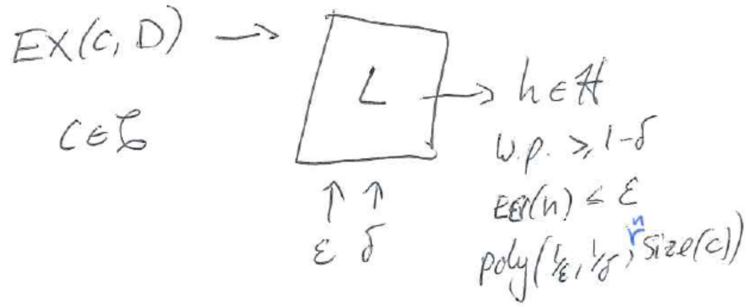

Figure 1: Scheme for PAC learning.

Hypothesis/Accuracy Boosting question:

If we have an algorithm for PAC-learning some class $\mathcal{C}$ but only for fixed ε_0, does this imply a full PAC algorithm (i.e. $\varepsilon \to 0$)

Remark 1.1. *Today we see that the answer to this question is Yes. By the remark above, this implies that the proof must use the fact that we are allowing* any *distribution.*

1.0.1 Confidence Boosting

- Given algorithm L that can achieve error ε ($\varepsilon \to 0$), but only for some *fixed* value of $\delta = \delta_0$ (e.g. $\delta_0 = \frac{9}{10}$) w.p. $1 - \frac{9}{10} = \frac{1}{10}$
- Run L k times, we get k hypothesis $h_1, \ldots, h_k$, which are independent. Note that each hypothesis has a chance $\frac{1}{10}$ of having error less than ε_0 so if k is big enough, choosing the one with the best error has as high probability as we want to achieve error less than ε_0. (It is basically the minimum of k independent Bernoulli trials).

$$\mathbb{P}_{S_1,\ldots,S_k \sim D}\left[\forall 1 \leq i \leq k, \quad Err(h_i) > \varepsilon\right] \leq \delta_0^k, \tag{1.1}$$

which is less than any chosen δ if k is big enough. It is enough to pick $k \geq \frac{\log\left(\frac{1}{\delta}\right)}{\log\left(\frac{1}{\delta_0}\right)}$

1.0.2 Accuracy Boosting

Given algorithm L such that for any distribution, w.p $\geq 1-\delta$, outputs h such that $Err(h) \leq \varepsilon_0 = \frac{1}{2} - \gamma$ (i.e. slightly better than random guessing) (weak learning algorithm). One my try a similar thing as above but with taking a majority vote:

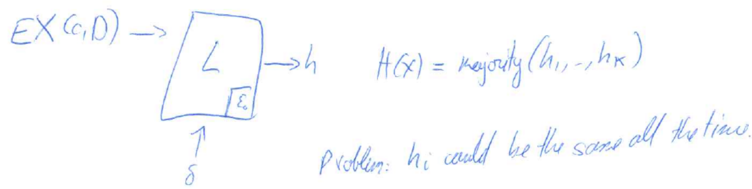

Figure 2: Majority vote.

The problem is that L could be evil and always output the same hypothesis. The key idea is to create filtered distributions to force L to learn something "new".

1.1 "Original" Boosting construction (Schapire)

1. Call weak learning algorithm L on $D_1 = D$ and we get h_1 such that $Err_{D_1}(h_1) \leq \varepsilon_0$

2. We create a new distribution, D_2. To sample from D_2:

   - Flip a fair coin.
   - If heads, sample $x \sim D_1$ until $h_1(x) \neq c(x)$.
   - If tails, sample until $h_1(x) = c(x)$.

   Remark 1.2. *Note that if these conditions are not met for a lot of samples, then either h_1 or $\neg h_1$ are already good enough hypothesis. This guarantees that L cannot output h_1 or its negation as hypothesis because they would have error 50/50.*

   Algebraically, this corresponds to modifying the weights of the original distribution, see the picture below.

   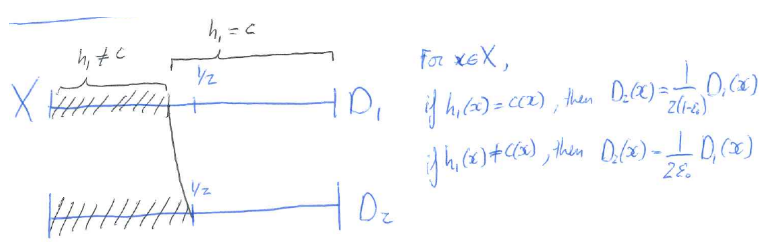

   Figure 3: Modify weights of original distribution.

   Then we run L on $EX(c, D_2)$ to get some $h_2 \neq h_1$ such that $Err_{D_2}(h_2) \leq \varepsilon_0$.

3. Define a distribution D_3. To sample from D_3:

   - Draw $x \sim D_1$ until $h_1(x) \neq h_2(x)$. We will quickly get such a an x by construction.
   - Create labeled example $\langle x, c(x) \rangle$.

   Run L on $EX(c, D_3)$ to get h_3 such that $Err_{D_3}(h_3) \leq \varepsilon_0$.

4. Final hypothesis, for any x, $h(x) = \text{majority}\{h_1(x), h_2(x), h_3(x)\}$.

Lemma 1.3. $Err_{D_1}(h) \leq 3\varepsilon_0^2 - 2\varepsilon_0^3$.

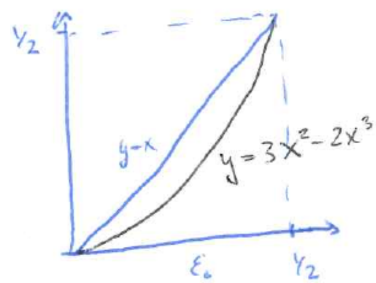

Figure 4: $3x^2 - 2x^3$.

Remark 1.4. *Note that it is a convex function below the line $\{y = x\}$. So we gain a little bit by boosting. Then we can iterate to obtain an arbitrary boosting.*

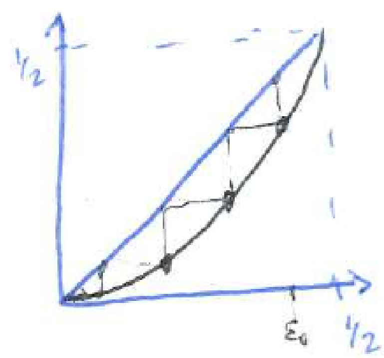

Figure 5: Boosting iterations.

Also note that the number of iterations needed to obtain error $\leq \varepsilon$ is $\sim \log\log\frac{1}{\varepsilon}$.

We are actually changing the hypothesis space in the final algorithm. We output a ternary tree of majority hypotheses

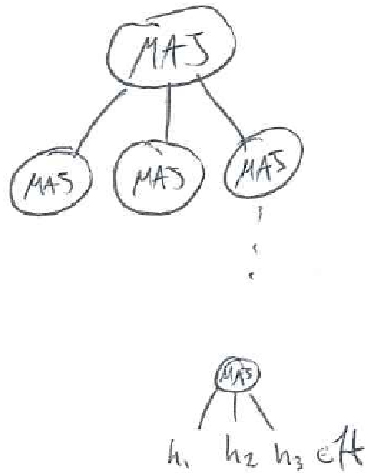

Figure 6: Majority tree.

1.2 Adaboost (Freund/Schapire)

- View input $S = (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \sim EX(c, D)$. Assume that $y_i \in \{-1, 1\}$. We want to use the weak learning algorithm to find a hypothesis that is consistent with S, recall that as we saw before this is enough for PAC learning.
- Start with D_1 the uniform distribution on S. If we can get h such that $Err_{D_1}(h) < \frac{1}{m}$, then h is consistent with S and apply VC-theory.
- Let's look at the "code", recall that the distributions refer to S.

  $D_1 = \text{Unif}(S)$

  for t in range $1, \ldots, T$

  - run weak learning algorithm L using D_t to get some $h_t in \mathcal{H}$
  - choose a weight $\alpha_t >= 0$ for the hypothesis h_t (See analysis below)
  - define the next distribution D_{t+1}

    $$D_{t+1}(x_i, y_i) = D_t(x_i, y_i) e^{-\alpha_t y_i h_t(x_i)} Z_t, \tag{1.2}$$

    where Z_t is the normalization factor.

    Remark 1.5. *Note that $y_i h_t(x_i)$ is 1 if they agree or -1 otherwise. This means that we are increasing the weight on the places where h_t was wrong and reducing the weight on the ones where it was right.*

  Final classifier: $h(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$

Analysis

Remark 1.6 (Notation)**.** *Let $\varepsilon_i := \mathbb{P}_{i \sim D_t}[h_t(x_i) \neq y_i]$ and $\gamma_t := \frac{1}{2} - \varepsilon_t$ "advantage" of h_t over random guessing.*

If $y_i \neq H(x_i) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x_i)\right)$, then $y_i \sum_{t=1}^{T} \alpha_t h_t(x_i) \leq 0$, which implies that $e^{-\sum_{t=1}^{T} \alpha_t h_t(x_i)} \geq 1$. So

$$\frac{1}{m}\left|\{i : H(x_i) \neq y_i\}\right| \leq \frac{1}{m}\sum_i e^{-\sum_t \alpha_t h_t(x_i)}. \tag{1.3}$$

Take any fixed i, we can measure how much the distribution changes from t to $t+1$

$$Z_t = \frac{D_t(x_i, y_i)e^{-\alpha_t y_i h_t(x_i))}}{D_{t+1}(x_i, y_i)} \tag{1.4}$$

Now we want take a look at how much the distribution has changed over t

$$\prod_t^T Z_t = \frac{D_1(x_i, y_i)}{D_{T+1}(x_i, y_i)} e^{-y_i \sum_t \alpha_t h_t(x_i)} = e^{-y_i \sum_t \alpha_t h_t(x_i)}, \tag{1.5}$$

since the Z_t's are independent of i so must be the right-hand-side above. We obtain,

$$\prod_t^T Z_t = \frac{m}{m}\prod_t^T Z_t = \frac{1}{m}\sum_i e^{-y_i \sum_t \alpha_t h_t(x_i)}, \tag{1.6}$$

so to estimate the error we only need to analyze the Z_t's and recall that we are still free to choose the weights α_t's

$$Z_t = \sum_i D_t(x_i, y_i)e^{-\alpha_t y_i h_t(x_i)}, \tag{1.7}$$

therefore, we can choose $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$ to obtain

$$Z_t = (1-\varepsilon_t)e^{-\alpha_t} + \varepsilon_t e^{\alpha_t} = \cdots \leq 2\sqrt{\varepsilon_t(1-\varepsilon_t)}. \tag{1.8}$$

Thus, we can bound the training error

$$\text{training error} \leq \prod_t 2\sqrt{\varepsilon_t(1-\varepsilon_t)} = \prod_t 2\sqrt{1-4\gamma_t^2} \leq e^{-2\sum_t \gamma_t}. \tag{1.9}$$

We have arrived at the following result.

Theorem 1.7. *Our training error on the original S is bounded above by $e^{-2\sum_t \gamma_t}$.*