Machine Learning: Boosting

Mahbub Majumdar

BRAC University
66 Mohakhali
Dhaka, Bangladesh

December 8, 2019

Boosting

- We will illustrate machine learning through a special example.
- Boosting is a theoretical idea implemented by machine learning pioneers that achieved dramatic success.
- Its an example of where theory played a driving role.
- It was used to first classify objects reliably and computationally cheaply.

Sacrifice Accuracy

Main theme: sacrifice accuracy for simplicity

- By not requiring too much accuracy, we hope to gain in computational complexity.
- We measure the accuracy of a predictor by ϵ.
- We will accept weak learners with accuracy $\epsilon \sim .45\%$.
- We will then "boost" these weak learners to become strong learners with $\epsilon \ll 1$.

Boosting ϵ

- We want to boost a "large" $\epsilon = \frac{1}{2} - \gamma$.
- Here, $\gamma \sim$ small.

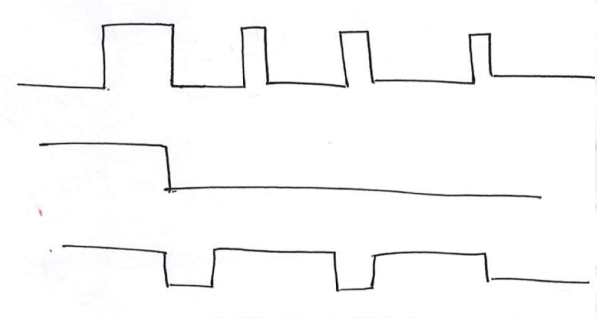


Figure: a) True data, b) weak predictor, c) boosted predictor

Formal Definition of Learning

Definition: Strong Learner

A hypothesis class $\mathcal{H}$ is (PAC) learnable if there exists a $m_{\mathcal{H}}$, and a learning algorithm with following property: For every $\epsilon, \delta \in (0, 1)$, for every distribution D over the domain set X, and for every labeling function, $f : X \to \pm 1$, then when running the learning algorithm on $m \geq m_{\mathcal{H}}$ iid samples and assuming *realizability*, the algorithm returns a hypothesis h such that with probability of at least $1 - \delta$, that the true error satisfies

$$L_{D,f}(h) \leq \epsilon$$

- A strong learner can choose ϵ to as small as they want.
- For a weak learner, the only change is that

$$L_{D,f}(h) \leq \frac{1}{2} - \gamma$$

Basic Hypotheses Classes

- The goal is therefore to look for *basic* hypotheses classes B, that can be efficiently implemented.
- We will apply Empirical Risk Minimization (minimizing the error on sampled data) to B.
- For this to work, we require
  - $\rightarrow$ ERM_B is efficiently implementable
  - $\rightarrow$ Any ERM_B hypothesis with have an error of at most $\frac{1}{2} - \gamma$.
- We will show that it is possible to boost efficient weak learners using B

Example: Decision Stumps

- Let's consider a simple example. Suppose, we divide up the real line into three pieces

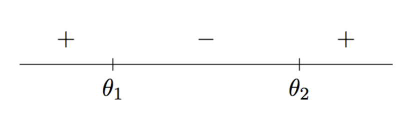
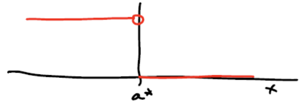


Figure: We want to learn the distribution of $+$'s and -'s.

- Let's use the basic threshold functions to learn the hypothesis h/distribution of $+$'s and -'s shown above.

- Threshold functions are simple predictors



Example: Decision Stumps

- Consider the class $\mathcal{H}$ of 3 piece classifiers

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

- In the first figure in the previous slide, $b = 1$.
- We want to learn $\mathcal{H}$ using the Basic class of decision stumps.
- Decision stumps are generalizations of threshold functions

$$B = \{\mathrm{sign}(x - \theta) \cdot b \,|\, \theta \in \mathbb{R}, b \in \{\pm 1\}\}$$

- There will be a decision stump that is wrong one only one of the regions of Figure 6
- Now associate a probability weight to each of the 3 regions of Figure 6.
- At least one of the regions will have a probability weight less than $1/3$.

Example: Decision Stumps

- Choose threshold functions that are wrong on this region with lowest probability weight.
- Then by picking enough samples, we will be able to find a decision stump with error of at most $\frac{1}{3} + \epsilon$.
- Choose $\epsilon = \frac{1}{12}$.
- Then the error of ERM_B is $\frac{1}{3} + \frac{1}{12} = \frac{1}{2} - \frac{1}{12}$.
- Thus, ERM_B is a weak learner for $\mathcal{H}$, where $\gamma = \frac{1}{12}$, and $\mathcal{H}$ are the 3-piece predictors.

Adaptive Boosting: Adaboost

- Suppose our weak learner outputs a hypothesis h_t.
- Then we calculate the error of h_t.
- At each step t, the booster defines a distribution D_t over the samples S. (Normalization: $\sum_{i=1}^{m} D_t(i) = 1$).
- We do this by weighting each point with a distribution D_t.
- The true error is defined to be

$$L_{D_t}(h_t) = \sum_{i}^{m} D_t(i) \mathbb{1}_{\{h_t(x_i) \neq y_i\}}$$

- We focus on the points h_t got wrong.

Adaptive Boosting: Adaboost

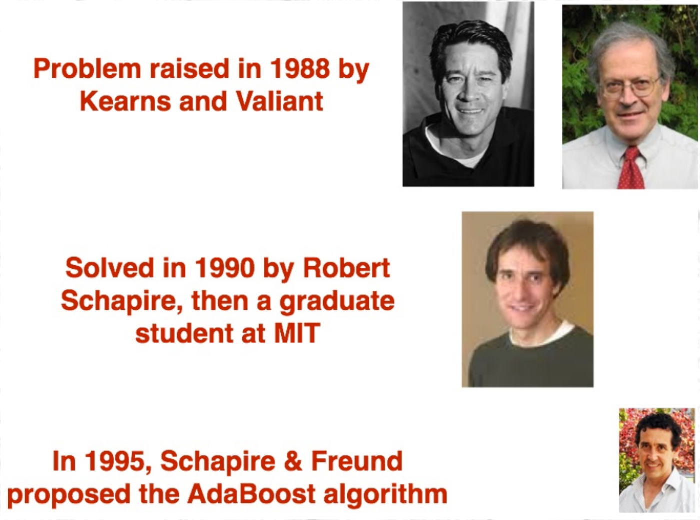


Figure: History of Adaboost

Adaboost

- To focus on the mistakenly labelled points, we modify/adapt the distribution D_{t+1} to have more weight on the "mistaken" points, and less weight on the correctly labeled points.
- We then run the ERM procedure using our new definition for true error.
- This will output a new predictor h_{t+1}.
- We now perform this iteration T times.
- We now have classifiers, $h_1, \ldots, h_t, h_{t+1}, \ldots, h_T$.

- Instead of choosing the last classifier h_T, we form a signed linear combination of the h_t.
- Each h_t is weighted by a weight factor w_t.
- Adaboost corresponds a particular way of assigning weights w_t and updating the distribution D_t.
- The boosted class is denoted $L(B, T)$

$$L(B,T) = \left\{ \text{sign}\left(\sum_{t=1}^{T} w_t h_t(x) \right) \middle| w \in \mathbb{R}^T, h_t \in B \right\}$$

Pseudocode for Adaboost

Input

Training set $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$

Weak Learner, WL

Number of iterations, T

Initialize

$D_1 = (\frac{1}{m}, \ldots, \frac{1}{m})$.

for $t = 1, \ldots, T$:

Call weak learner $h_t = \mathrm{WL}(D_t, S)$

Compute the error $\epsilon_t = \sum_i^m D_t(i) \mathbb{1}_{\{h_t(x_i) \neq y_i\}}$

Set the weight $w_t = \frac{1}{2} \log \left(\frac{1}{\epsilon_t} - 1 \right)$

Update for all $i \in \{1, \ldots, m\}$

$$D_{t+1}(i) = \frac{D_t(i) e^{-w_t y_i h_t(x_i)}}{\sum_{j=1}^m D_t(j) e^{-w_t y_j h_t(x_j)}}$$

- **Output**

The classifier $h_S(x) = \mathrm{sign} \left(\sum_{t=1}^T w_t h_t(x) \right)$

Adaboost is a Strong Learner

- We can make the sample error as small as we want for the boosted class $L(B, T)$ using Adaboost.

Theorem: Adaboosts Converges

Let S be a training set and assume that at each iteration of Adaboost, the weak learner returns a hypothesis for which $\epsilon_t \leq \frac{1}{2} - \gamma$. Then the training error of the output hypothesis of Adaboost is at most

$$L_S(h_S) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{\{h_S(x_i) \neq y_i\}} \leq e^{-2\gamma^2 T}$$

Proof: CSE427

People/Thing Discrimination using Adaboost

- Viola and Jones used basic hypothesis classes to tell if an object is a person or non-person.

- They used aligned-axis rectangles.

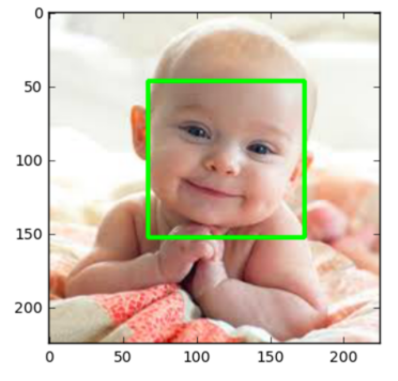


Figure: Aligned-axis rectangle used as a basic hypothesis class.

People/Thing Discrimination using Adaboost

- They then boosted this class to create a strong learner.
- We want to generate a strong classifier that outputs a 1 if the object is a person, and -1 otherwise.
- The images taken were 24×24 pixels.
- The number of aligned axis rectangles is 24^4

- Four types of rectangles were used, $t \in \{A, B, C, D\}$. They are called masks.

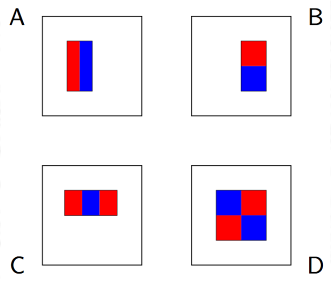


Figure: 4 different masks

- The greyscale values of the rectangles are calculated.
  1. A – blue minus red
  2. B – blue minus red
  3. C – blue minus red
  4. D – blue minus red

People/Thing Discrimination using Adaboost

- These masks are meant to capture these basic rules of thumb
    1. The nose region is darker than the cheek regions.
    2. The eyebrow region is darker than lower part of the face.

Figure: How the masks are chosen.

- Call the function taking an image/rectangle to its subtracted greyscale value g. Then

$$g : \mathbb{R}^{24,24} \to \mathbb{R}$$

People/Thing Discrimination using Adaboost

- We now boost construct another base class using decision stumps f, on $g(x)$.
- Thus our base class is $h(x) = f(g(x))$.
- We then use Adaboost to boost to boost the h.
- Since this is now a strong learner, we can make the estimation error as small as needed.

What's Left?

Much more to say...

- Computational complexity of decision stumps/half-spaces
- VC-Dimension
- How T allows Adaboost to directly tradeoff estimation error with approximation error.
- Much more...
- To learn more, come to CSE427.