# Machine learning
# *Theoretical Views of Boosting*

Riyane SID-LAKHDAR

October 24, 2016

**Abstract**

In this document, we refer to the paper of Robert E. Schapir [1] in order to prove that the AdaBoost algorithm is indeed a "boosting" algorithm. Our demonstration is based on the steps described on slide 50 of the lecture (all the questions have been considered and fulfilled).

## Contents

# 1 Motivation and scope of the study

## 1.1 AdaBoost algorithm: An example of boosting algorithm

The AdaBoost algorithm belongs to the set of boosting algorithms. This kind of learning algorithms are based on the following principle: Assuming a set of input weak classifier (with relatively high respective empirical risks), it builds an output classifier with a lower empirical risk. In this document, we prove that the empirical risk of the AdaBoost output classifier drops exponentially with the number of weak classifiers (assuming each weak classifier is slightly better than random).

## 1.2 Scope and limits

In the paper of Robert E. Schapir [1], an implementation of the AdaBoost algorithm is presented. This algorithm only considers building binary classifiers (using vectors with output set $= [+1, -1]$).

One of the limits of the current study is that it only considers the evolution and the convergence of the empirical risk of the output classifier on the considered training set. We do not consider any aspect of its generalization to the hall set of possible input points (generalization error).

## 1.3 Used notations

In the rest of the document, we use the following notations.

- $m$: Size of the training set.
- $T$: Number of iterations (or classifiers to be combined) in the AdaBoost algorithm execution.
- $H(x)$: Prediction of the output AdaBoost classifier for an input x.
- $F(x)$: Prediction of the output AdaBoost classifier for an input x, transposed to the binary output space $[+1, -1]$ ($F(x) = sign(H(x))$).
- $D_t(i)$: Distribution probability of the training point i at the iteration t.
- $\alpha_t$: weight of the $t^{th}$ weak classifier in the output classifier of the AdaBoost algorithm.
- $Z_t$: Normalisation term of the weak classifier t, with respect to the combination weights.
- $1_{E_0 rel E_1}$ is the function that associates to the elements $E_0$ and $E_1$ the value 1 if the relation $rel$ between $E_0$ and $E_1$ is verified, and 0 otherwise.

# 2 Analyse of the AdaBoost algorithm

## 2.1 Bounding the training error

In this section, we prove that the empirical risk $\sum_{i=1}^{m} 1_{y_i \neq F(x_i)}$ of a binary classifier $F$ computed by the AdaBoost algorithm, is bounded as follows:

$$\frac{1}{m} * \sum_{i=1}^{m} 1_{y_i \neq F(x_i)} \leq \prod_{t=1}^{T} Z_t \tag{1}$$

First, we can notice that for a classifier H computed by the AdaBoost algorith and for each training sample $(x_i, y_i)$, either

- $y_i = sign(H(x_i))$. Thus $\exp^{-y_i H(x_i)} \in ]0, 1[$. Hence $1_{y_i \neq H(x_i)} = 0 \leq \exp^{-y_i H(x_i)}$

or

- $y_i \neq sign(H(x_i))$. Thus $\exp^{-y_i H(x_i)} \geq 1$. Hence Hence $1_{y_i \neq H(x_i)} = 1 \leq \exp^{-y_i H(x_i)}$

Thus, $\forall i \in [1, m], 1_{y_i \neq sign(H(x))} \leq \exp^{-y_i H(x_i)}$. As each of this elements is positive and as m is positive, we have

$$\frac{1}{m} * \sum_{i=1}^{m} 1_{y_i \neq F(x_i)} \leq \frac{1}{m} * \sum_{i=1}^{m} \exp^{-y_i H(x_i)} \tag{2}$$

Meanwhile, we can notice by definition of the function H that for each training sample $(x_i, y_i)$

$$\begin{aligned}\sum_{i=1}^{m} \exp^{-y_i H(x_i)} &= \sum_{i=1}^{m} \exp^{-y_i \sum_{t=1}^{T} \alpha_t f_t(x_i)} = \sum_{i=1}^{m} \prod_{t=1}^{T} \exp^{-y_i \alpha_t f_t(x_i)} \\ &= \sum_{i=1}^{m} [\exp^{-y_i \alpha_1 f_1(x_i)} * \prod_{t>1}^{T} \exp^{-y_i \alpha_t f_t(x_i)}]\end{aligned}$$

We can also notice by definition of the AdaBoost algorithm that $\forall i \in [1, m]$

$$\begin{aligned} D_2(i) &= \frac{D_1(i) \exp^{-\alpha_1 y_i * f_1(x_i)}}{Z_1} = \frac{\frac{1}{m} \exp^{-\alpha_1 y_i * f_1(x_i)}}{Z_1} \\ &Hence \\ \exp^{-\alpha_1 y_i * f_1(x_i)} &= m * Z_1 * D_2(i)\end{aligned}$$

Thus

$$\begin{aligned}
\frac{1}{m}\sum_{i=1}^{m}\exp^{-y_iH(x_i)} &= \sum_{i=1}^{m}[Z_1 * D_2(i) * \prod_{t>1}^{T}\exp^{-y_i\alpha_t f_t(x_i)}] \\
&= \sum_{i=1}^{m}[Z_1 * D_2(i) * \prod_{t>1}^{T}[D_{t+1}(i)Z_t]] \text{ by definition of } D_t \\
&= \sum_{i=1}^{m}\prod_{t=1}^{T}[D_{t+1}(i)Z_t] = \sum_{i=1}^{m}[[\prod_{t=1}^{T}Z_t] * [\prod_{t=1}^{T}D_{t+1}(i)]] \\
&= \prod_{t=1}^{T}Z_t * \sum_{i=1}^{m}\prod_{t=1}^{T}D_{t+1}(i) \\
\frac{1}{m}\sum_{i=1}^{m}\exp^{-y_iH(x_i)} &= \prod_{t=1}^{T}Z_t \text{ See: }^{2}
\end{aligned}$$

Finally, by combining the previous equation with the inequality 2, we can reach the announced equation 1.

## 2.2 Minimizing the upper bound of the training error

Thanks to the equation 1, we know that for a given set $\{\alpha_t\}_{t\in[1,T]}$, an upper bound of the training error of the AdaBoost algorithm is $\prod_{t=1}^{T}Z_t$. Thus, in order to minimize this training error, we could find the set $\{\alpha_t\}_{t\in[1,T]}$ that minimizes $\prod_{t=1}^{T}Z_t$.
For each $t \in [1,T]$, we know by definition that $Z_t$ is positive. Thus, in this section, we minimize $\prod_{t=1}^{T}Z_t$ by minimizing $Z_t$ for each t (not the optimal way).
First, by definition of $Z_t$, we have

$$\begin{aligned}
Z_t(\alpha_t) = \sum_{i=1}^{m}D_t(i)\exp^{-\alpha_t y_i f_t(x_i)} &= \sum_{\substack{i=1\\y_i=f_t(x_i)}}^{m}D_t(i)\exp^{-\alpha_t y_i f_t(x_i)} + \sum_{\substack{i=1\\y_i\neq f_t(x_i)}}^{m}D_t(i)\exp^{-\alpha_t y_i f_t(x_i)} \\
&= \sum_{\substack{i=1\\y_i=f_t(x_i)}}^{m}D_t(i)\exp^{-\alpha_t} + \sum_{\substack{i=1\\y_i\neq f_t(x_i)}}^{m}D_t(i)\exp^{\alpha_t} \text{ See: }^{1} \\
&= \exp^{-\alpha_t}\sum_{\substack{i=1\\y_i=f_t(x_i)}}^{m}D_t(i) + \exp^{\alpha_t}\sum_{\substack{i=1\\y_i\neq f_t(x_i)}}^{m}D_t(i) \\
Z_t(\alpha_t) &= (1-\epsilon_t)\exp^{-\alpha_t} + \epsilon_t\exp^{\alpha_t} \text{ By definition of } \epsilon_t
\end{aligned} \tag{3}$$

[2] The equality $\sum_{i=1}^{m}\prod_{t=1}^{T}D_{t+1}(i) = 1$ is obviously wrong (some of the products of a non constant probability). The error that occurs here comes probably from the fact that the parameter t is not well bounded in this expression.

Second, we can notice on the equation 3 that the function $Z_t$ is derivable with respect to $\alpha_t$ ($\epsilon_t$ does not depend on $\alpha_t$ and the exponential function is derivable on its hole definition interval). As $Z_t$is continue, the values $\alpha_t$ $Z_t$ reaches an extremum are the solutions of the equation

$$\begin{aligned} \frac{dZ_t}{d\alpha_t}(\alpha_t) &= 0 \\ \epsilon_t \exp^{\alpha_t} + (\epsilon_t - 1)\exp^{-\alpha_t} &= 0 \\ \epsilon_t + (\epsilon_t - 1)\exp^{-2\alpha_t} &= 0 \text{ cause } \exp^{\alpha_t} \neq 0 \\ \exp^{-2\alpha_t} &= \frac{\epsilon_t}{1-\epsilon_t} \\ -2\alpha_t &= ln(\frac{\epsilon_t}{1-\epsilon_t}) \\ \alpha_t &= \frac{-1}{2}ln(\frac{\epsilon_t}{1-\epsilon_t}) = \frac{1}{2}ln((\frac{\epsilon_t}{1-\epsilon_t})^{-1}) \\ \alpha_t &= \frac{1}{2}ln(\frac{1-\epsilon_t}{\epsilon_t}) \end{aligned} \tag{4}$$

As this solution is unique, the function $Z_t$ has a unique extremum reached for

$$\alpha_t = \alpha_t^* = \frac{1}{2}ln(\frac{1-\epsilon_t}{\epsilon_t}). \tag{5}$$

Furthermore,

$$\begin{aligned} \frac{dZ_t}{d\alpha_t}(\alpha_t) \underset{\alpha_t \to +\infty}{\approx} & \quad \exp^{\alpha_t} - \exp^{-\alpha_t} \text{ Cause } \epsilon_t \in ]0,1[ \text{ , hence } \epsilon_t - 1 < 0 \\ \underset{\alpha_t \to +\infty}{\approx} & \quad +\infty - 0 = +\infty \end{aligned} \tag{6}$$

Thus, as $Z_t$ is continue with respect to $\alpha_t$, we can deduce from 5 and 6 that $\alpha_t = \alpha_t^* = \frac{1}{2}ln(\frac{1-\epsilon_t}{\epsilon_t})$ is a minimum of $Z_t$.

## 2.3 Minimizing the training error

By choosing $\alpha_t = \alpha_t^*$ we are not sure to reach the minimal value of the training error. However, we are sure that the training error will be smaller that $Z_t(\alpha_t^*)$. In this section, we choose $\alpha_t = \alpha_t*$ and we show that if the empirical error $\epsilon_t$ of each weak classifier is slightly better than random ($\epsilon_t < 1/2$) then the training error decreases exponentially to 0 with the number of weak classifiers (T).
In this section we also suppose proved the result $(6) : \forall t \in [1,T], Z_t = \sqrt{1 - 4\gamma_t^2}$.

[1] If $y_i == f_t(x_i)$ then $y_i f_t(x_i) = 1$, else $y_i f_t(x_i) = -1$

First, let’s prove, by reduction to the absurd, that $\sqrt{1-4\gamma_t^2} \leq \exp^{-2\gamma_t^2}$

$$\begin{aligned} \sqrt{1-4\gamma_t^2} &> \exp^{-2\gamma_t^2} \\ \exp^{\frac{1}{2}ln(1-4\gamma_t^2)} &> \exp^{-2\gamma_t^2} \text{ Exponential form of the power function} \\ \frac{1}{2}ln(1-4\gamma_t^2) &> -2\gamma_t^2 \text{ Cause exp continue and strictly increasing} \\ ln(1-4\gamma_t^2) &> -2\gamma_t^2 \\ 1-4\gamma_t^2 &> \exp^{-2\gamma_t^2} \end{aligned} \tag{7}$$

Which is absurd because $\forall x \in \mathbb{R}$, the function $x \rightarrow (1-4x)$ is greater or equal than the exponential function.
Thus, $\forall t \in [1, T], Z_t \leq \exp^{-2\gamma_t^2}$.
Furthermore, we can notice that:

$$\begin{aligned} \prod_{t=1}^{T} \sqrt{1-4\gamma_t^2} = \prod_{t=1}^{T} \exp^{\frac{1}{2}ln(1-4\gamma_t^2)} &= \exp^{\frac{1}{2}\sum_{t=1}^{T} ln(1-4\gamma_t^2)} \\ = \exp^{\frac{1}{2}ln(\prod_{t=1}^{T} 1-4\gamma_t^2)} &= \exp^{\frac{1}{2}\sum_{t=1}^{T} ln(Z_t^2)} \\ = \exp^{\sum_{t=1}^{T} ln(Z_t)} & \end{aligned}$$

But we know that $\forall x, y \in \mathbb{R}^+$ , $x <= y \Rightarrow ln(x) \leq ln(y)$. Thus 7 induce: $\sum_{t=1}^{T} ln(Z_t) \leq \sum_{t=1}^{T} ln(\exp^{-2\gamma_t^2}) = \sum_{t=1}^{T} -2\gamma_t^2$. Thus:

$$\begin{aligned} \prod_{t=1}^{T} \sqrt{1-4\gamma_t^2} \leq \exp^{\sum_{t=1}^{T} -2\gamma_t^2} &= \prod_{t=1}^{T} \exp^{-2\gamma_t^2} \\ \prod_{t=1}^{T} \sqrt{1-4\gamma_t^2} \leq \exp^{\sum_{t=1}^{T} -2\gamma_t^2} &= \prod_{t=1}^{T} \exp^{-2\gamma_t^2} \\ Hence : & \\ EmpiricalMisclassification \leq m \prod_{t=1}^{T} Z_t \leq m \prod_{t=1}^{t} \exp^{-2\sum_{t=1}^{T}\gamma_t^2} & \end{aligned} \tag{8}$$

# 3 Interpretation

Let assume that $\alpha_t = \alpha_t^*$ for each step t of the AdaBoost algorithm. Thanks to the equation 8, we can deduce that each time we process an addition step t in the AdaBoost algorithm, the empirical risk is divided by $E_t = \exp^{1-2\epsilon_t}$. Thus if the empirical risk of the weak classifier t is slightly better than random ($\epsilon_t < \frac{1}{2}$), $E_t > 1$. In this conditions, he training error of the AdaBoost output classifier drops exponentially fast with the number of iterations.

The AdaBoost algorithm may then be described as a boosting algorithm: it takes a set of input weak classifiers (with respective empirical risks slightly better than random) and outputs a classifier with a much better (proportionally) empirical risk.

## References

[1] R. E. Schapire. Theoretical views of boosting and applications. In *International Conference on Algorithmic Learning Theory*, pages 13–25. Springer, 1999.