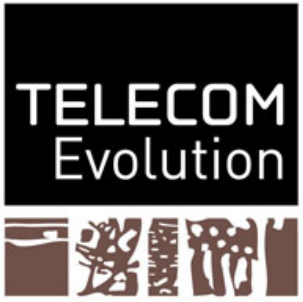


Boosting

Florence d'Alché-Buc,
florence.dalche@telecom-paristech.fr

CES Data Scientist

Outline

Motivation

Boosting
- AdaBoost as a Greedy Scheme
- Gradient Boosting

References

Outline

Motivation

Boosting

References

TELECOM Evolution

Ensemble methods for classification and regression

- Machine Learning not so "automatic": too many hyperparameters to tune

1. **Committee learning** or **wisdom of the crowd**: better results are obtained by combining the predictions of a set of **diverse** classifiers/regressors
2. **Meta-learning**: a procedure to automatically use a base classifier/regressor even weak to produce a performant classifier/regressor
3. **Ensemble learning**: is a kind of meta-learning, improves upon a single predictor by building an ensemble of predictors (with no hyperparameter)

Ensemble methods at a glance

- 1995: Boosting, Freund and Schapire
- 1996: Bagging, Breiman
- 1999: GradientBoosting, Friedman et al.
- 2001: Random forests, Breiman
- 2006: Extra-trees, Geurts, Ernst, Wehenkel

Outline

Motivation

Boosting
- AdaBoost as a Greedy Scheme
- Gradient Boosting

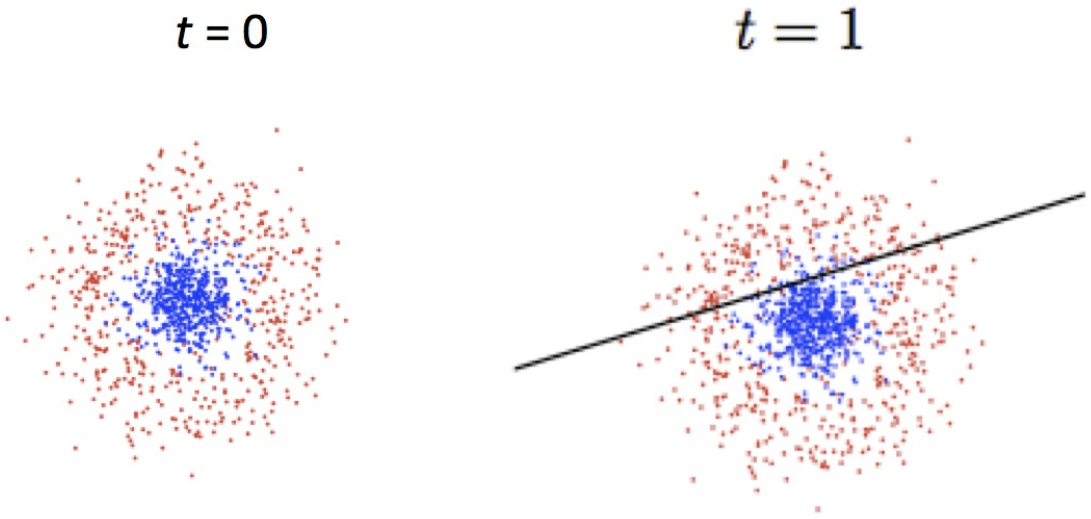
References

A preliminary question

- **Is it possible to "boost" a weak learner into a strong learner ?** Michael Kearns
- Yoav Freund and Rob Schapire proposed an iterative scheme, called, Adaboost to solve this problem
  - Idea: train a sequence of learners on weighted datasets with weights depending on the loss obtained so far.
  - Freund and Schapire received the Godel prize in 2003 for their work on AdaBoost.

Boosting a linear classifier

$H_1(x) = h_1(x)$
Binary Classifier: $F_1(x) = \text{sign}(H_1(x))$
Here: h_1: linear classifier
Training error= R_n

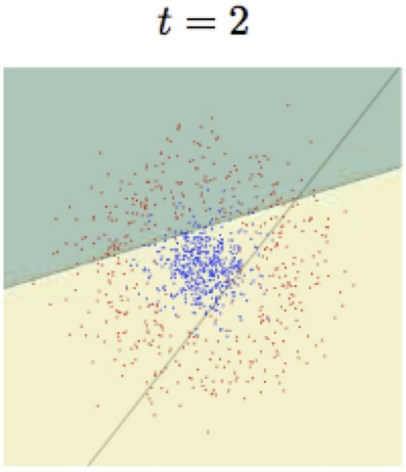


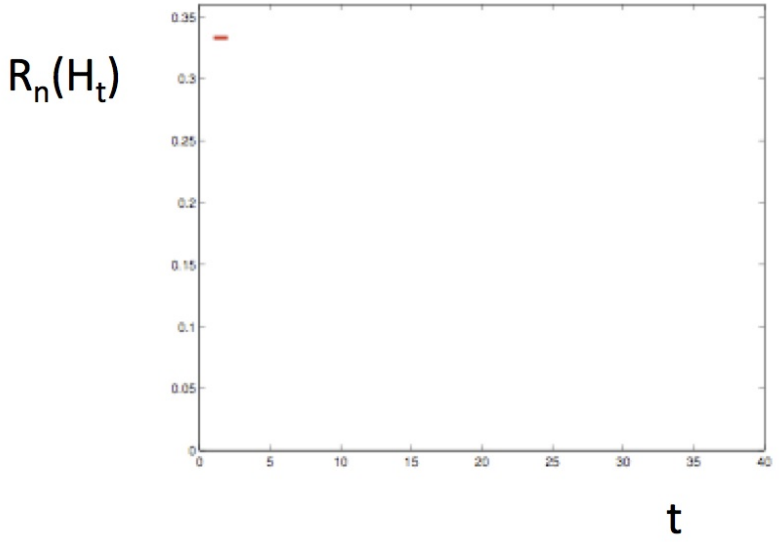
Source Jiri Matas (Oxford U.)

Boosting a linear classifier

$H_2(x) = \alpha_1 h_1(x) + \alpha_2 h_2(x)$

Binary Classifier: $F_2(x) = \text{sign}(H_2(x))$

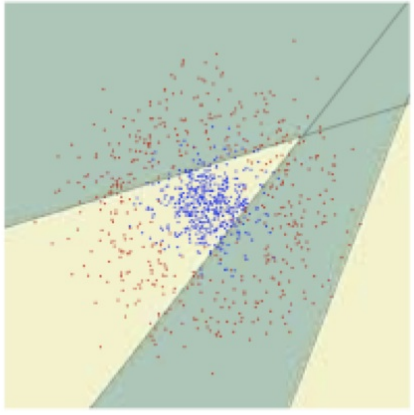




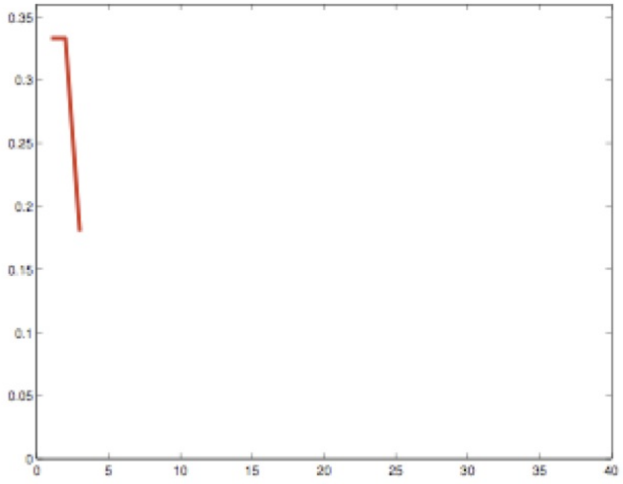
Source Jiri Matas (Oxford U.)

Boosting a linear classifier

$t = 3$



$R_n(H_t)$

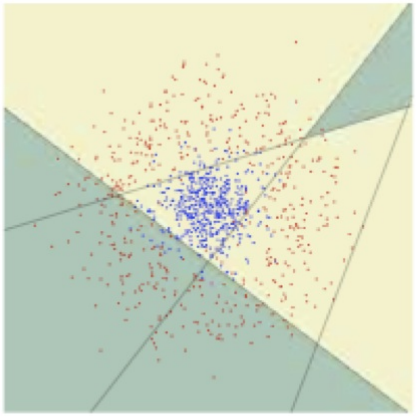


t

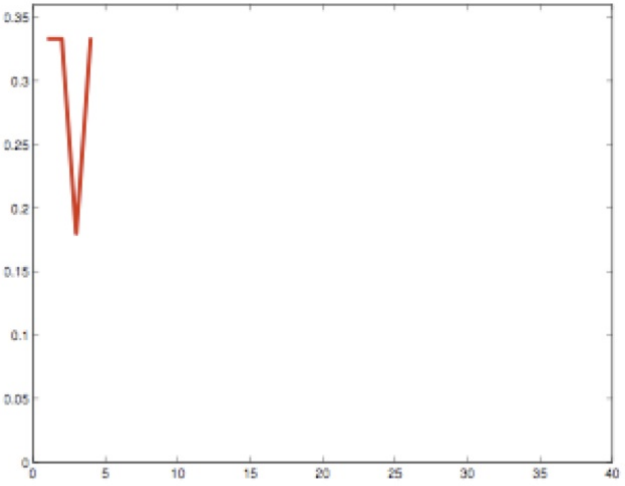
Source Jiri Matas (Oxford U.)

Boosting a linear classifier

$t = 4$

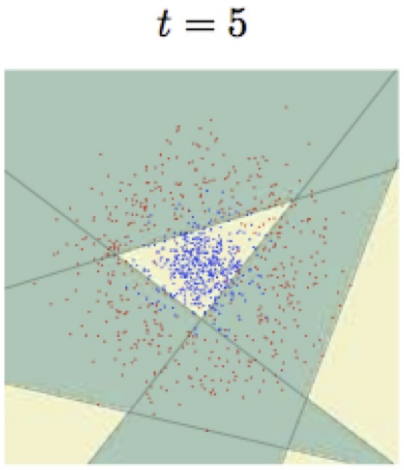


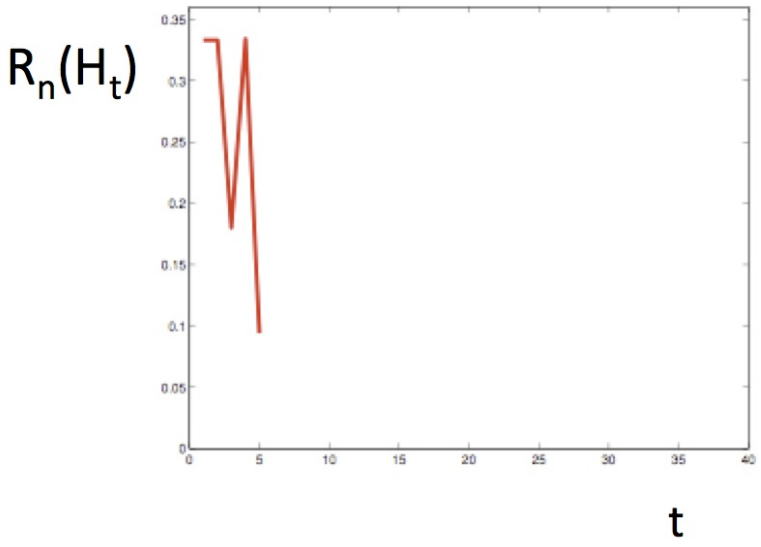
$R_n(H_t)$



Source Jiri Matas (Oxford U.)

Boosting a linear classifier

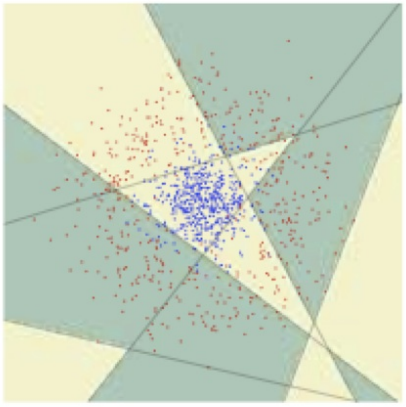




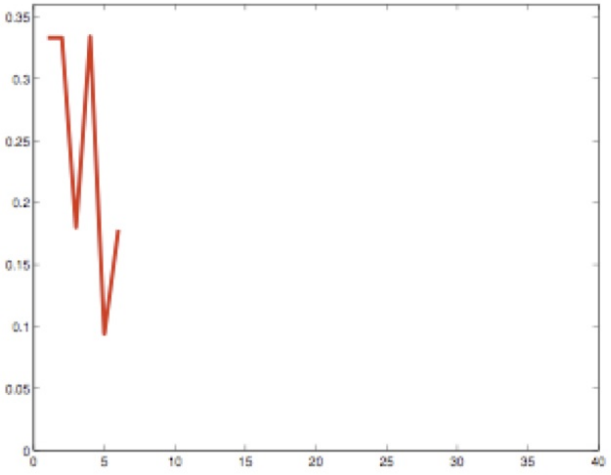
Source Jiri Matas (Oxford U.)

Boosting a linear classifier

$t = 6$

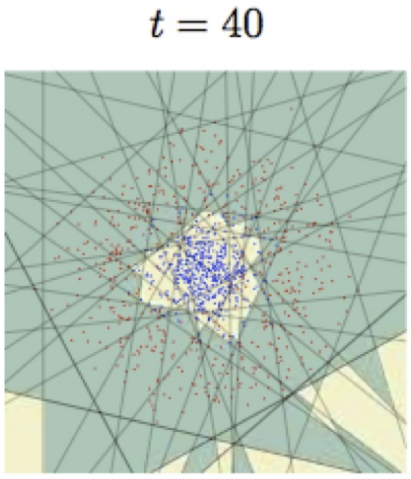


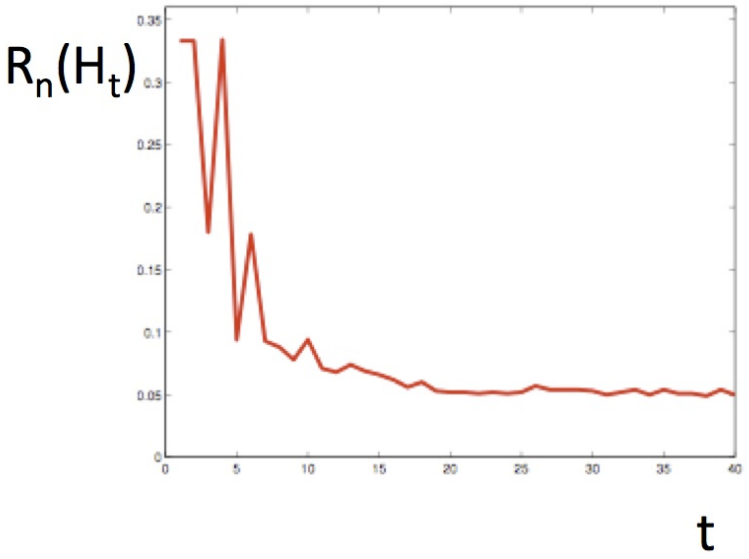
$R_n(H_t)$



Source Jiri Matas (Oxford U.)

Boosting a linear classifier





Source Jiri Matas (Oxford U.)

Weak classifier

Definition: weak classifier

A classifier whose average training error is no more than 0.5

NB : it means that we do not need to have a deep architecture as the base classifier (a "short" tree will fit for instance, a linear classifier will be perfect and so on...)

Adaboost idea

1. $\mathcal{H}$: a chosen class of "weak" binary classifiers, $\mathcal{A}$: a learning algorithm for $\mathcal{H}$

- Set $w_1(i) = 1/n$; $H_0 = 0$
- For $t = 1$ to T
  - $h_t = \arg\min_{h \in \mathcal{H}} \epsilon_t(h)$
  - with $\epsilon_t(h) = \mathbb{P}_{i \sim \mathbf{w}_t}[h(x_i) \neq y_i]$
  - Choose α_t
  - Choose w_{t+1}
  - $H_t = H_{t-1} + \alpha_t h_t$
- Output $F_T = \text{sign}(H_t)$

$\mathcal{H}$: a chosen class of "weak" binary classifiers

- Set $w_1(i) = 1/n$; $H_0 = 0$
- For $t = 1$ to T
  - $h_t = \arg\min_{h \in \mathcal{H}} \sum_{i=1}^{n} \epsilon_t(h)$
  - With $\epsilon_t(h) = \mathbb{P}_{i \sim \mathbf{w}_t}[h(x_i) \neq y_i]$
  - $\epsilon_t = \epsilon_t(h_t)$
  - $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
  - let $w_{t+1,i} = \frac{w_t(i) e^{-\alpha_t y_i h_t(x_i)}}{Z_{t+1}}$ where Z_{t+1} is a renormalization constant such that $\sum_{i=1}^{n} w_{t+1,i} = 1$
- $H_t = H_{t-1} + \alpha_t h_t$

Output $F_T = \text{sign}(H_t)$

What weight to choose ?

With the chosen definition, we have:

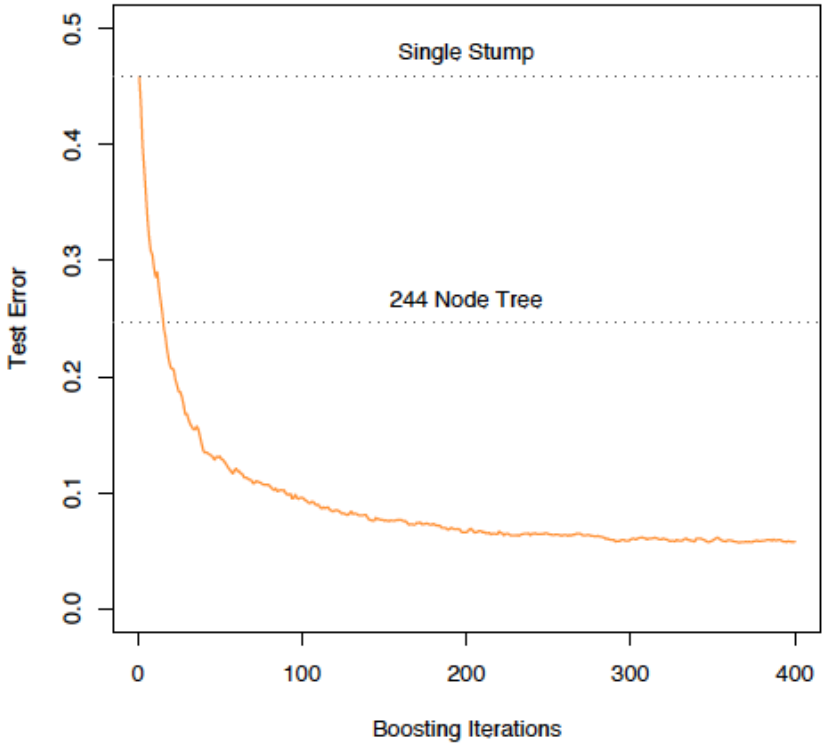
$$
\begin{aligned}
w_{t+1,i} &= \frac{w_{t,i} e^{-\alpha_t y_i h_t(x_i)}}{Z_t} \\
&= \frac{w_{t-1,i} e^{-\alpha_{t-1} y_i h_{t-1}(x_i)} e^{-\alpha_t y_i h_t(x_i)}}{Z_{t-1} Z_t} \\
&= \frac{e^{-y_i \sum_{s=1}^{t} \alpha_s h_s(x_i)}}{n \prod_{s=1}^{t} Z_s} \\
&= \frac{e^{-y_i H_t(x_i)}}{n \prod_{s=1}^{t} Z_s}
\end{aligned}
$$

You see the weights encourage to correct examples badly classified by the whole combination H_t

First of all let us study Z_t

$$
\begin{aligned}
Z_t &= \sum_{i=1}^{n} w_t(i) e^{-\alpha_t y_i h_t(x_i)} \\
&= \sum_{i=1}^{n} w_t(i) e^{-\alpha_t y_i h_t(x_i)} \\
&= \sum_{i:y_i h_t(x_i)=+1} w_t(i) e^{-\alpha_t} + \sum_{i:y_i h_t(x_i)=-1} w_t(i) e^{\alpha_t} \\
&= (1-\epsilon_t) e^{-\alpha_t} + \epsilon_t e^{\alpha_t} \\
&= (1-\epsilon_t) \sqrt{\frac{\epsilon_t}{1-\epsilon_t}} + \epsilon_t \sqrt{\frac{1-\epsilon_t}{\epsilon_t}} \\
&= \dots \\
&= 2\sqrt{\epsilon_t(1-\epsilon_t)}
\end{aligned}
$$

Typical behavior of boosting



Bound on the empirical error

Theorem

The empirical error of the classifier returned by Adaboost at time T verifies:

$$R_n(F_T) \leq e^{-2\sum_{t=1}^{T}(\frac{1}{2}-\epsilon_t)^2}.$$

Furthermore, if for all $t \in [1, T]$, $\gamma \leq (\frac{1}{2} - \epsilon_t)$, then

$$R_n(F_T) \leq e^{-2\gamma^2 T}.$$

Bound on the empirical error: proof

For all $u \in \mathbb{R}$, we have $1_{u \leq 0} \leq \exp(-u)$.
Then

$$
\begin{aligned}
, R_n(F_T) &= \frac{1}{n} \sum_{i=1}^{n} 1_{y_i F_T(x_i) \leq 0} \\
&\leq \frac{1}{n} \sum_{i=1}^{n} \exp(-y_i F_T(x_i)) = \frac{1}{n} \sum_{i=1}^{n} [n \prod_{t=1}^{T} Z_t] w_{t+1,i} = \prod_{t=1}^{T} Z_t
\end{aligned}
$$

Bound on the empirical error: proof ctd'

We can now express $\prod Z_t$ in terms of ϵ_t:

$$\begin{aligned}
\prod_{t=1}^{T} Z_t &= \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)} \\
&= \text{by remarkable identity} \\
&= \prod_{t=1}^{T} \sqrt{1-4(1/2-\epsilon_t)^2} \\
&\leq \prod_{t} e^{-2(1/2-\epsilon_t)^2} = e^{-2\sum_{t=1}^{T}(1/2-\epsilon_t)^2}
\end{aligned}$$

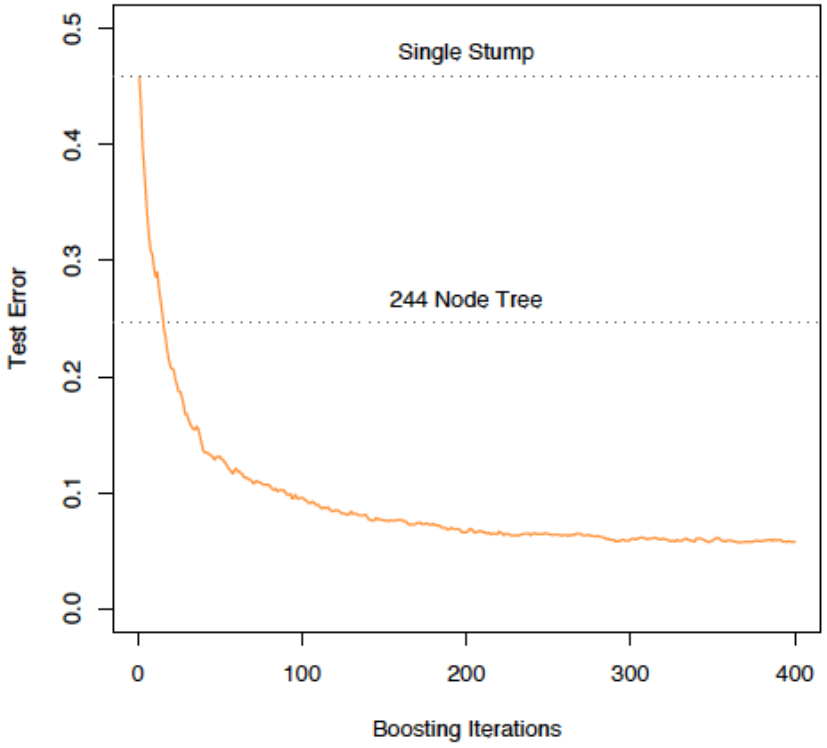
using the identity $1-u \leq \exp(-u)$.

Choice of α_t

The proof reveals serval interesting properties:

1. α_t is chosen to minimize $\prod_t Z_t = g(\alpha)$ with $g(\alpha) = (1-\epsilon_t)e^{-\alpha} + \epsilon_t e^{\alpha}$
   - $g'(\alpha) = -(1-\epsilon_t)e^{-\alpha} + \epsilon_t e^{\alpha}$
   - $g'(\alpha) = 0$ iff $(1-\epsilon_t)e^{-\alpha} = \epsilon_t e^{\alpha}$ iff $\alpha = 1/2 \log \frac{1-\epsilon_t}{\epsilon_t}$
2. The equality $(1-\epsilon_t)e^{-\alpha} = \epsilon_t e^{\alpha}$ means that Adaboost assigns at each time t the same distribution mass to correctly classified examples and incorrectly classified ones. However there is no contradiction because the number of incorrectly examples decreases.

Typical behavior of boosting



Adaboost with scikitlearn

http://scikit-learn.org/stable/modules/ensemble.htmladaboost

```
>>> from sklearn.crossvalidation import crossval/score
>>> from sklearn.datasets import loadiris
>>> from sklearn.ensemble import AdaBoostClassifier
>>> iris = loadiris()
>>> clf = AdaBoostClassifier(nestimators=100)
>>> scores = crossvalscore(clf, iris.data, iris.target)
>>> scores.mean() 0.9...
```

Boosting and regularization

- You have to wait a long time to see Boosting overfit. However contrary to first assertions, Adaboost does overfit
- Early stopping:answer

  Fitting the residuals.

Boosting as a coordinate descent

At the same time, different groups proved that Adaboost writes as a coordinate descent in the convex hull of $\mathcal{H}$.

- Greedy function approximation, Friedman, 1999.
- MarginBoost and AnyBoost : Mason et al. 1999.

Gradient Boosting

- At each boosting step, one need to solve

$$(h_t, \alpha_t) = \arg\min_{h,\alpha} \sum_{i=1}^{n} \ell(y_i, H_{t-1}(x_i) + \alpha h) = L(y, H_{t-1} + \alpha h)$$

- Gradient approximation
  $L(y, H_{t-1} + \alpha h) \sim L(y, H_{t-1}) + \alpha \langle \nabla L(H_{t-1}), h \rangle$.
- Gradient boosting: replace the minimization step by a *gradient descent* type step:
  - Choose h_t as the best possible descent direction in $\mathcal{H}$
  - Choose α_t that minimizes $L(y, H + \alpha h_t)$
- Easy if finding the best descent direction is easy!

Gradient boosting and adaboost

Those two algorithms are equivalent!

- Denoting $H_t = \sum_{t'=1}^{t} \alpha_{t'} h_{t'}$,

$$\sum_{i=1}^{n} e^{-y_i(H_{t-1}(x_i)+\alpha h(x_i))} = \sum_{i=1}^{n} e^{-y_i H_{t-1}(x_i)} e^{-\alpha y_i h(x_i)}$$

$$= \sum_{i=1}^{n} w_i'(t) e^{-\alpha y_i h(x_i)}$$

$$= (e^{\alpha} - e^{-\alpha}) \sum_{i=1}^{n} w_i'(t) \ell^{0/1}(y_i, h(x_i)) + e^{-\alpha} \sum_{i=1}^{n} w_i'(t)$$

Gradient boosting and adaboost (ctd)

Those two algorithms are equivalent!

- The minimizer h_t in h is independent of α and is also the minimizer of

$$\sum_{i=1}^{n} w_i'(t)\ell^{0/1}(y_i, h(x_i))$$

Gradient boosting and Adaboost

- The optimal α_t is then given by

$$\alpha_t = \frac{1}{2} \log \frac{1 - \epsilon'_t}{\epsilon'_t}$$

with $\epsilon'_t = (\sum_{i=1}^n w'_i(t) \ell^{0/1}(y_i, h_t(x_i)))/(\sum_{i=1}^n w'_i(t))$

- One verify then by recursion that

$$w_i(t) = w'_i(t)/(\sum_{i=1}^n w'_i(t))$$

and thus the two procedures are equivalent!

AnyBoost or Foward Stagewise Additive model

- General greedy optimization strategy to obtain a linear combination of *weak* predictor
  - Set $t = 0$ and $H_0 = 0$.
  - For $t = 1$ to T,
    - $(h_t, \alpha_t) = \arg\min_{h,\alpha} \sum_{i=1}^{n} \ell(y_i, H_{t-1}(x_i) + \alpha h(x_i))$
    - $H_t = H_{t-1} + \alpha_t h_t$
  - Output $H_T = \sum_{t=1}^{T} \alpha_t h_t$

Losses in Forward Stagewise Additive Modeling

- AdaBoost with $\ell(y, h) = e^{-yh}$
- LogitBoost with $\ell(y, h) = \log(1 + e^{-yh})$
- L_2Boost with $\ell(y, h) = (y - h)^2$ (Matching pursuit)
- L_1Boost with $\ell(y, h) = |y - h|$
- HuberBoost with
  $\ell(y, h) = |y - h|^2 \mathbf{1}_{|y-h|<\epsilon} + (2\epsilon|y - h| - \epsilon^2)\mathbf{1}_{|y-h|\geq\epsilon}$

  Simple principle but no easy numerical scheme except for AdaBoost and L_2Boost...

Continuous loss functions and gradient boosting

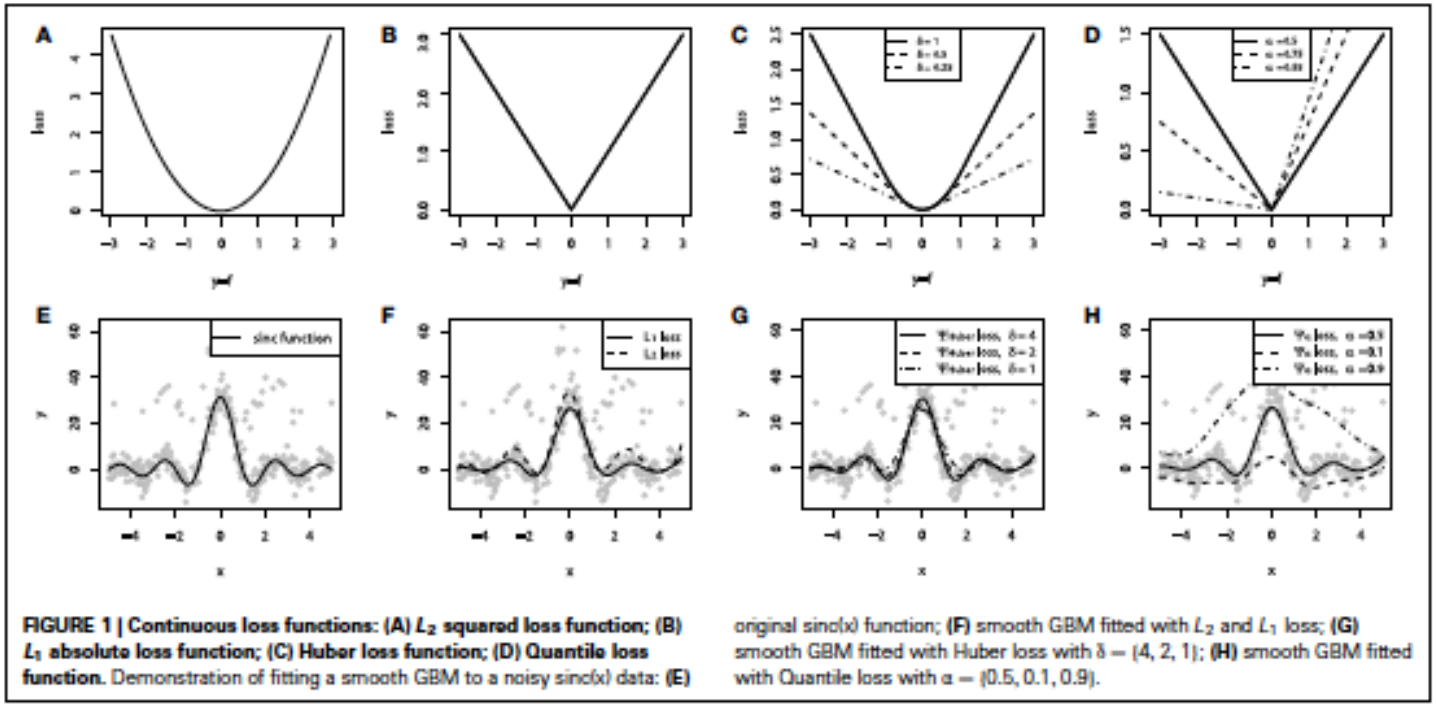


FIGURE 1 | Continuous loss functions: (A) L_2 squared loss function; (B) L_1 absolute loss function; (C) Huber loss function; (D) Quantile loss function. Demonstration of fitting a smooth GBM to a noisy sinc(x) data: **(E)** original sinc(x) function; **(F)** smooth GBM fitted with L_2 and L_1 loss; **(G)** smooth GBM fitted with Huber loss with $\delta = (4, 2, 1)$; **(H)** smooth GBM fitted with Quantile loss with $\alpha = (0.5, 0.1, 0.9)$.

L_2 Boosting

- Loss function for regression: $\ell(y, h) = (y - h)^2$
- $(h_t, \alpha_t) = \arg\min_{h,\alpha} \sum_{i=1}^{n} (y_i - H_t(x_i) + \alpha h)^2$

  Fitting the residuals.

Outline

Motivation

Boosting

References

References

- Freund and Schapire, 1996
- Greedy function approximation, Friedman, 1999.
- MarginBoost and AnyBoost : Mason et al. 1999.
- Tutorial Boosting de Buehlman
- Boosting, Schapire and Freund, 2012