AdaBoost

Jiri Matas and Jan Šochman

Centre for Machine Perception
Czech Technical University, Prague
http://cmp.felk.cvut.cz

Outline:

- AdaBoost algorithm
  - Why is of interest?
  - How it works?
  - Why it works?
- AdaBoost variants
- AdaBoost with a Totally Corrective Step (TCS)
- Experiments with a Totally Corrective Step

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

- 1990 – Boost-by-majority algorithm (Freund)
- 1995 – AdaBoost (Freund & Schapire)
- 1997 – Generalized version of AdaBoost (Schapire & Singer)
- 2001 – AdaBoost in Face Detection (Viola & Jones)

Interesting properties:

- AB is a linear classifier with all its desirable properties.
- AB output converges to the logarithm of likelihood ratio.
- AB has good generalization properties.
- AB is a feature selector with a principled strategy (minimisation of upper bound on empirical error).
- AB close to sequential decision making (it produces a sequence of gradually more complex classifiers).

- AdaBoost is an algorithm for constructing a ”strong” classifier as linear combination

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

of “simple” “weak” classifiers $h_t(x)$.

- AdaBoost is an algorithm for constructing a ”strong” classifier as linear combination

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

of “simple” “weak” classifiers $h_t(x)$.

Terminology

- $h_t(x)$ ... “weak” or basis classifier, hypothesis, ”feature”
- $H(x) = \mathit{sign}(f(x))$ ... ‘’strong” or final classifier/hypothesis

- AdaBoost is an algorithm for constructing a ”strong” classifier as linear combination
$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$
of “simple” “weak” classifiers $h_t(x)$.

Terminology

- $h_t(x)$ ... “weak” or basis classifier, hypothesis, ”feature”
- $H(x) = \mathit{sign}(f(x))$ ... ‘’strong” or final classifier/hypothesis

Comments

- The $h_t(x)$’s can be thought of as features.
- Often (typically) the set $\mathcal{H} = \{h(x)\}$ is infinite.

(Discrete) AdaBoost Algorithm – Singer & Schapire (1997)

Given: $(x_1, y_1), ..., (x_m, y_m)$; $x_i \in \mathcal{X}, y_i \in \{-1, 1\}$

Initialize weights $D_1(i) = 1/m$

For $t = 1, ..., T$:

1. (Call *WeakLearn*), which returns the weak classifier $h_t : \mathcal{X} \rightarrow \{-1, 1\}$ with minimum error w.r.t. distribution D_t;
2. Choose $\alpha_t \in R$,
3. Update

$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where Z_t is a normalization factor chosen so that D_{t+1} is a distribution

Output the strong classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

WeakLearn

Loop step: Call *WeakLearn*, given distribution D_t;
returns weak classifier $h_t : \mathcal{X} \to \{-1, 1\}$ from $\mathcal{H} = \{h(x)\}$

- Select a weak classifier with the smallest weighted error
  $h_t = \arg \min\limits_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- Prerequisite: $\epsilon_t < 1/2$ (otherwise stop)
- *WeakLearn* examples:
  - Decision tree builder, perceptron learning rule – $\mathcal{H}$ *infinite*
  - Selecting the best one from given *finite* set $\mathcal{H}$

Loop step: Call *WeakLearn*, given distribution D_t;
returns weak classifier $h_t : \mathcal{X} \rightarrow \{-1, 1\}$ from $\mathcal{H} = \{h(x)\}$

- Select a weak classifier with the smallest weighted error
  $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- Prerequisite: $\epsilon_t < 1/2$ (otherwise stop)
- *WeakLearn* examples:
  - Decision tree builder, perceptron learning rule – $\mathcal{H}$ *infinite*
  - Selecting the best one from given *finite* set $\mathcal{H}$

Demonstration example

Weak classifier = perceptron

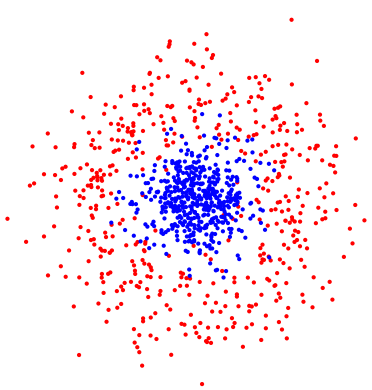
$\bullet \sim N(0,1)$ $\quad$ $\bullet \sim \frac{1}{r\sqrt{8\pi^3}} e^{-1/2(r-4)^2}$

Loop step: Call *WeakLearn*, given distribution D_t;
returns weak classifier $h_t : \mathcal{X} \to \{-1, 1\}$ from $\mathcal{H} = \{h(x)\}$

- Select a weak classifier with the smallest weighted error
  $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- Prerequisite: $\epsilon_t < 1/2$ (otherwise stop)
- *WeakLearn* examples:
  - Decision tree builder, perceptron learning rule – $\mathcal{H}$ *infinite*
  - Selecting the best one from given *finite* set $\mathcal{H}$

Demonstration example

Training set



Weak classifier = perceptron

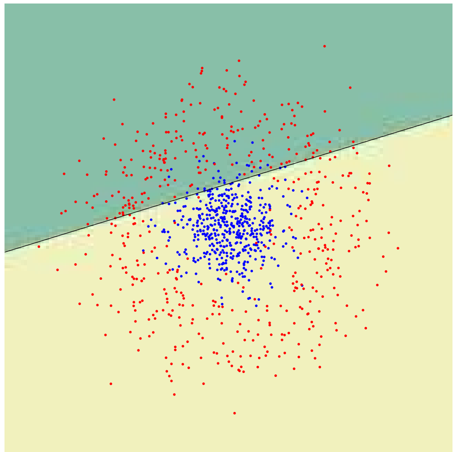
● $\sim N(0,1)$ ● $\sim \frac{1}{r\sqrt{8\pi^3}} e^{-1/2(r-4)^2}$

Loop step: Call *WeakLearn*, given distribution D_t;
returns weak classifier $h_t : \mathcal{X} \to \{-1, 1\}$ from $\mathcal{H} = \{h(x)\}$

- Select a weak classifier with the smallest weighted error
  $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- Prerequisite: $\epsilon_t < 1/2$ (otherwise stop)
- *WeakLearn* examples:
  - Decision tree builder, perceptron learning rule – $\mathcal{H}$ *infinite*
  - Selecting the best one from given *finite* set $\mathcal{H}$

Demonstration example

Training set



Weak classifier = perceptron

$\bullet \sim N(0,1)$ $\quad\quad$ $\bullet \sim \frac{1}{r\sqrt{8\pi^3}} e^{-1/2(r-4)^2}$

- The main objective is to minimize $\varepsilon_{tr} = \frac{1}{m}|\{i : H(x_i) \neq y_i\}|$
- It can be upper bounded by $\varepsilon_{tr}(H) \leq \prod_{t=1}^{T} Z_t$

Effect on the training set

Reweighting formula:

$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t} = \frac{exp(-y_i \sum_{q=1}^{t} \alpha_q h_q(x_i))}{m \prod_{q=1}^{t} Z_q}$$

$$exp(-\alpha_t y_i h_t(x_i)) \begin{cases} < 1, & y_i = h_t(x_i) \\ > 1, & y_i \neq h_t(x_i) \end{cases}$$

}

$\Rightarrow$ Increase (decrease) weight of wrongly (correctly) classified examples. The weight is the upper bound on the error of a given example!

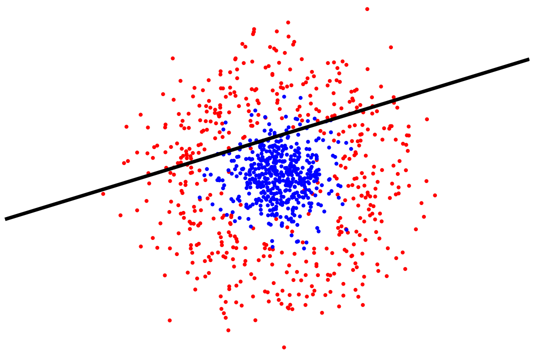
Effect on the training set

Reweighting formula:

$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t} = \frac{exp(-y_i \sum_{q=1}^{t} \alpha_q h_q(x_i))}{m \prod_{q=1}^{t} Z_q}$$

$$exp(-\alpha_t y_i h_t(x_i)) \begin{cases} < 1, & y_i = h_t(x_i) \\ > 1, & y_i \neq h_t(x_i) \end{cases}$$

}



$\Rightarrow$ Increase (decrease) weight of wrongly (correctly) classified examples. The weight is the upper bound on the error of a given example!

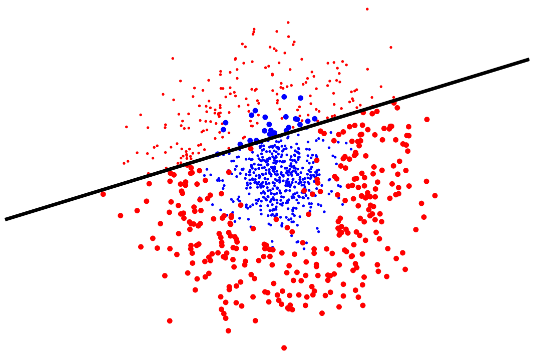
Effect on the training set

Reweighting formula:

$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t} = \frac{exp(-y_i \sum_{q=1}^{t} \alpha_q h_q(x_i))}{m \prod_{q=1}^{t} Z_q}$$

$$exp(-\alpha_t y_i h_t(x_i)) \begin{cases} < 1, & y_i = h_t(x_i) \\ > 1, & y_i \neq h_t(x_i) \end{cases}$$

$\}$



$\Rightarrow$ Increase (decrease) weight of wrongly (correctly) classified examples. The weight is the upper bound on the error of a given example!

Effect on the training set

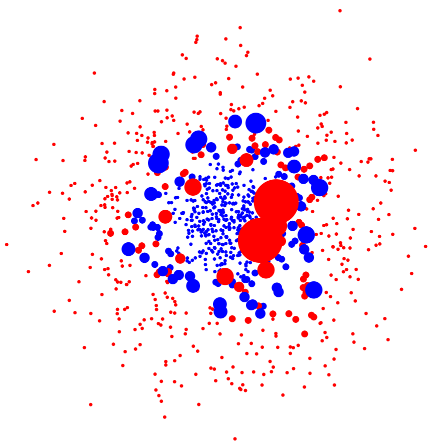
Reweighting formula:

$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t} = \frac{exp(-y_i \sum_{q=1}^{t} \alpha_q h_q(x_i))}{m \prod_{q=1}^{t} Z_q}$$

$$exp(-\alpha_t y_i h_t(x_i)) \left\{ \begin{array}{ll} < 1, & y_i = h_t(x_i) \\ > 1, & y_i \neq h_t(x_i) \end{array} \right.$$

$\}$

$\Rightarrow$ Increase (decrease) weight of wrongly (correctly) classified examples. The weight is the upper bound on the error of a given example!



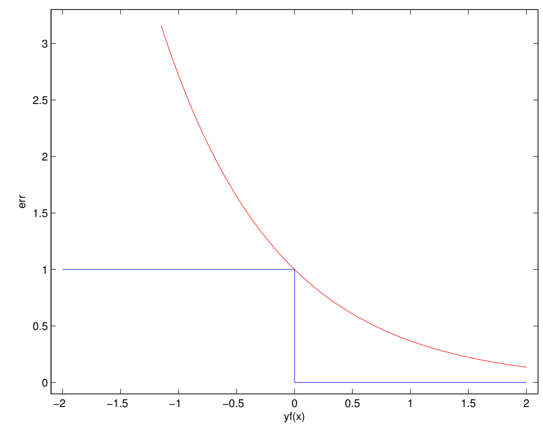
Effect on the training set

Reweighting formula:

$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t} = \frac{exp(-y_i \sum_{q=1}^{t} \alpha_q h_q(x_i))}{m \prod_{q=1}^{t} Z_q}$$

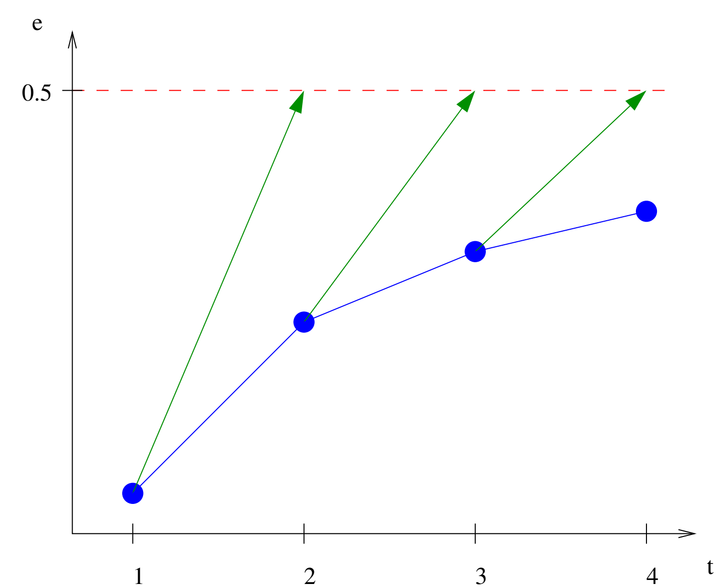
$$exp(-\alpha_t y_i h_t(x_i)) \begin{cases} < 1, & y_i = h_t(x_i) \\ > 1, & y_i \neq h_t(x_i) \end{cases}$$

}



$\Rightarrow$ Increase (decrease) weight of wrongly (correctly) classified examples. The weight is the upper bound on the error of a given example!

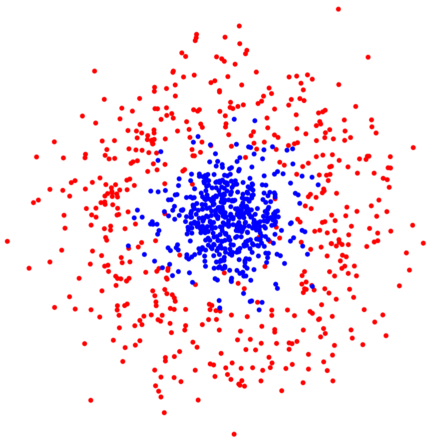
Effect on h_t



- α_t minimize $Z_t \Rightarrow$
  $$\sum_{i:h_t(x_i)=y_i} D_{t+1}(i) = \sum_{i:h_t(x_i)\neq y_i} D_{t+1}(i)$$
- Error of h_t on D_{t+1} is $1/2$
- Next weak classifier is the most "independent" one

Summary of the Algorithm

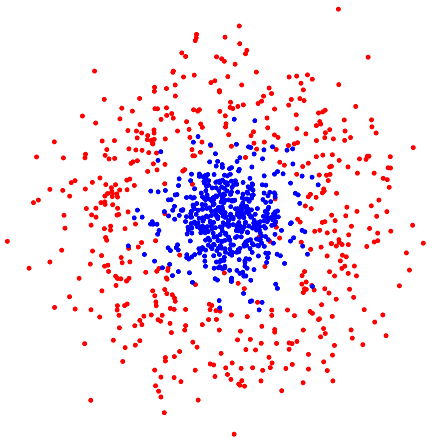
Initialization...



1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

Initialization...

For $t = 1, ..., T$:

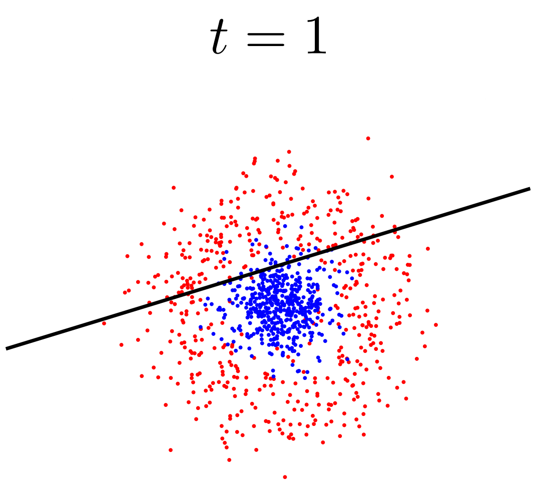


Summary of the Algorithm

Initialization...

For $t = 1, ..., T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$

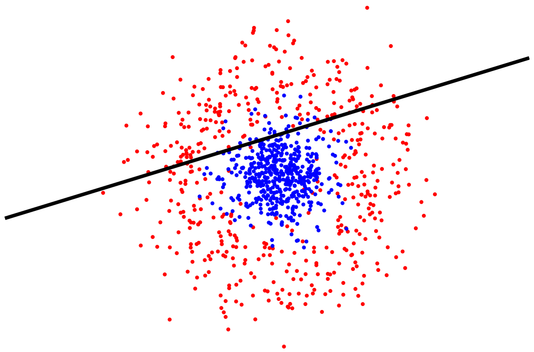


Initialization...

For $t = 1, ..., T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop

$t = 1$

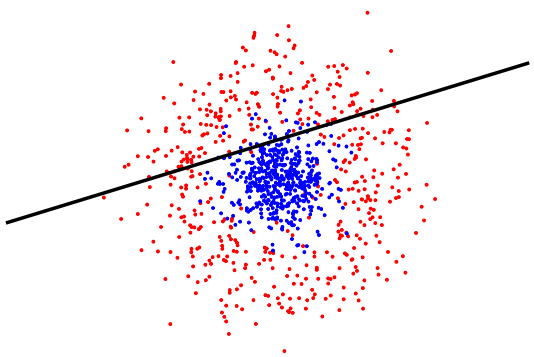


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Initialization...
For $t = 1, ..., T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1+r_t}{1-r_t})$

$t = 1$

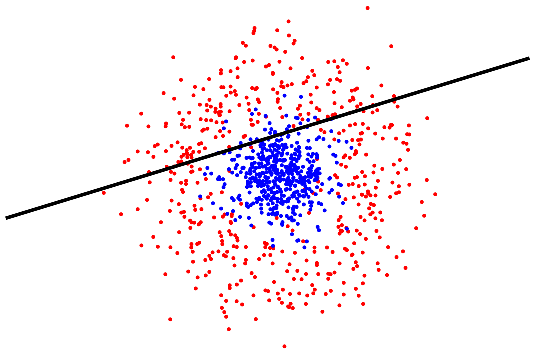


Initialization...
For $t = 1, ..., T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1+r_t}{1-r_t})$
- Update

$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

$t = 1$



Summary of the Algorithm

Initialization...

For $t = 1, ..., T$:

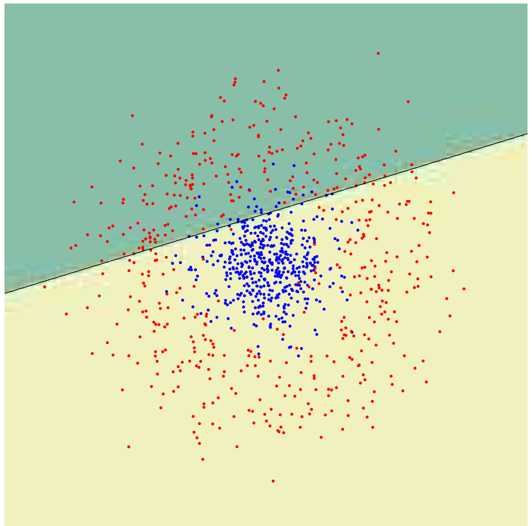
- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1+r_t}{1-r_t})$
- Update

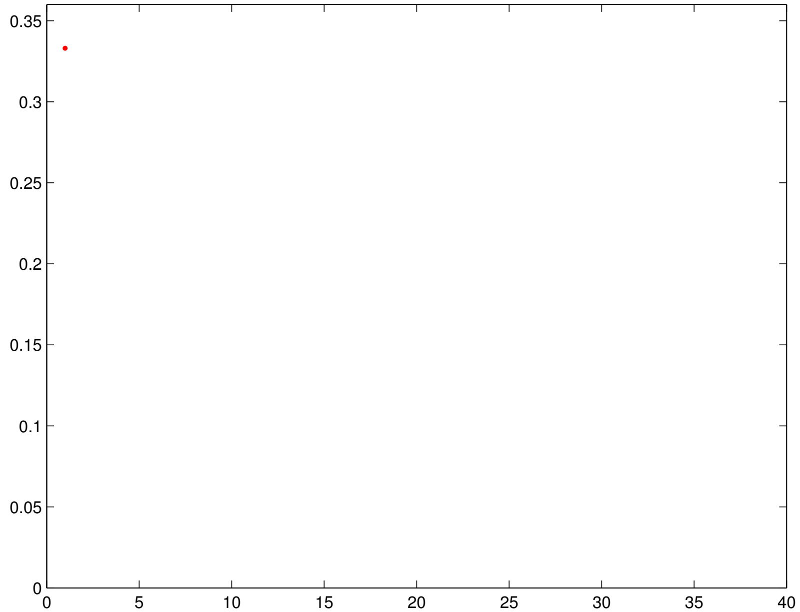
$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 1$





Summary of the Algorithm

Initialization...
For $t = 1, ..., T$:

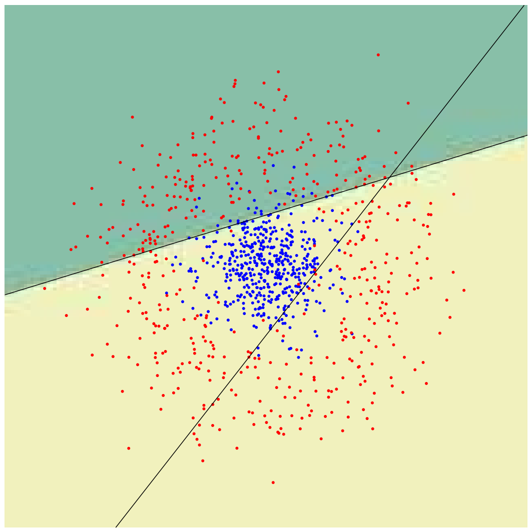
- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1+r_t}{1-r_t})$
- Update

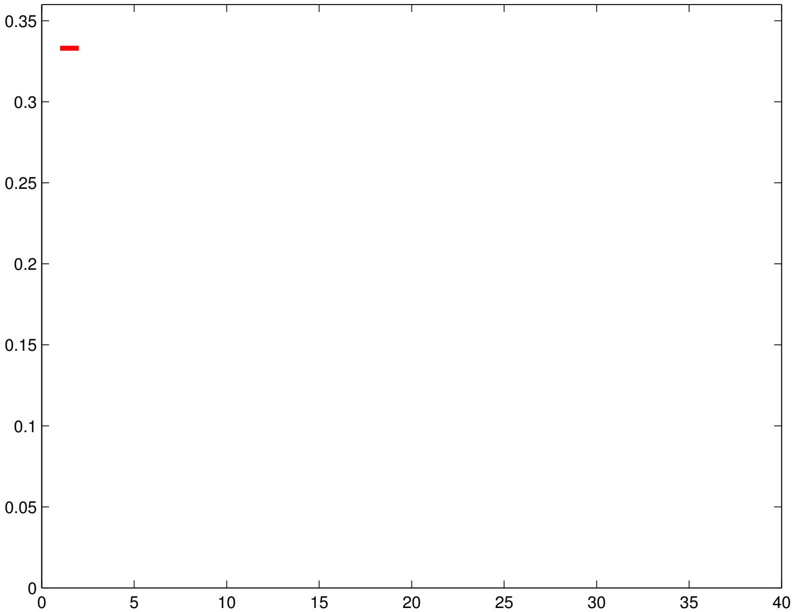
$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 2$





Summary of the Algorithm

Initialization...
For $t = 1, ..., T$:

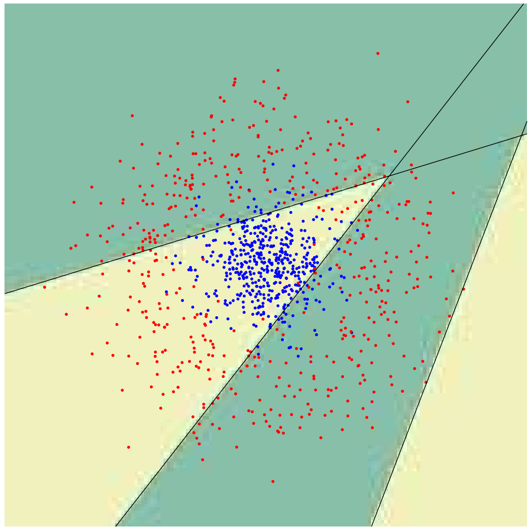
- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1+r_t}{1-r_t})$
- Update

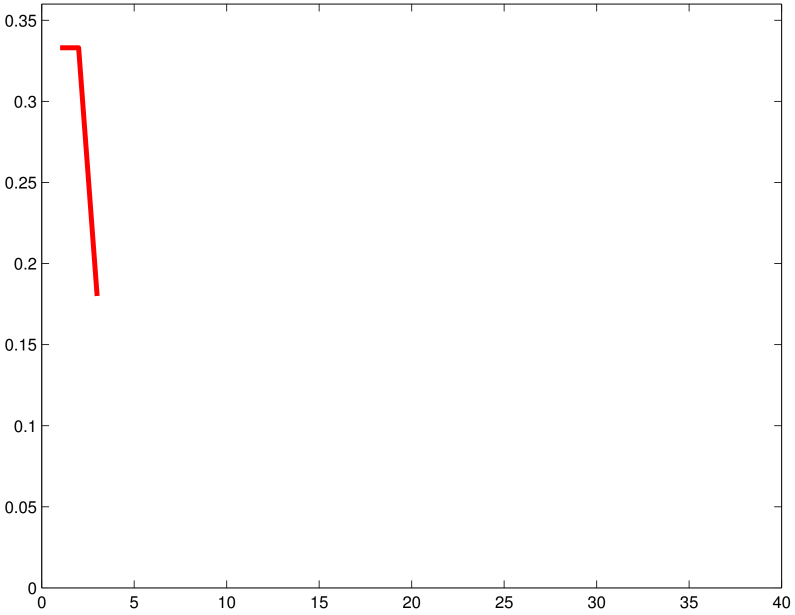
$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 3$





Summary of the Algorithm

Initialization...
For $t = 1, ..., T$:

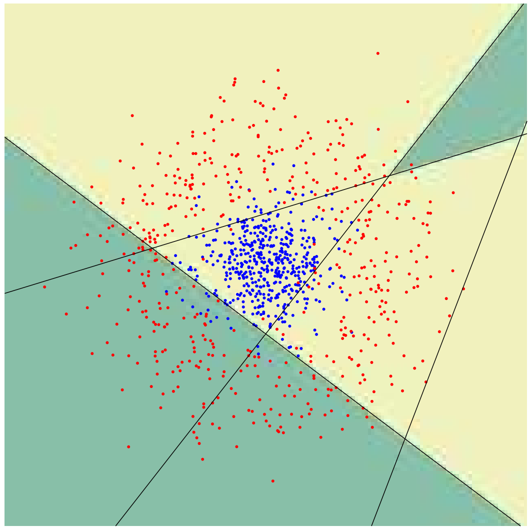
- Find $h_t = \arg \min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2} \log(\frac{1+r_t}{1-r_t})$
- Update

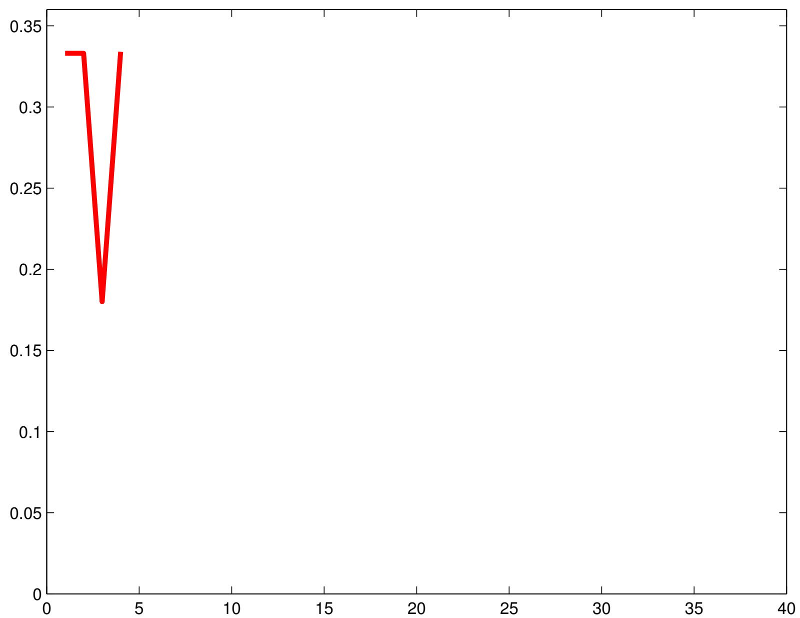
$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 4$





Summary of the Algorithm

Initialization...
For $t = 1, ..., T$:

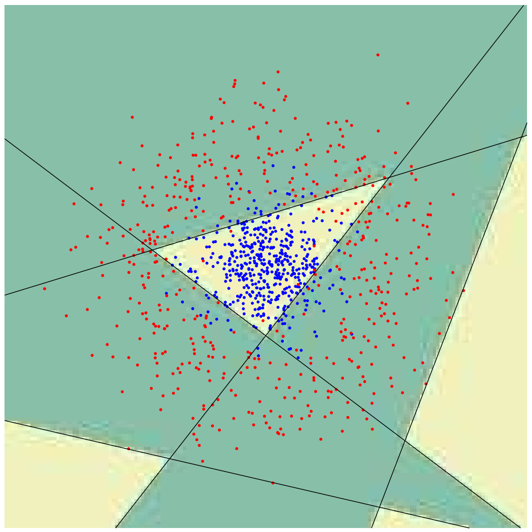
- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1+r_t}{1-r_t})$
- Update

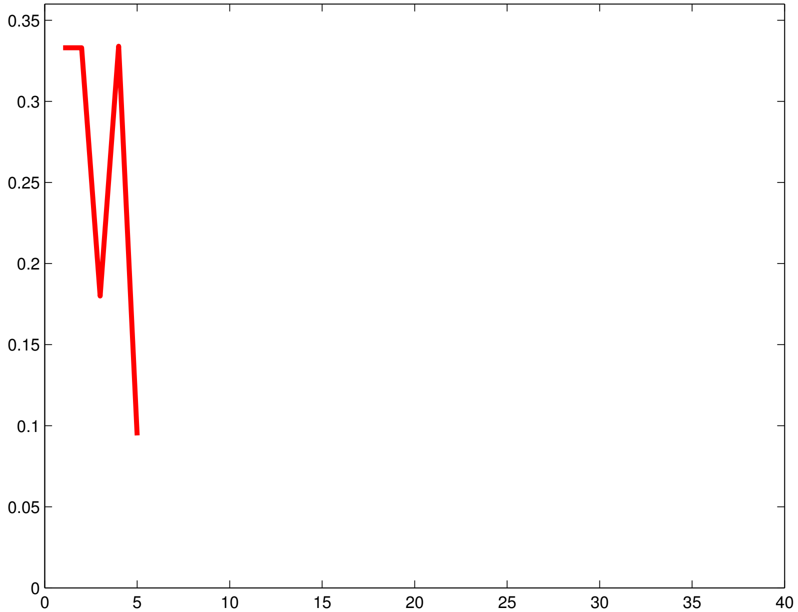
$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 5$





Summary of the Algorithm

Initialization...

For $t = 1, ..., T$:

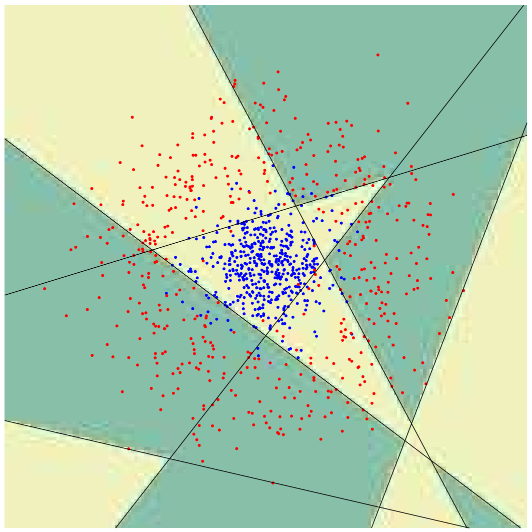
- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1+r_t}{1-r_t})$
- Update

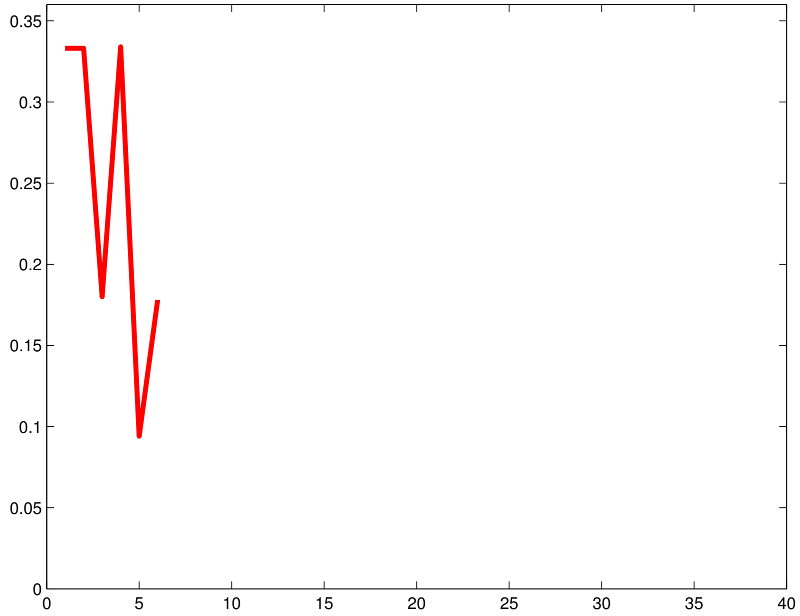
$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 6$





Summary of the Algorithm

Initialization...
For $t = 1, ..., T$:

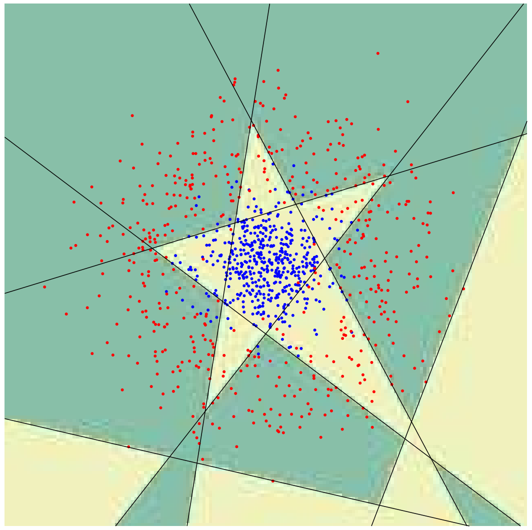
- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1+r_t}{1-r_t})$
- Update

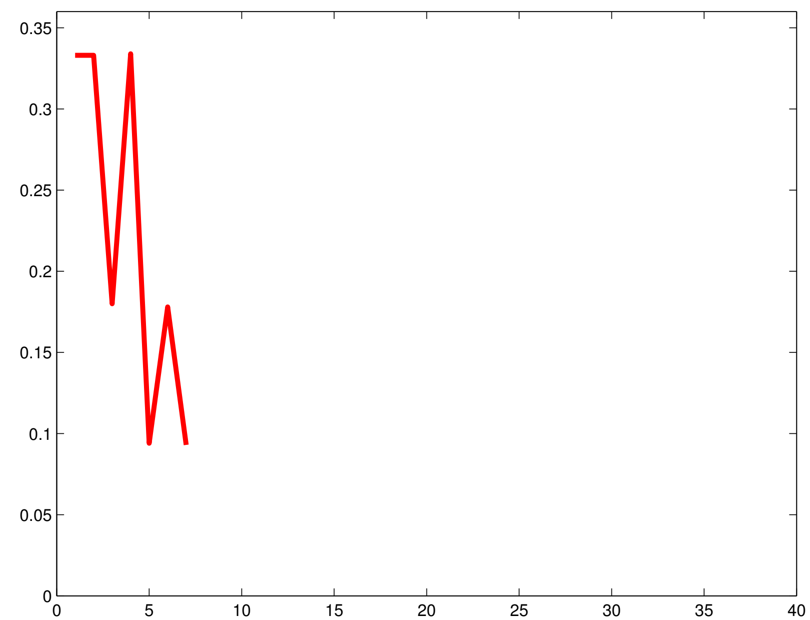
$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 7$





Summary of the Algorithm

Initialization...
For $t = 1, ..., T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1+r_t}{1-r_t})$
- Update

$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 40$