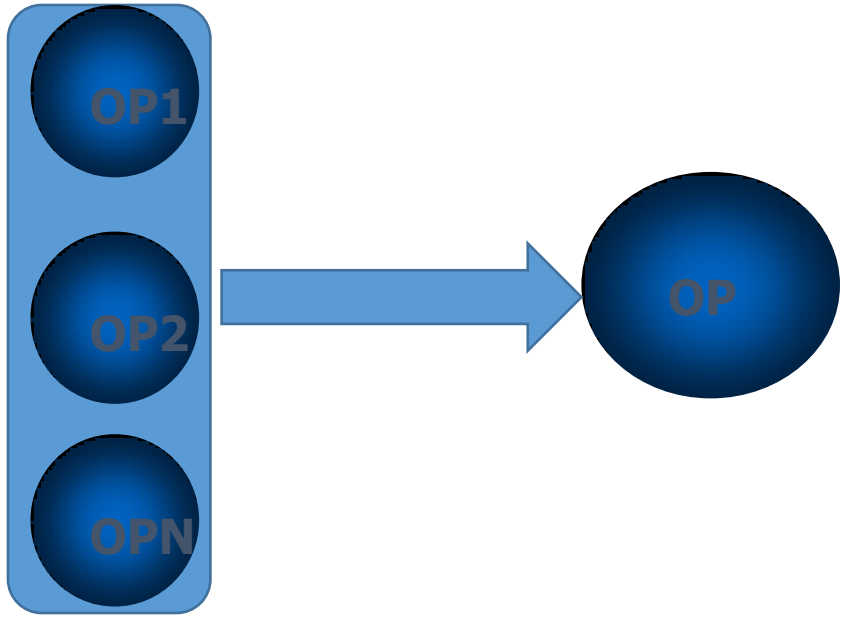
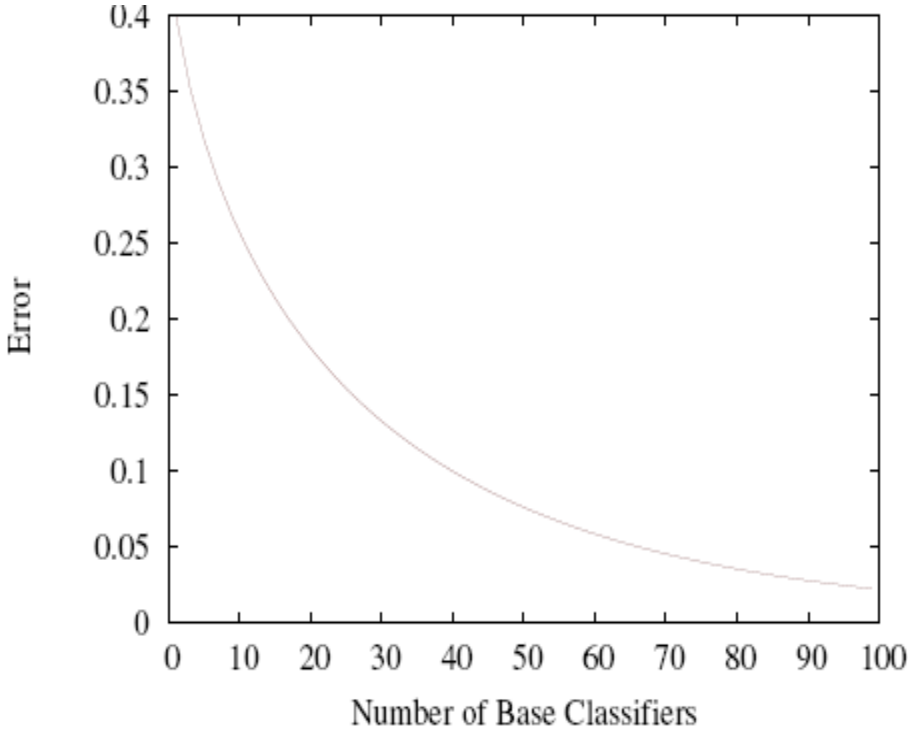
Boosting và Adaboost

Ý tưởng

Kết hợp nhiều quan điểm cá nhân để đưa ra quan điểm cuối cùng [Polikar (2006)].





Ứng dụng thành công trong các lĩnh vực : tài chánh, sinh tin học, hóa tin học, y khoa, sản xuất, địa lý, và truy vấn ảnh.

Ứng dụng trong thực tế thường gặp

- Thảo luận làm việc theo nhóm
- Bầu cử theo đa số (Majority Voting)
- Dự đoán

Định lý *"Condorcet's Jury"*

Hầu tước De Condorcet (1743-1794)

Giả thuyết rằng :

- Những voter chỉ có 2 khả năng bỏ phiếu đúng hoặc sai
- Gọi p là xác suất bình chọn đúng của mỗi voter
- M là xác suất tổng hợp từ các voter:
  - p>0.5 ngầm hiểu rằng M>p
  - và M sẽ tiệm cận 1 nếu tất cả p>0.5 và số lượng voter là lớn
- Có hai giới hạn :
  - Giả sử về sự độc lập của các voter (trong thực tế thì khó đạt được)
  - Định lý chỉ áp dụng trên việc bình chọn trên 2 khả năng

Trí tuệ tập thể

(The wisdom of crowds)

- Sir Francis Galton (1822-1911)
  - Dự đoán trọng lượng 1 con bò.
  - Quan sát của tác giả thấy rằng : tất cả dự đoán đều không chính xác nhưng khi lấy trung bình của tất cả các dự đoán thì gần đúng với kết quả thực sự.
- Tuy nhiên không phải đám đông là chân lý
- James Michael Surowiecki, an American financial journalist, xuất bản sách "The Wisdom of Crowds" đề xuất:
  - **Tính đa dạng của các quan điểm** : mỗi thành viên có thông tin riêng của mình thậm chí có các giải thích lập dị
  - **Tính độc lập** : không bị ảnh hưởng bởi các tác động xung quanh
  - **Tính phân cấp** : thành viên có thể rút ra quan điểm dựa trên tri thức cục bộ
  - **Tính tập hợp** : tồn tại một vài cơ chế cho việc điều chỉnh các phán đoán riêng sang một quyết định tập hợp.

Áp dụng học tổ hợp trong máy học

- A strong learner (tạm dịch đối tượng học mạnh) : trong việc phân lớp đó là những phân lớp có độ chính xác cao (không phải là 100%).
- A weak learner (tạm dịch đối tượng học yếu) trong việc phân lớp đó là những phân lớp có độ chính xác không cao (nhưng không có nghĩa weak là quá thấp) phải hơn việc phân lớp ngẫu nhiên *một chút.*
- *Tại sao phải học tổ hợp?*
  - Vì khó tạo trực tiếp được một strong learner
  - Nhưng dễ tạo ra được weak learner
  - Vậy có thể tạo ra được A strong learner từ các weak learner? Câu trả lời là OK!

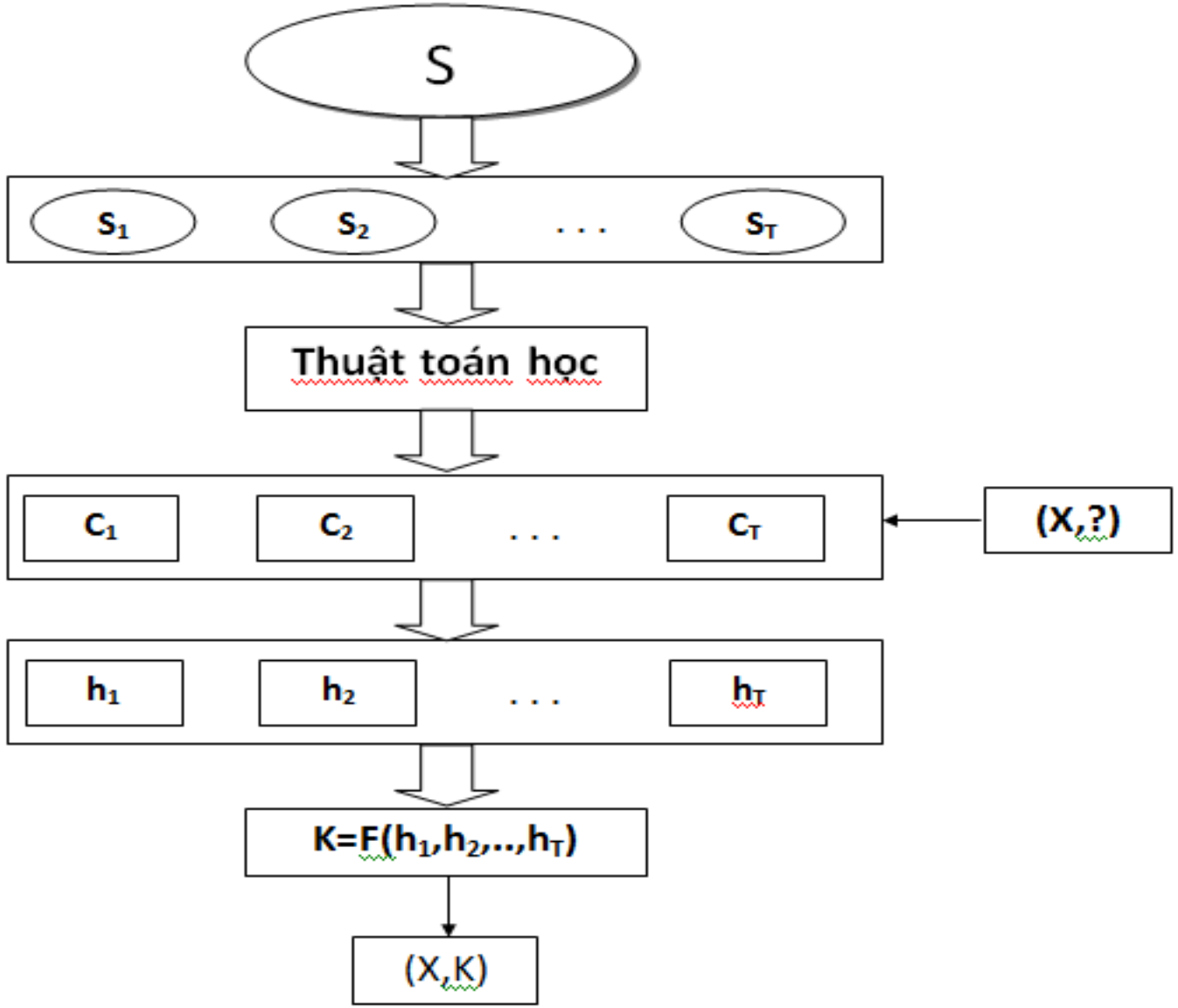
Mục tiêu của học tổ hợp

- Cải thiện độ chính xác phân lớp
- Xét phương thức dựa trên Majority Voting
  - T phân lớp yếu độc lập với nhau
  - Mục tiêu phân chia thành 2 lớp
  - Mỗi phân lớp yếu có xác suất p
  - Thì phân lớp tổng hợp Pens

$$\text{Pens} = \sum_{k=\frac{T}{2}+1}^{T} \binom{T}{k} p^k (1-p)^{T-k}$$

- Nhận xét :
  - $P_{ens} \rightarrow 1$ khi $T \rightarrow \infty$ và $p>0.5$
  - $P_{ens} \rightarrow 0$ khi $T \rightarrow \infty$ và $p<0.5$
  - Vậy để phân lớp tốt thì $p>0.5$ và chúng ta có nhiều phân lớp yếu độc lập

Mô hình tổng quát



Các bước tiến hành trong việc học tổ hợp

- Bước 1 xây dựng các phân lớp yếu từ tập huấn luyện
- Bước 2 với mỗi phân lớp yếu gán nhãn dự đoán cho nó
- Bước 3 kết hợp các phân lớp yếu để được phân lớp mạnh

2 vấn đề chính của việc học tổ hợp

- Tạo ra các các weak learner như thế nào?

- Kết hợp các weak learner để tạo thành một strong learner?

Phân tích Bias và Variance

- Phân tích lỗi dựa vào 3 đặc điểm sau: Intrisic(bản chất), Variance, Bias.
- Intrisic chủ yếu là do nhiễu tập huấn luyện gây ra
- Variance, Bias tác nhân chính là do các inducer
- Mục tiêu của chúng ta là lựa chọn và xây dựng các inducer như thế nào để giảm lỗi variance và bias

Kinh nghiệm thực nghiệm

- Mô hình đơn giản có bias error cao hơn và variance error nhỏ hơn mô hình phứt tạp
- Bagging có cơ chế thu nhỏ variance, Adaboost thu nhỏ cả 2
- Trong trường hợp cụ thể thì dường như bias sẽ giảm trong những tác động ban đầu còn variance tăng trong những tác động phía sau

Nguyên tắc Occam’s Razor

- William Occam nhà triết học Anh thế kỷ thứ 14
- [Domingos (1999)] áp dụng trong nhận dạng
  - Nguyên tắc 1 : nếu 2 phân lớp có cùng lỗi tổng hợp (generation error), thì nên chọn phân lớp đơn giản hơn.
  - Nguyên tắc 2 : nếu 2 phân lớp có cùng lỗi tập huấn luyện (training set error), thì nên chọn phân lớp đơn giản hơn.

Thực nghiệm

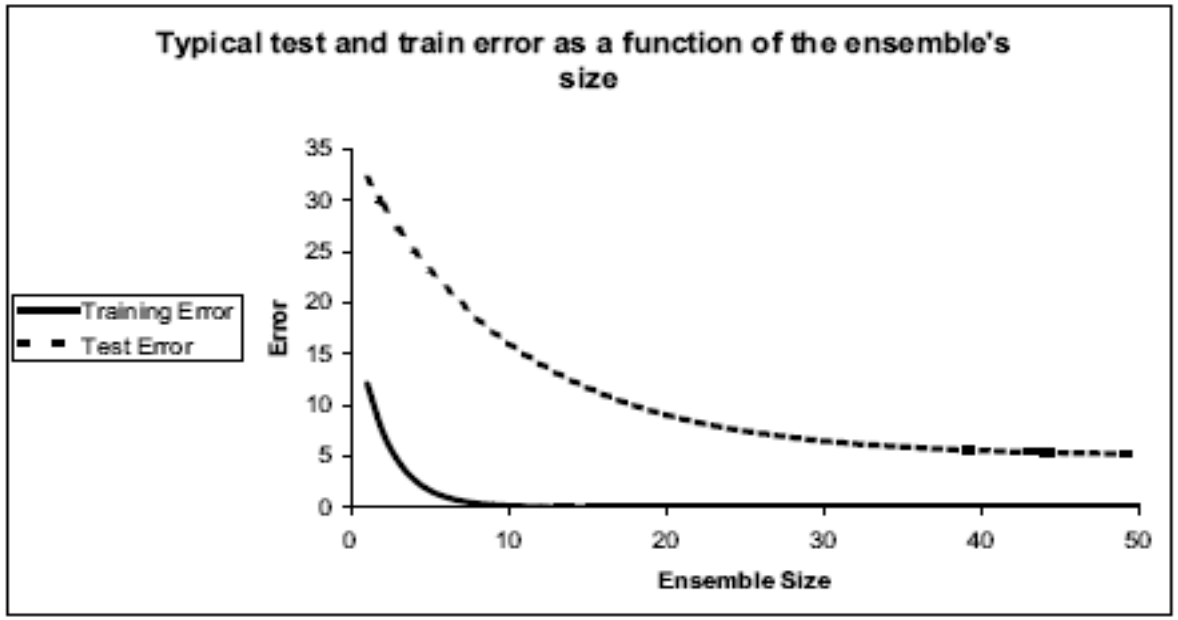


Fig. 2.15 Graphs of the training and test errors produced by an ensemble algorithm as a function of its size.

Lỗi huấn luyện sẽ giảm tới zero trong các mẫu đầu tiên (tới 10), Và cả hai gần như bằng nhau ở 20. Theo nguyên tắc ta nên chọn tổ hợp đơn giản nhất tức chọn 20

Những luật kết hợp thông dụng

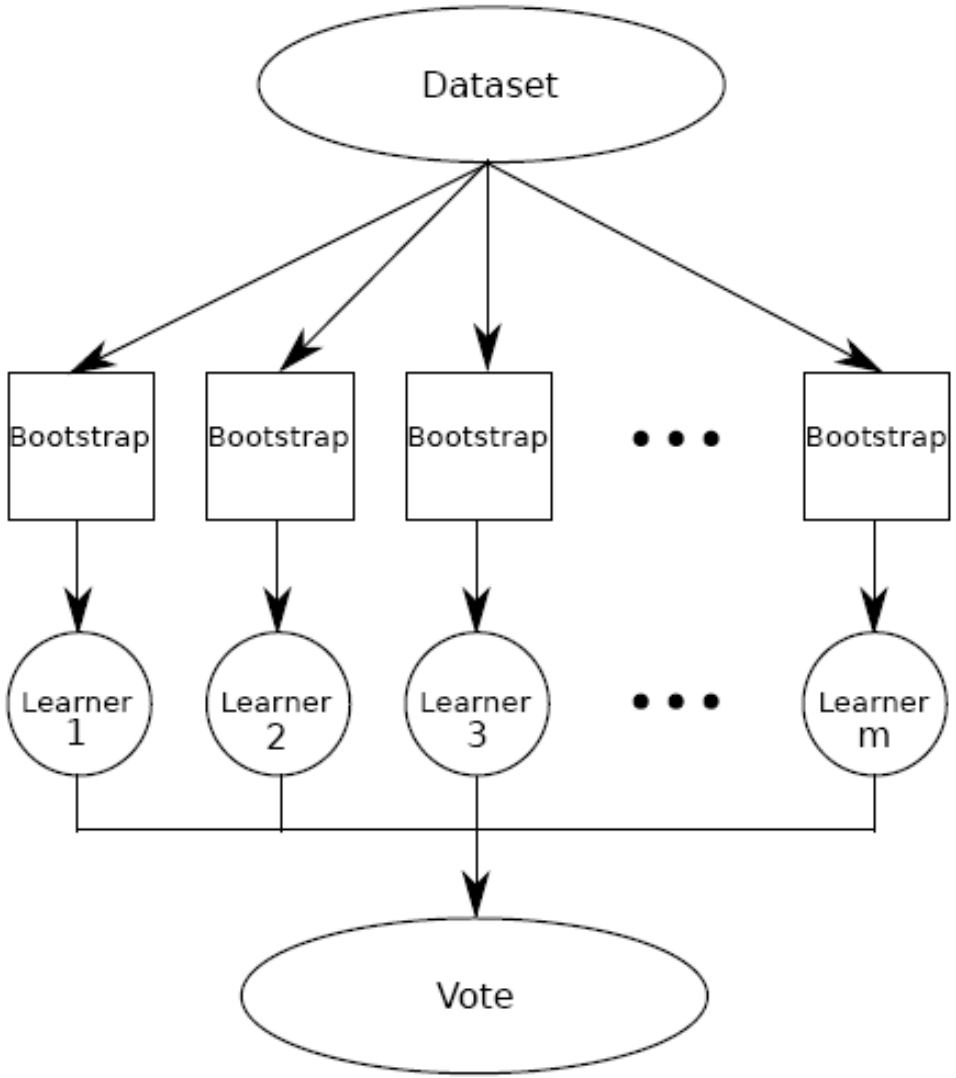
Rule	Fusion function $f(\cdot)$
Sum	$y_i = \frac{1}{L}\sum_{j=1}^{L} d_{ji}$
Weighted sum	$y_i = \sum_j w_j d_{ji}, w_j \geq 0, \sum_j w_j = 1$
Median	$y_i = \text{median}_j d_{ji}$
Minimum	$y_i = \min_j d_{ji}$
Maximum	$y_i = \max_j d_{ji}$
Product	$y_i = \prod_j d_{ji}$

	C_1	C_2	C_3
d_1	0.2	0.5	0.3
d_2	0.0	0.6	0.4
d_3	0.4	0.4	0.2
Sum	0.2	**0.5**	0.3
Median	0.2	**0.5**	0.4
Minimum	0.0	**0.4**	0.2
Maximum	0.4	**0.6**	0.4
Product	0.0	**0.12**	0.032

Thuật toán Bagging
(Bootstrap Aggregating)

- Một thuật toán đơn giản, hiệu quả trong học tổ hợp
- Ý tưởng là từ tập huấn luyện ta tạo ra các boostrap và huấn luyện thành các lớp độc lập với nhau. Dựa trên luật Majority Voting cho các lớp độc lập này cho 1 phân lớp tổng hợp.
- Boostrap có thể hiểu là 1 tập con của tập huấn luyện

Mô hình


Dataset
Bootstrap
Bootstrap
Bootstrap
Bootstrap
Learner 1
Learner 2
Learner 3
Learner m
Vote

Các tham số

- I :một cảm ứng cơ sở (a base inducer)
- B :hàm tạo ra 1 boostrap
- T :số phân lớp yếu được tạo ra
- S :tập huấn luyện nguồn
- μ :kích thước mẫu
- X :phần tử cần phân lớp
- N :số phân lớp

Thuật toán huấn luyện

- Input : I,B,T,S, μ
- Output : C_1, C_2, .., C_T

For i:=1 to T do

S_i=B(S, μ); // tạo một boostap thứ i

C_i=I(S_i); // tạo một phân lớp yếu C_i

End for;

Thuật toán phân lớp

- Input : X
- Output : class

$cout_1$<-0; …;$cout_N$

for i:=1 to T do

K<-Predic(Ci,X) // lấy dự đoán phân lớp Ci thuộc lớp thứ K

$cout_K$=$cout_K$+1; //tăng bình chọn lớp thứ k

end for;

class<- index (max($cout_1$,..,$cout_N$))

Giới thiệu

- Adaboost = Adaptive Boosting = tăng cường thích nghi (R.Scharpire, Y.Freund, ICML, 1996)

- **Ý tưởng chính**:
  - Đánh trọng số cho từng mẫu huấn luyện.
  - Cập nhật bộ trọng số sau mỗi lần lặp để tập trung (focus) vào các mẫu khó phân lớp (phân lớp sai).
  - kết hợp nhiều bộ phân lớp yếu (weak learners) thành một bộ phân lớp mạnh (strong learners)

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

Adaboost

- Tăng độ chính xác bài toán phân lớp
- Sử dụng với nhiều bài toán phân lớp khác nhau
- Ứng dụng rộng rãi trong các lĩnh vực (nhận dạng mặt người,…)
- Dễ dàng cài đặt

Một số thuật ngữ & ký hiệu

- Bộ phân lớp = Learner = Hypothesis = Classifier
- $h_t(x)$: Bộ phân lớp yếu (weak learner), phải thỏa điều kiện < 50% tỷ lệ lỗi trên một phân bố bất kỳ.
- H(x) : Bộ phân lớp mạnh, là sự kết hợp tuyến tính của các bộ phân lớp yếu.
- Tập huấn luyện $(x_1, y_1),\ldots,(x_m,y_m)$ với $x_i \in X$, $y_i \in \{-1,1\}$
- T: số lần lặp
- $D_t(i)$: trọng số của x_i tại bước lặp thứ t

Thuật toán Adaboost (huấn luyện)

- Input
  - Tập huấn luyện $(x_1,y_1),..,(x_m,y_m)$, $x_i \in X$, $y_i \in \{-1,1\}$
  - Hàm WeakLearn để tìm bộ phân lớp yếu dựa vào phân bố của mẫu huấn luyện (D_t)
  - Số lần lặp T.

- Output
  - T bộ phân lớp yếu và trọng số tương ứng (h_t,α_t) , t=1..T

Thuật toán Adaboost (huấn luyện)

1: Khởi tạo trọng số cho các mẫu $D_1(i) \leftarrow 1/m$; i=1..m

2: For t=1,…,T

3: Gọi hàm WeakLearner trả về 1 bộ phân lớp yếu h_t: X → {-1,1}

4: Tính tỷ lệ lỗi ε_t của h_t (tổng số mẫu học sai có tính trọng số)

$$\varepsilon_t = \sum_{i:h_t(x_i)\neq y_i} D_t(i)$$

5: Nếu $\varepsilon_t > 0.5$ thì dừng lặp

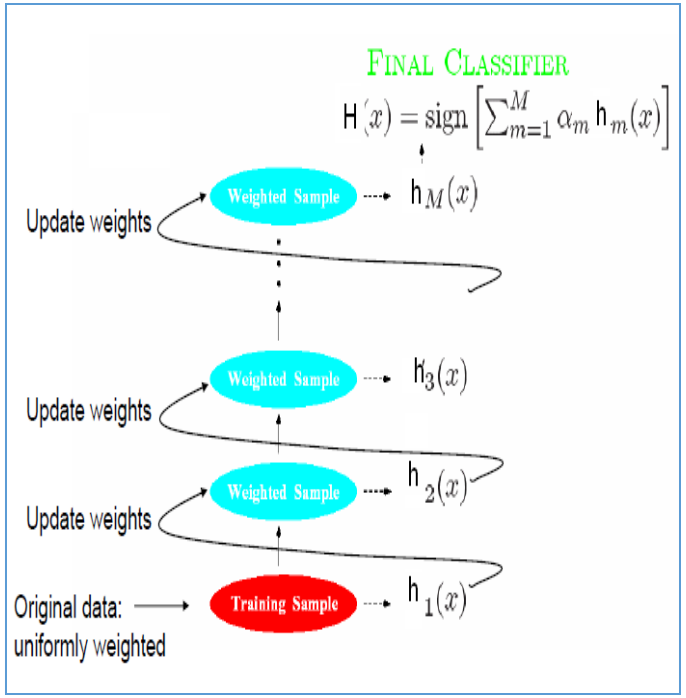
6: Tính độ chính xác α_t của h_t

$$\alpha_t \leftarrow \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

7: Cập nhật bộ trọng số D_t cho bước tiếp theo

$$D_{t+1}(i) = \frac{D_t(i).e^{-\alpha_t y_t h_t(x_i)}}{Z_t}$$

8: End for

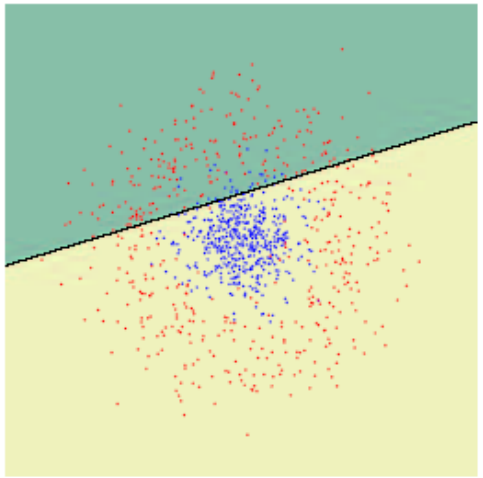


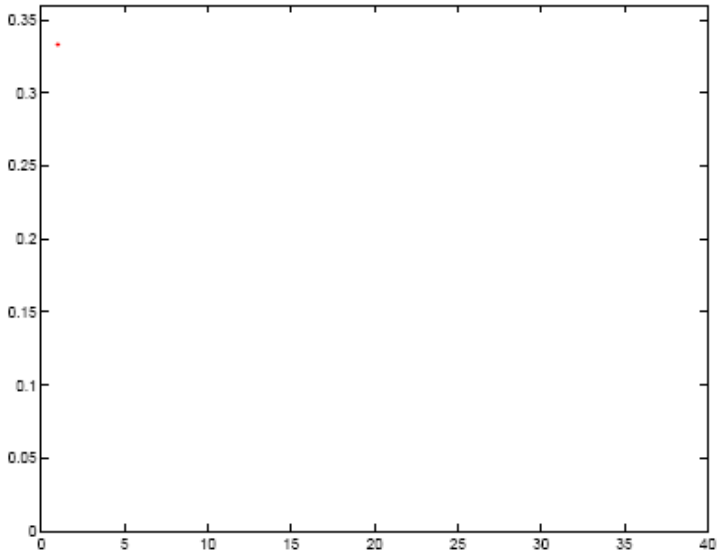
➔ **Kết quả cuối cùng**:

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

Thuật toán Adaboost (huấn luyện)

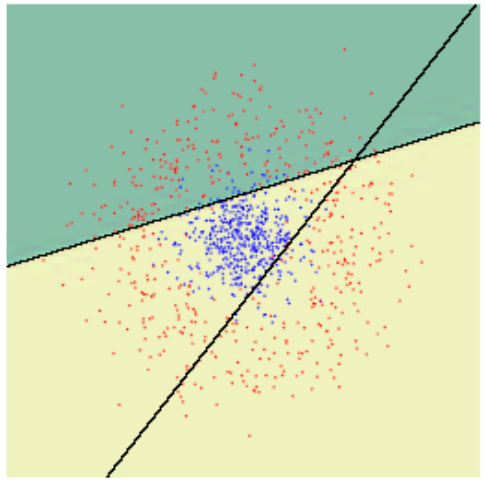
$$t = 1$$

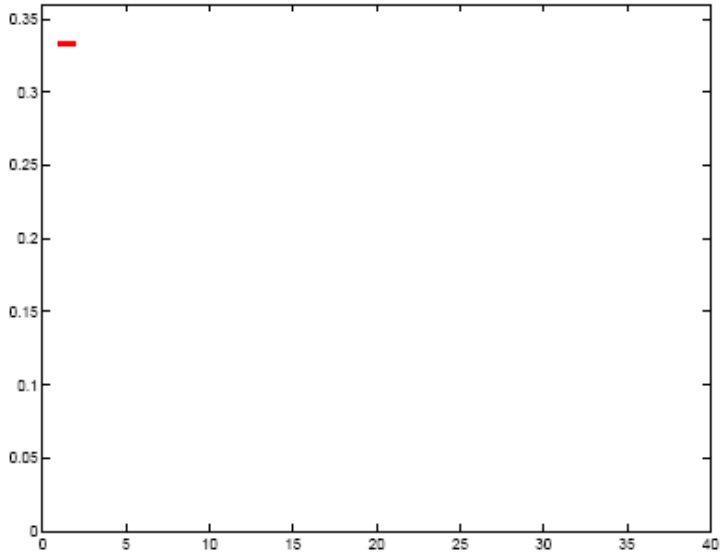




Thuật toán Adaboost (huấn luyện)

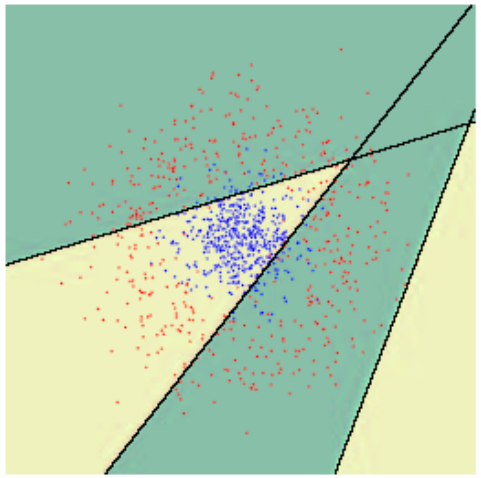
$t = 2$

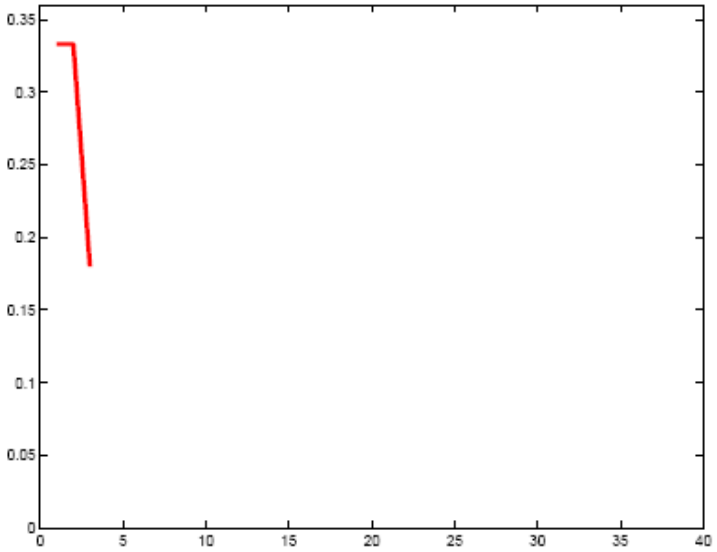




Thuật toán Adaboost (huấn luyện)

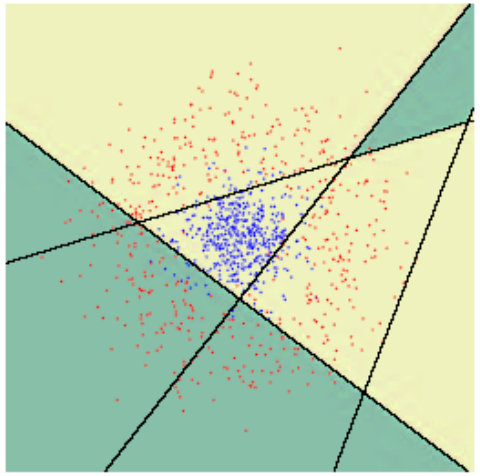
$$t = 3$$

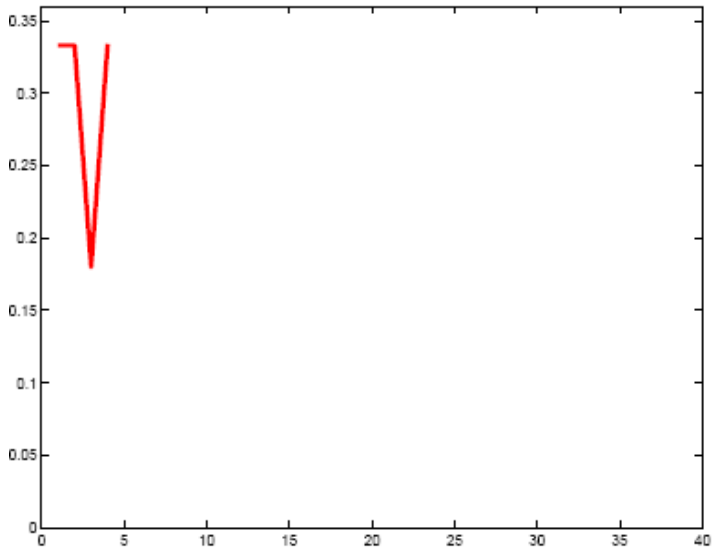




Thuật toán Adaboost (huấn luyện)

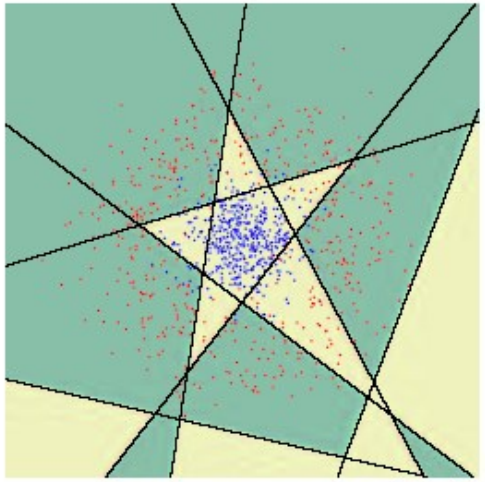
$$t = 4$$

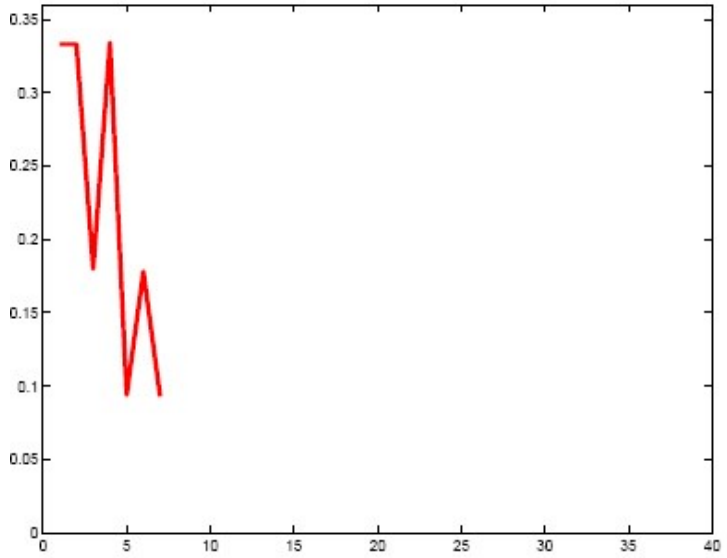




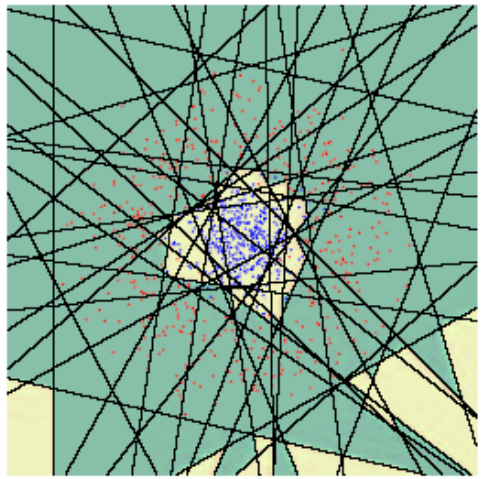
Thuật toán Adaboost (huấn luyện)

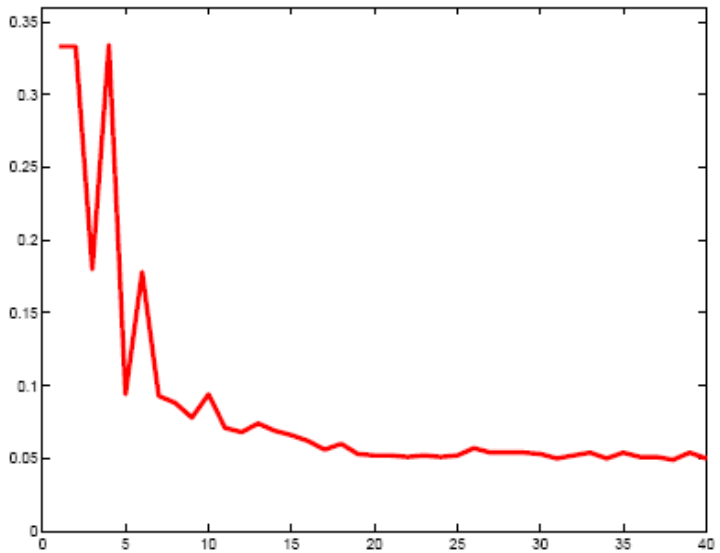
$$t = 7$$





$t = 40$





Thuật toán Adaboost (huấn luyện)

Hàm WeakLearn

- Trong các bước lặp, hàm WeakLearn được gọi để chọn ra bộ phân lớp yếu h_t: X → {-1,1} từ tập H={h(x)}

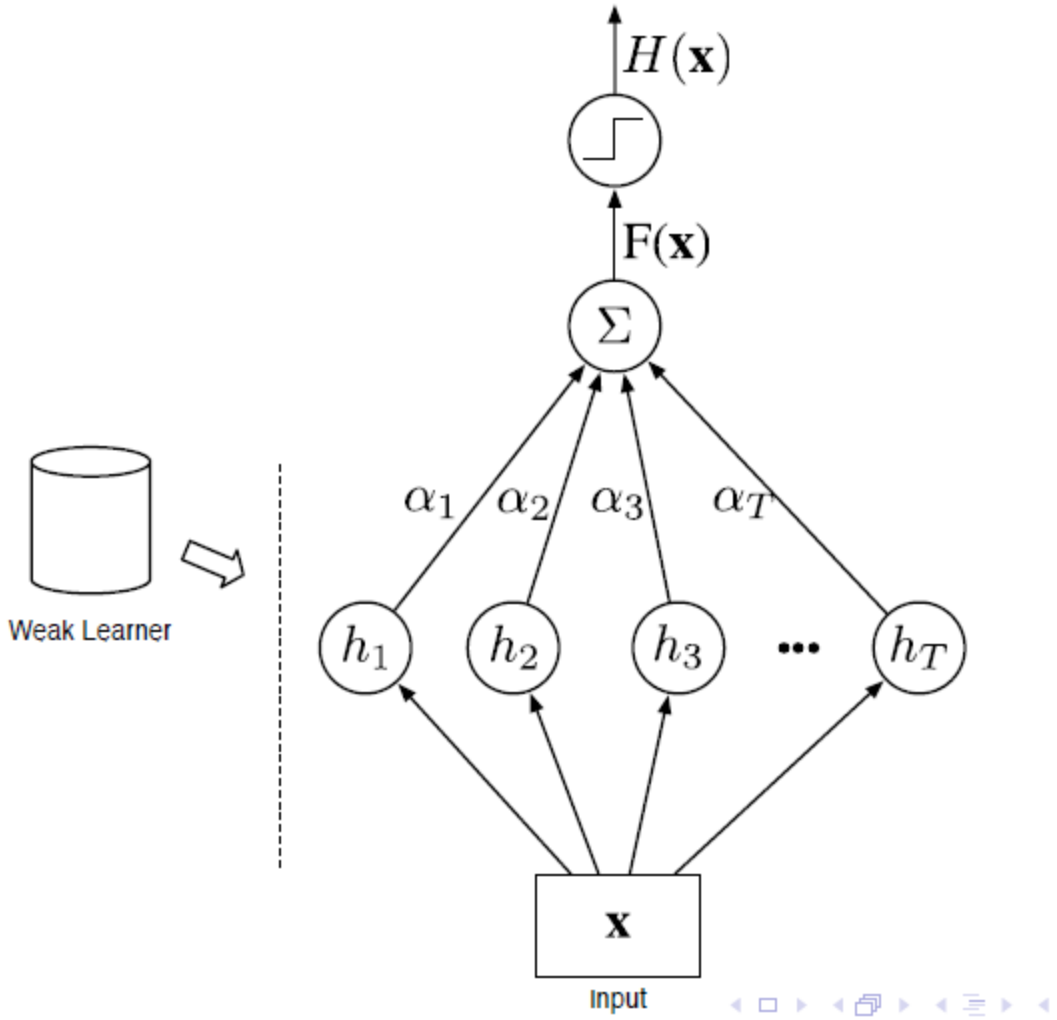
- Tiêu chí: chọn bộ phân lớp h_t với tỷ lệ lỗi thấp nhất

$$h_t = \arg\min_{h_j \in H} \varepsilon_j = \sum_{i=1}^{n} D_t(i)[y_i \neq h_j(x_i)]$$

- Một số ví dụ hàm WeakLearn
  - Decision stumps (axis pararell splits)
  - Decision tree (C4.5 by Quinlan 1996)
  - Multi-layer neural network (VD cho OCR)
  - Chọn từ tập H hữu hạn cho trước

Thuật toán Adaboost (phân lớp)

- Input : x
- Output: y=H(x)



Trọng số α_t

- Trọng số (độ chính xác) cho từng bộ phân lớp yếu luôn luôn là số dương

$$\epsilon_t(h_t) < \frac{1}{2} \Rightarrow \alpha_t = \frac{1}{2} \ln \frac{1 - \epsilon_t(h_t)}{\epsilon_t(h_t)} > 0$$

- Tỷ lệ lỗi ε_t càng nhỏ thì trọng số càng lớn α_t và bộ phân lớp yếu tương ứng sẽ có vai trò lớn trong kết quả cuối cùng.

$$\epsilon(h_A) < \epsilon(h_B) \Rightarrow \alpha_A > \alpha_B$$

Vi sao ε_t<0.5 ?

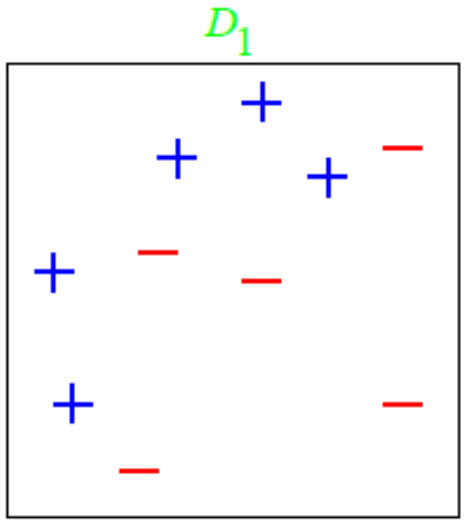
- Khái niệm bộ phân lớp yếu: là bộ phân lớp tốt hơn bộ phân lớp ngẫu nhiên, và tỷ lệ lỗi > 50%

- Nếu tỷ lệ lỗi > 0.5 thì sẽ làm giảm độ chính xác của bộ phân lớp mạnh cuối cùng.

$$\epsilon_t(h_t) > \frac{1}{2} \Rightarrow \alpha_t = \frac{1}{2} \ln \frac{1 - \epsilon_t(h_t)}{\epsilon_t(h_t)} < 0$$

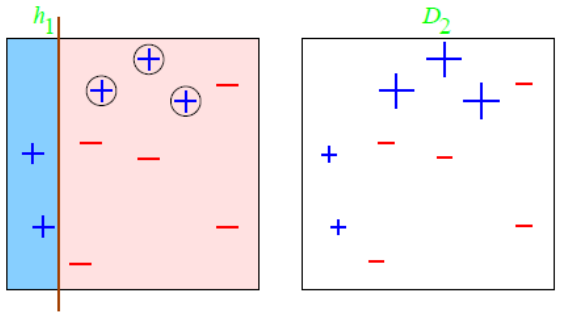
Ví dụ minh họa

- Tập huấn luyện gồm 10 mẫu phân thành 2 lớp + hoặc - (gồm 5 mẫu thuộc lớp + và 5 mẫu thuộc lớp -). Số lần lặp là T=3.



Ví dụ minh họa

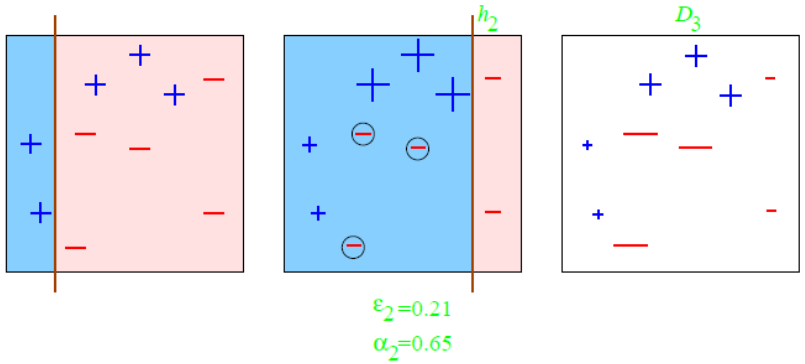
- *Lặp lần 1*:



- Tìm được bộ phân lớp yếu h1 với tỷ lệ lỗi ε_1=0.3, và độ chính xác α_1=0.42.
- Cập nhật lại bộ trọng D1 thành D2.
- Các mẫu phân lớp sai (được khoanh tròn) sẽ được tăng trọng số.

Ví dụ minh họa

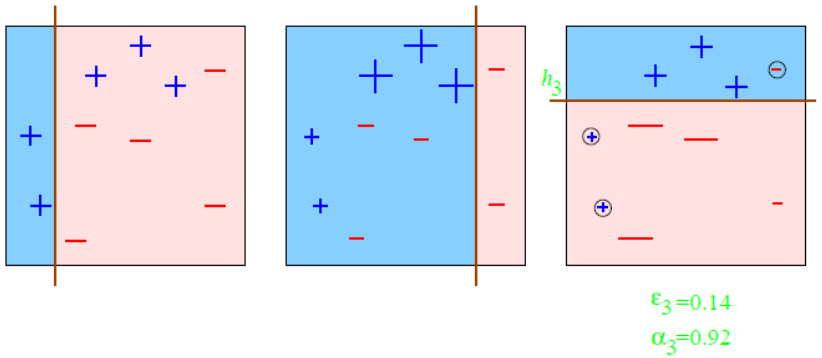
- *<u>Lặp lần 2</u>*:



- Tìm được bộ phân lớp yếu h2 với tỷ lệ lỗi ε_1=0.21, và độ chính xác α_1=0.65.
- Cập nhật lại bộ trọng D2 thành D3.
- Các mẫu phân lớp sai (được khoanh tròn) sẽ được tăng trọng số.

Ví dụ minh họa

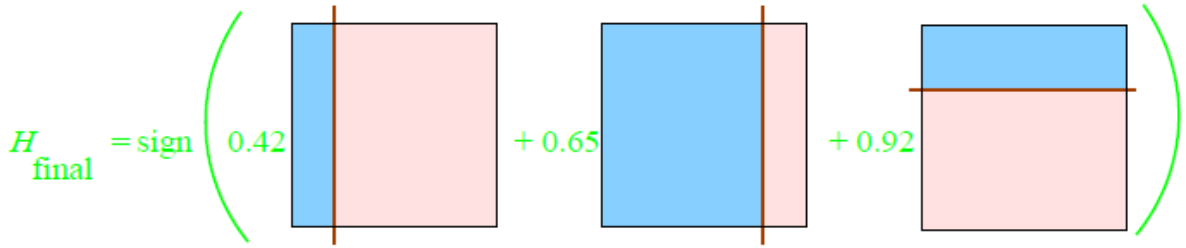
- *Lặp lần 3*:

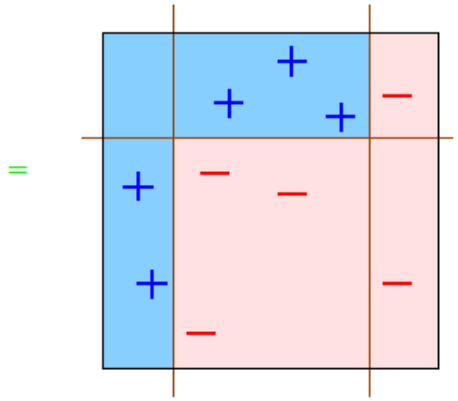


- Tìm được bộ phân lớp yếu h3 với tỷ lệ lỗi ε_1=0.14, và độ chính xác α_1=0.92. Thuật toán dừng sau 3 vòng lặp.

Ví dụ minh họa

- *<u>Kết quả</u>*:





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