Supervised Machine learning
(Học có giám sát)

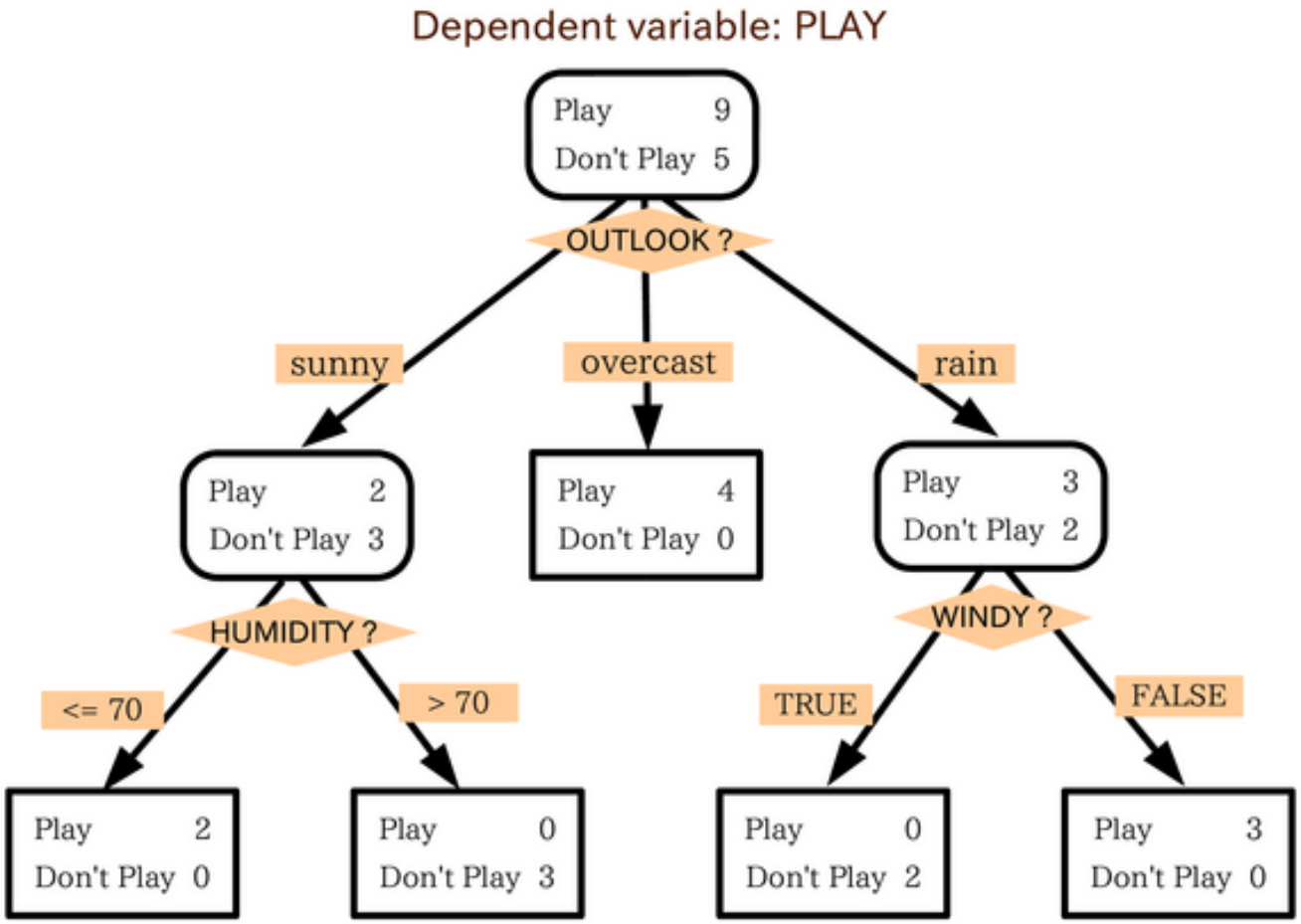
PGS.TS. Hoàng Văn Dũng

dunghv@hcmute.edu.vn

Nội dung

1. Cây quyết định
2. Bộ phân lớp Random Trees
3. Các bộ phân lớp Boosting

CÂY QUYẾT ĐỊNH



Cây quyết định

- Cây quyết định là một cây phân cấp có cấu trúc
- Dùng để phân lớp các đối tượng dựa vào dãy các luật (quy tắc).
- Thuộc tính của đối tượng có thể thuộc các kiểu dữ liệu khác nhau trong khi đó thuộc tính phân lớp phải có kiểu dữ liệu là nhị phân (Binary) hoặc có thứ tự (Ordinal)

Cây quyết định

- Là một giải thuật học đơn giản nhưng thành công
- Cây quyết định là một cách biểu diễn cho phép chúng ta xác định phân loại của một đối tượng bằng cách kiểm tra giá trị của một số thuộc tính.
- Giải thuật có:
  - **Đầu vào:** Một đối tượng hay một tập hợp các thuộc tính mô tả một tình huống
  - **Đầu ra:** thường là quyết định yes/no, hoặc các phân loại.
- Trong cây quyết định:
  - Mỗi nút trong biểu diễn một sự kiểm tra trên một thuộc tính nào đó, mỗi giá trị có thể của nó tương đương với một nhánh của cây
  - Các nút lá thể hiện sự phân loại.
- Kích cỡ của cây QĐ tùy thuộc vào thứ tự của các kiểm tra trên các thuộc tính.

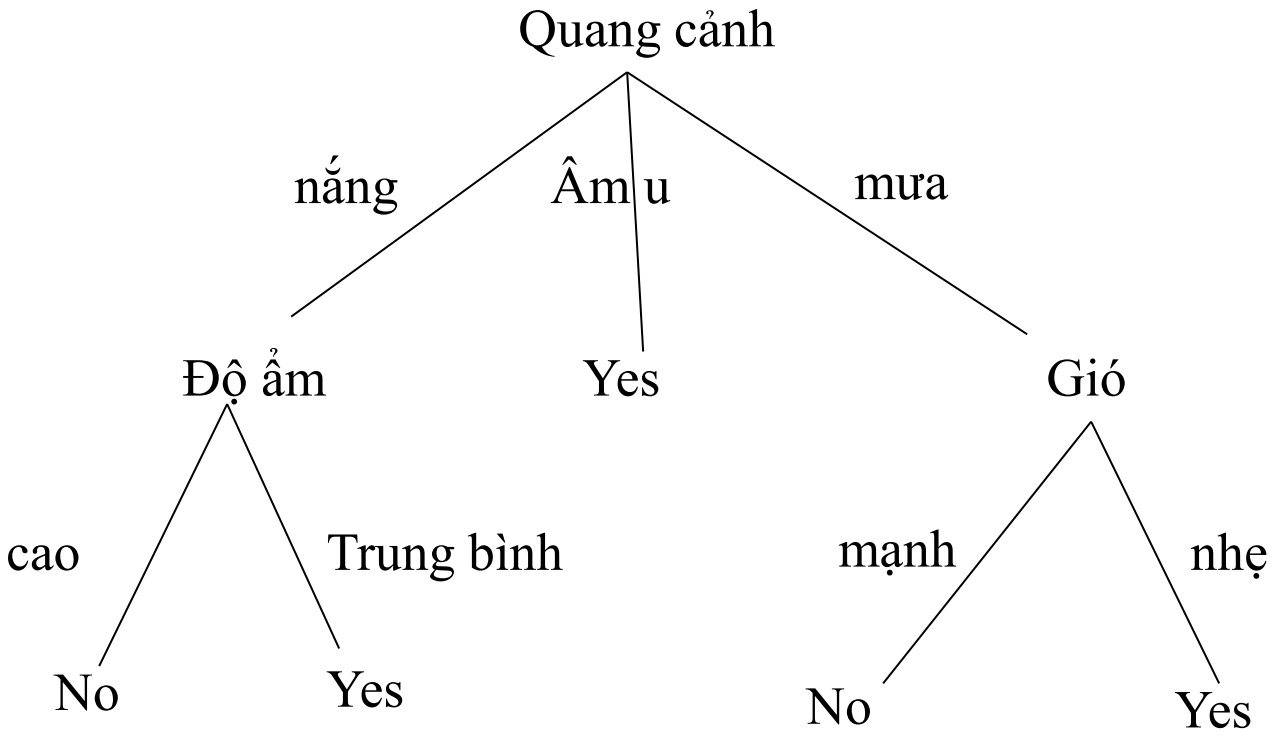
Quy nạp cây QĐ từ các mẫu

- Mẫu (hay dữ liệu rèn luyện cho hệ thống) gồm:
  Giá trị của các thuộc tính + Phân loại của mẫu

Ngày	Quang cảnh	Nhiệt độ	Độ ẩm	Gió	Chơi Tennis
D1	Nắng	Nóng	Cao	nhẹ	Không
D2	Nắng	Nóng	Cao	Mạnh	Không
D3	Âm u	Nóng	Cao	Nhẹ	Có
D4	Mưa	ấm áp	Cao	nhẹ	Có
D5	Mưa	Mát	TB	nhẹ	Có
D6	Mưa	Mát	TB	Mạnh	Không
D7	Âm u	Mát	TB	Mạnh	Có
D8	Nắng	ấm áp	Cao	nhẹ	Không
D9	Nắng	Mát	TB	nhẹ	Có
D10	Mưa	ấm áp	TB	nhẹ	Có
D11	Nắng	ấm áp	TB	Mạnh	Có

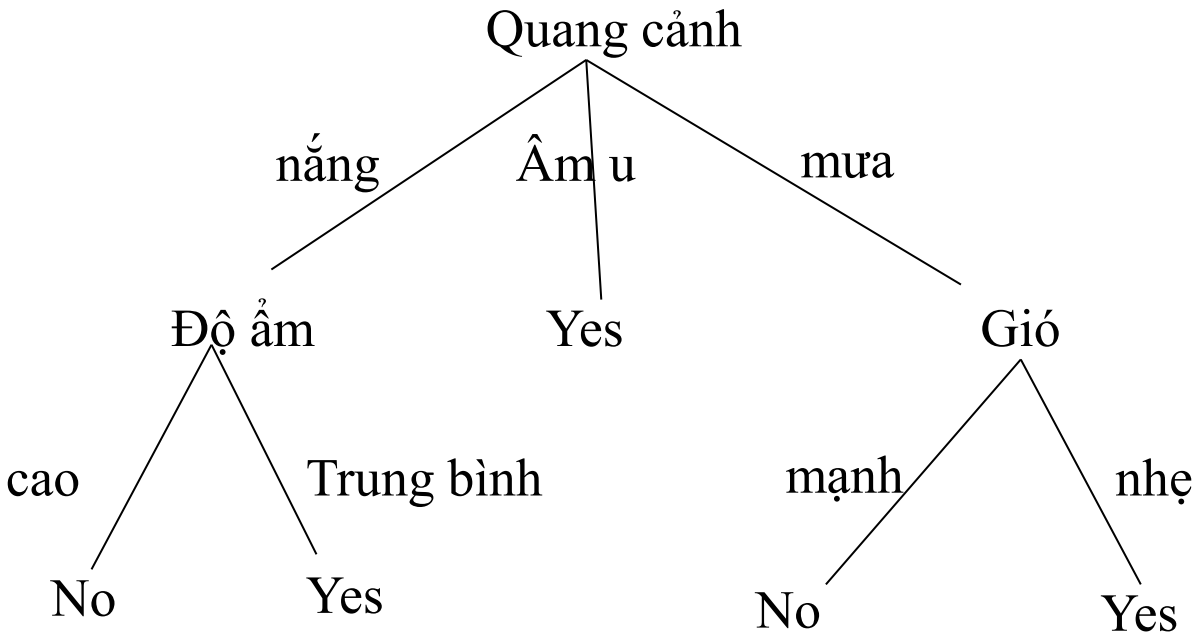
Ví dụ

- Mục đích: học từ dữ liệu để dự đoán xem có chơi thể thao không?
- Cây quyết định:



Chuyển cây thành các luật

If (Quang-cảnh =nắng) ∧ (Độ ẩm = Cao) Then Chơi-Tennis = No
If (Quang-cảnh =nắng) ∧ (Độ ẩm = TB) Then Chơi-Tennis = Yes
If (Quang-cảnh =Âm u) Then Chơi-Tennis = Yes
…



Làm sao để học được cây QĐ

- Tiếp cận đơn giản
  - Học một cây mà có một lá cho mỗi mẫu.
  - Học tất cả các mẫu.
  - Có thể sẽ không thực hiện tốt trong các trường hợp khác.
- Tiếp cận tốt hơn:
  - Học một cây nhỏ nhưng chính xác phù hợp với các mẫu
  - Occam's razor – cái đơn giản thường là cái tốt nhất!

    Giả thuyết có khả năng nhất là giả thuyết đơn giản nhất thống nhất với tất cả các quan sát.

Xây dựng cây QĐ: Trên - xuống

Vòng lặp chính:

1. A <- thuộc tính quyết định "tốt nhất" cho nút kế
2. Gán A là thuộc tính quyết định cho nút
3. Với mỗi giá trị của A, tạo một nút con mới cho nút
4. Sắp xếp các mẫu vào các nút lá
5. If các mẫu đã được phân loại thì dừng
   Else lặp lại trên mỗi nút lá mới

Để phân loại một trường hợp, có khi cây QĐ không cần sử dụng tất cả các thuộc tính đã cho, mặc dù nó vẫn phân loại đúng tất cả các mẫu.

Các khả năng có thể của nút con

- Các mẫu có cả âm và dương (negative, positive):
  - Tách một lần nữa
- Tất cả các mẫu còn lại đều neg hoặc đều Pos
  - Trả về cây quyết định
- Không còn mẫu nào
  - Trả về mặc định
- Không còn thuộc tính nào (nhiễu)
  - Quyết định dựa trên một luật nào đó. VD luật đa số

Ví dụ cây quyết định

- Bài toán đoán nhận lựa chọn phương tiện giao thông

Gender	Car ownership	Travel Cost	Income Level	Transportation mode
Male	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Female	1	Cheap	Medium	Train
Female	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Male	0	Standard	Medium	Train
Female	1	Standard	Medium	Train
Female	1	Expensive	High	Car
Male	2	Expensive	Medium	Car
Female	2	Expensive	High	Car

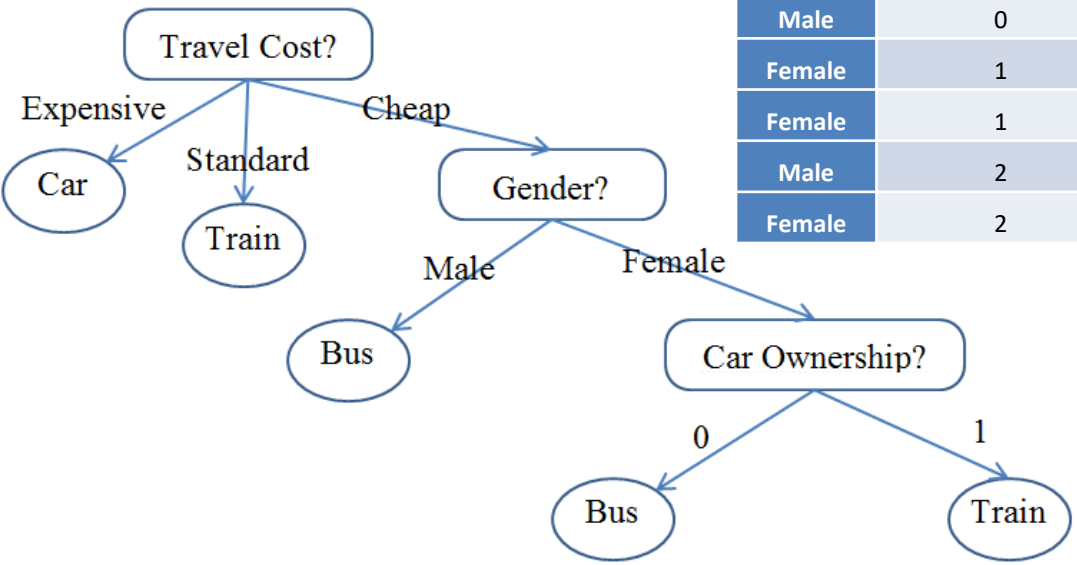
Học quy luật

Dự đoán

Person name	Gender	Car ownership	Travel Cost	Income	Transportation Mode
Tran	Male	1	Standard	High	?
Nguyen	Male	0	Cheap	Medium	?
Pham	Female	1	Cheap	High	?

Ví dụ

Gender	Car ownership	Travel Cost	Income Level	Transportation mode
Male	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Female	1	Cheap	Medium	Train
Female	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Male	0	Standard	Medium	Train
Female	1	Standard	Medium	Train
Female	1	Expensive	High	Car
Male	2	Expensive	Medium	Car
Female	2	Expensive	High	Car



Person name	Gender	Car ownership	Travel Cost	Income	Transportation Mode
Tran	Male	1	Standard	High	Train
Nguyen	Male	0	Cheap	Medium	Bus
Pham	Female	1	Cheap	High	Train

Đánh giá

- Mục tiêu muốn có một cây có thể phân loại đúng một mẫu mà nó chưa từng thấy qua.
- Việc học sử dụng một "tập huấn luyện" (training set),

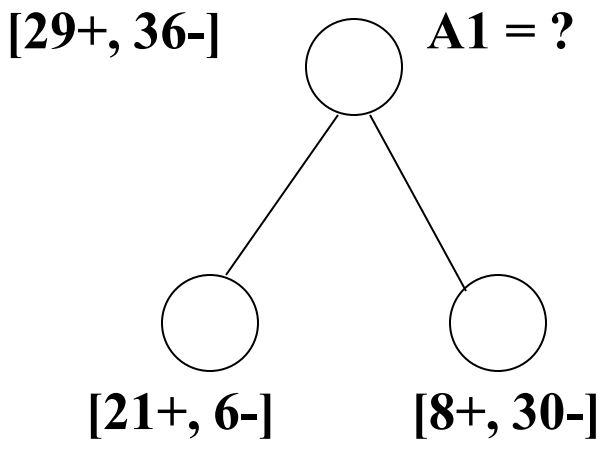
việc đánh giá hiệu suất sử dụng một "tập kiểm tra" (test set):

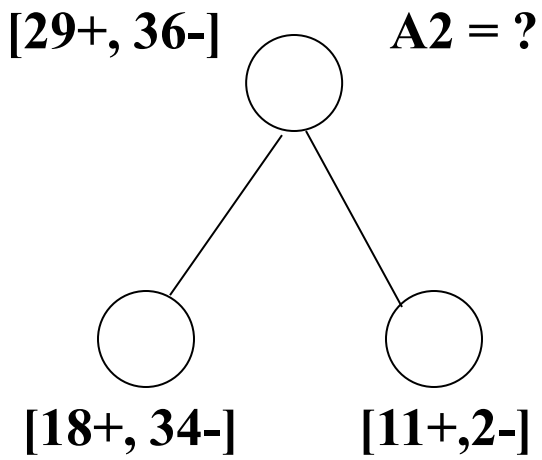
1. Thu thập một tập hợp lớn các mẫu
2. Chia thành tập huấn luyện và tập kiểm tra
3. Sử dụng giải thuật và tập huấn luyện để xây dựng giả thuyết *h* cho cây QĐ.
4. Đo phần trăm tập kiểm tra được phân loại đúng bởi *h*
5. Lặp lại bước 1 đến 4 cho các kích cỡ tập kiểm tra khác nhau được chọn một cách nhẫu nhiên.

Sử dụng lý thuyết thông tin

- Mục tiêu: chọn các thuộc tính nào đó để có thể giảm thiểu chiều sâu của cây QĐ.
- Thuộc tính tốt nhất: chia các mẫu vào các tập hợp chứa toàn mẫu âm hoặc mẫu dương.
- Chúng ta cần một phép đo để xác định thuộc tính nào cho khả năng chia tốt hơn.

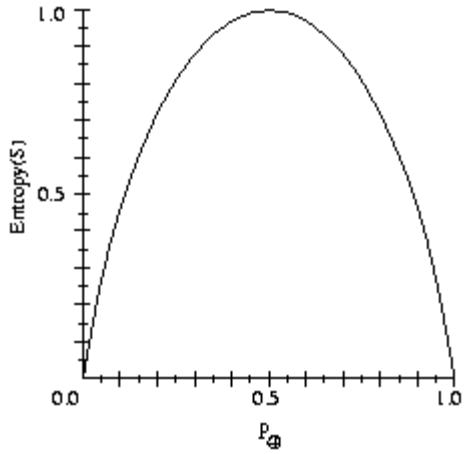
Thuộc tính nào tốt hơn?





Entropy

- Entropy(S) = số lượng mong đợi các bit cần thiết để mã hóa một lớp (+ hay –) của một thành viên rút ra một cách ngẫu nhiên từ S (trong trường hợp tối ưu, mã có độ dài ngắn nhất).
- Theo lý thuyết thông tin: mã có độ dài tối ưu là mã gán – $\log_2 p$ bits cho thông điệp có xác suất là p.



- S là một tập huấn luyện
- $p_\oplus$ là phần các mẫu dương trong tập S
- p_Θ là phần các mẫu âm trong tập S
- Entropy đo độ pha trộn của tập S:

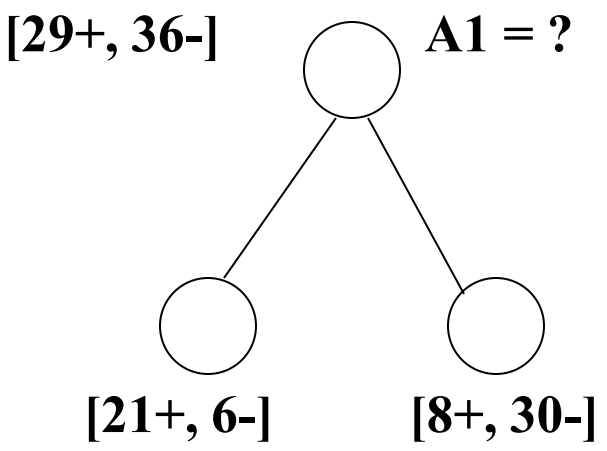
$$Entropy(S) = -p_\oplus \log_2 p_\oplus - p_\Theta \log_2 p_\Theta$$

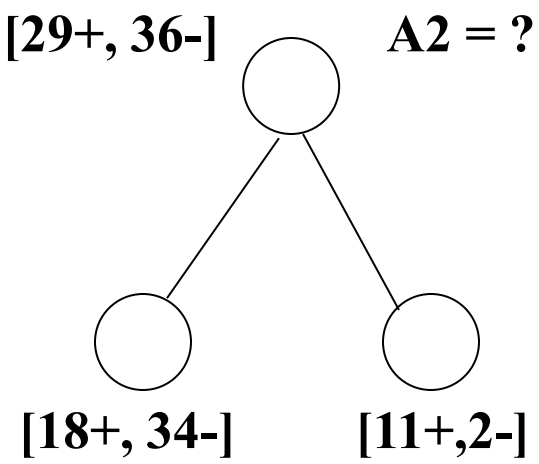
$$Entropy(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Lượng thông tin thu được Information Gain

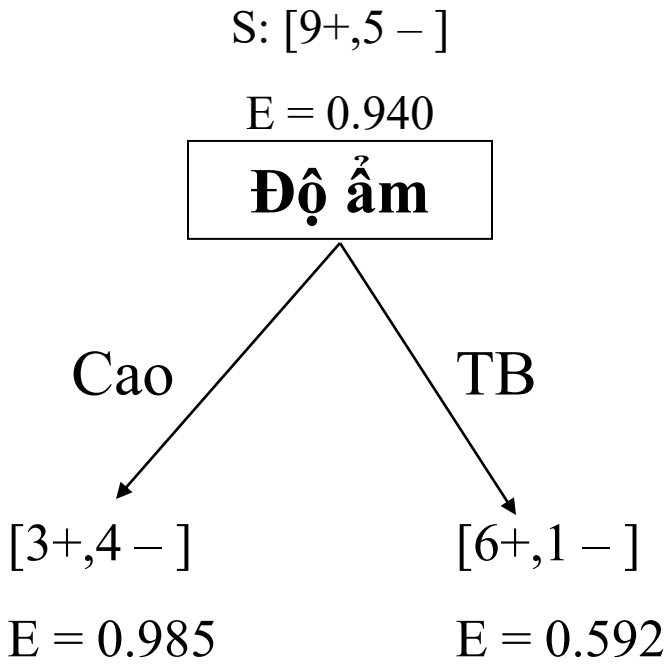
- Gain(S, A) = Lượng giảm entropy mong đợi qua việc chia các mẫu theo thuộc tính A

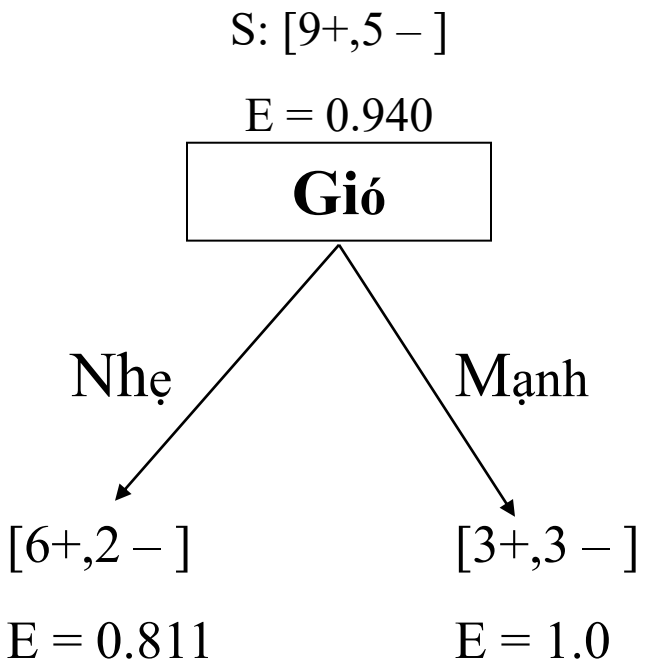
$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$





Chọn thuộc tính kế tiếp





Gain(S, Độ ẩm)
= .940 – (7/14).985 – (7/14).592
= .151

Gain(S, Gió)
= .940 – (8/14).811 – (6/14)1.0
= .048

Ví dụ tính Entropy

#	*Outlook*	*Temperature*	*Humidity*	*Wind*	*Target*
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Ví dụ tính Entropy

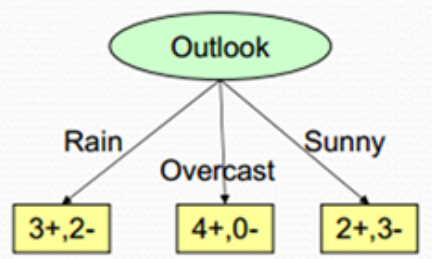
Áp dụng thuật toán ID3 cho bài toán học *chơi thể thao*:

Thực hiện trình tự các bước lặp của thuật toán, tuần tự các bước như sau:

Lặp lần 1: Xét lần lượt 4 thuộc tính (Outlook, Temperature, Humidity, Wind)

để tính giá trị H và AE tương ứng của nó.

Ví dụ thuộc tính Outlook ta có



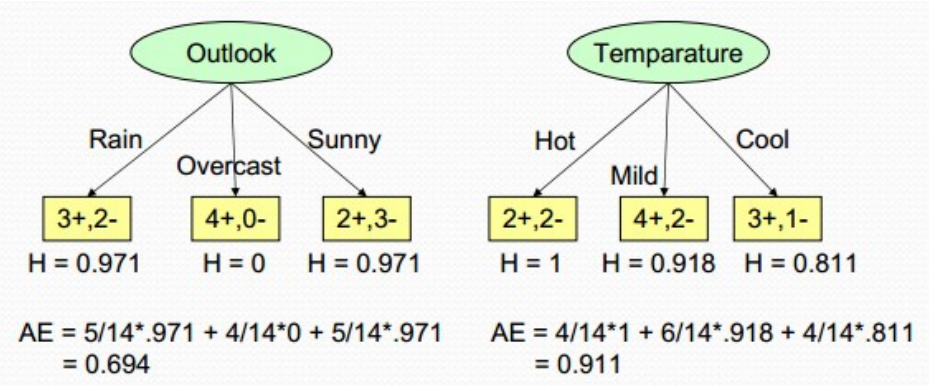
Vậy: $H_{Rain}=-(3/5)*\log_2(3/5)-(2/5)*\log_2(2/5)=0.971$

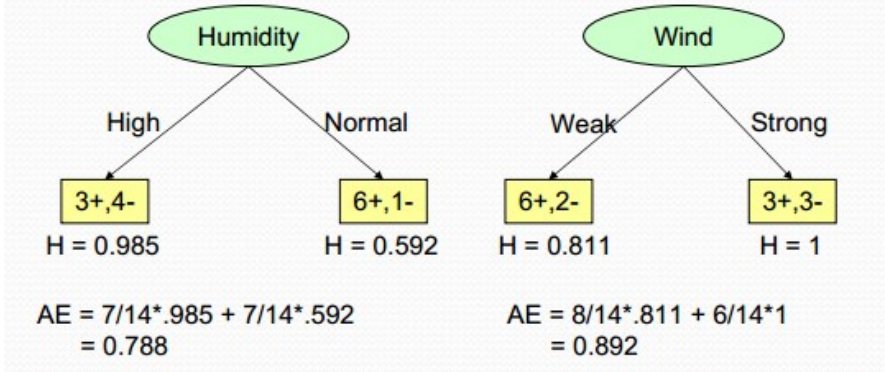
$H_{Overcast}=-(4/4)*\log_2(4/4)-(0/4)*\log_2(0/4)=0$

$H_{Sunny}=-(2/5)*\log_2(2/5)-(3/5)*\log_2(3/5)=0.971$

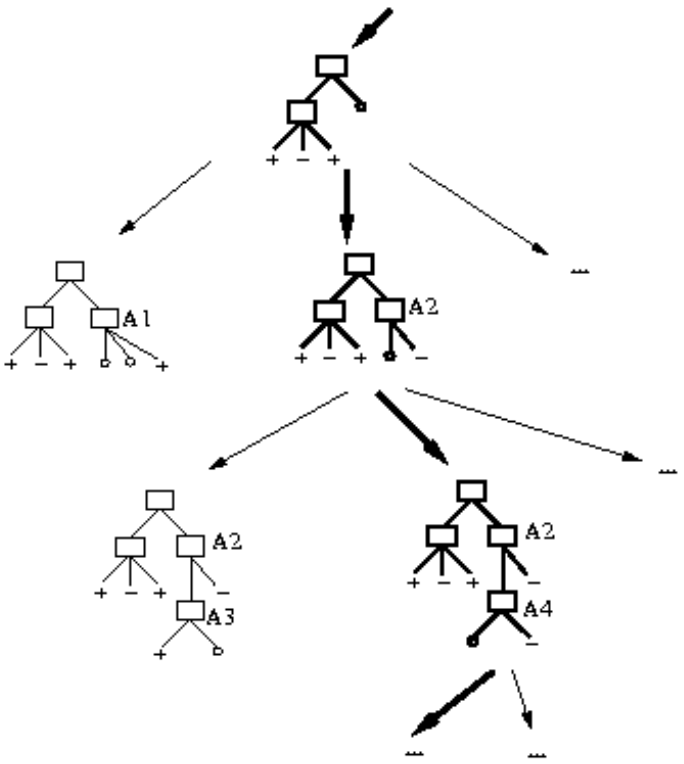
Tương tự cho các thuộc tính còn lại.

Tính AE cho các thuộc tính





Tìm kiếm không gian giả thuyết trong ID3 (1)



- Không gian giả thuyết đầy đủ =>giả thuyết chắc chắn thuộc KG này
- Đầu ra là một giả thuyết (cây QĐ) =>Cây nào? Không thể chọn cây với 20 câu hỏi
- Không quay lui => cực tiểu địa phương
- Lựa chọn tìm kiếm dựa trên thống kê => chịu được dữ liệu nhiễu
- Thiên lệch quy nạp: thích cây ngắn hơn.

Khi nào nên sử dụng cây QĐ

- Các mẫu được mô tả bằng các cặp "thuộc tính – giá trị", vd: Gió - mạnh, Gió - nhẹ
- Kết quả phân loại là các giá trị rời rạc, vd: Yes, No
- Dữ liệu rèn luyện có thể chứa lỗi (bị nhiễu)
- Dữ liệu rèn luyện có thể thiếu giá trị thuộc tính

Ví dụ:

- Phân loại bệnh nhân theo các bệnh của họ
- Phân loại hỏng hóc thiết bị theo nguyên nhân
- Phân loại người vay tiền theo khả năng chi trả

Ví dụ

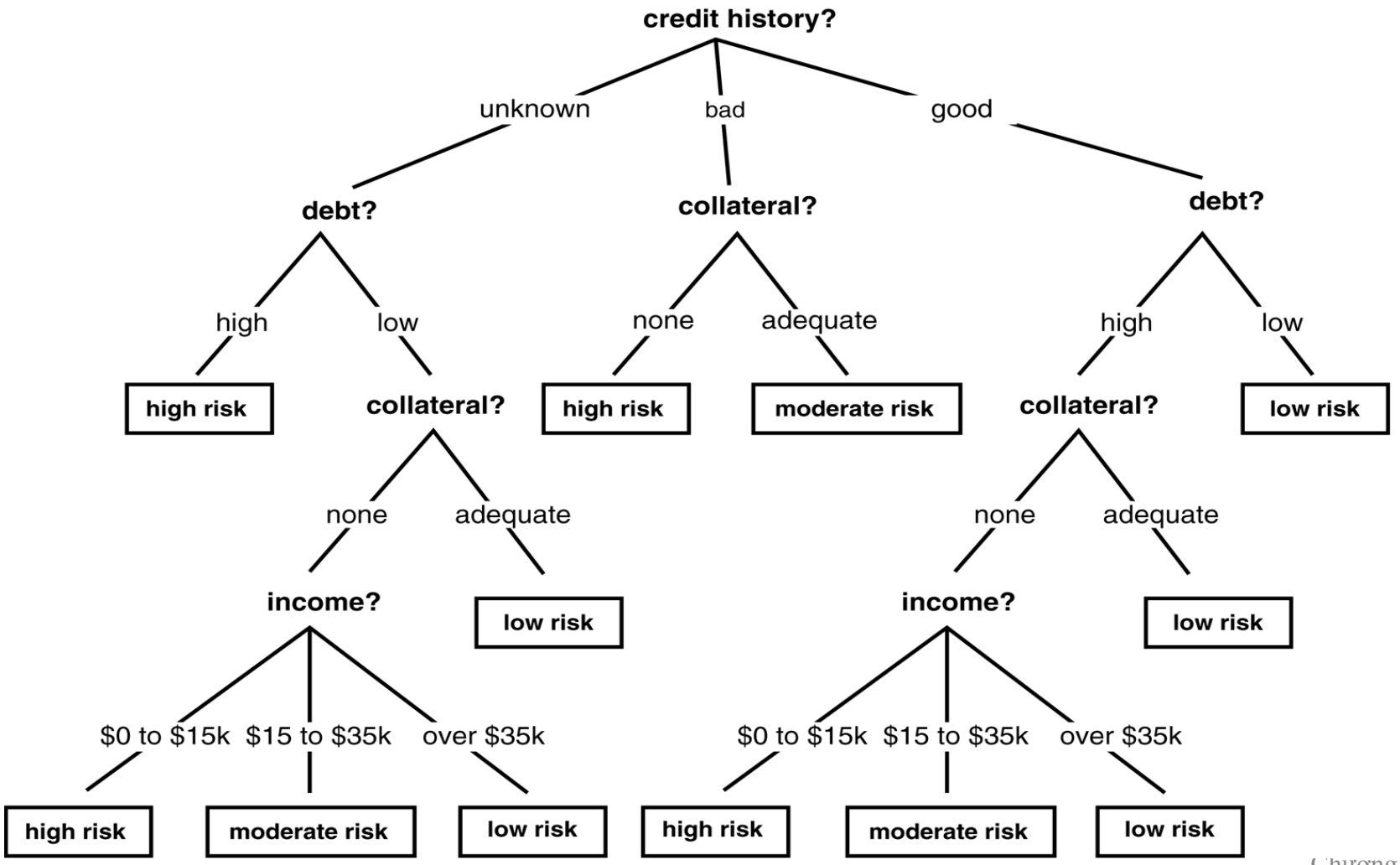
- Ước lượng độ an toàn của một tài khoản tín dụng

Data from credit history of loan applications.

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	$0 to $15k
2.	high	unknown	high	none	$15 to $35k
3.	moderate	unknown	low	none	$15 to $35k
4.	high	unknown	low	none	$0 to $15k
5.	low	unknown	low	none	over $35k
6.	low	unknown	low	adequate	over $35k
7.	high	bad	low	none	$0 to $15k
8.	moderate	bad	low	adequate	over $35k
9.	low	good	low	none	over $35k
10.	low	good	high	adequate	over $35k

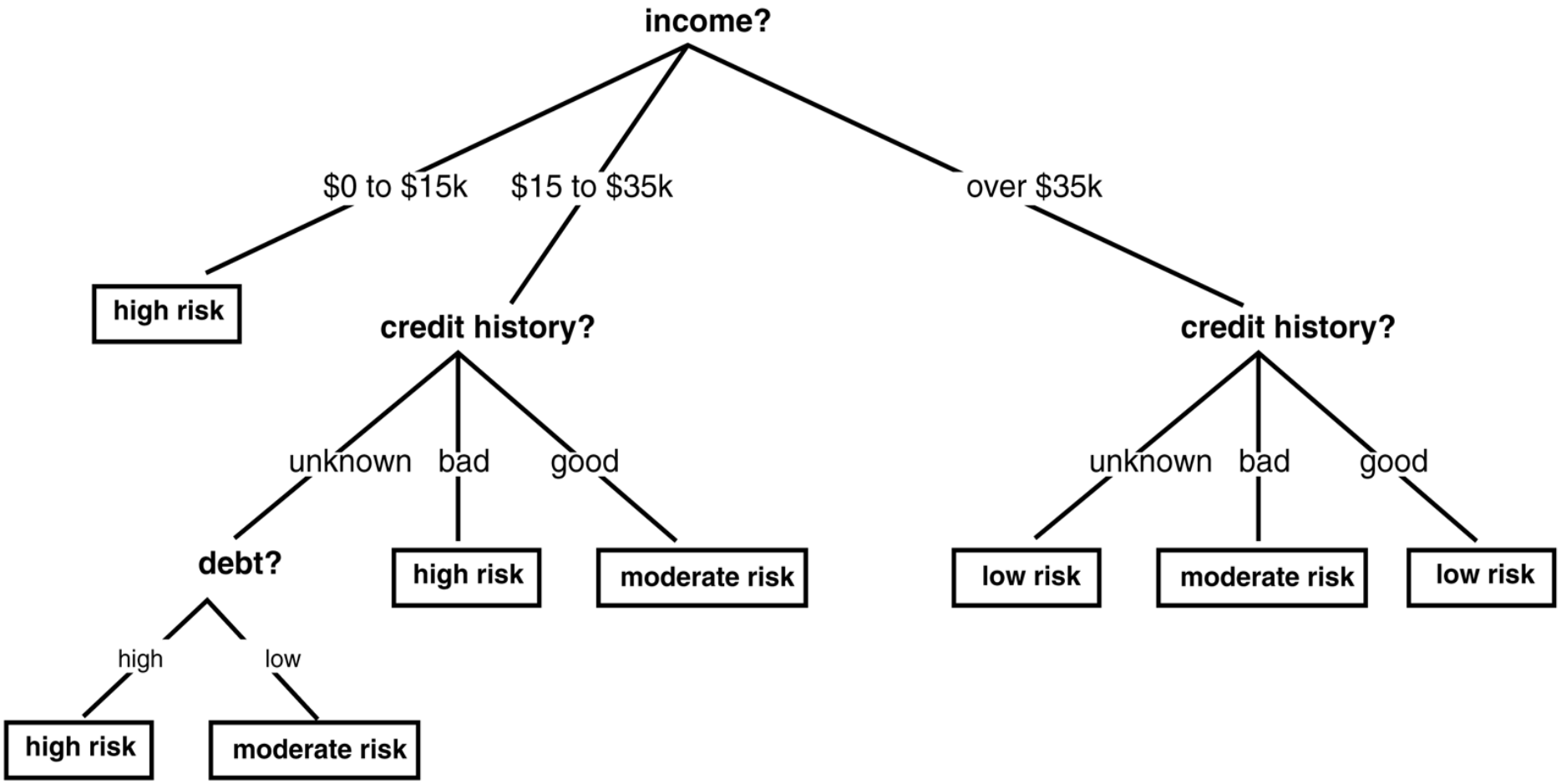
Ví dụ

- Một cây QĐ cho bài toán đánh giá độ an toàn của tín dụng.



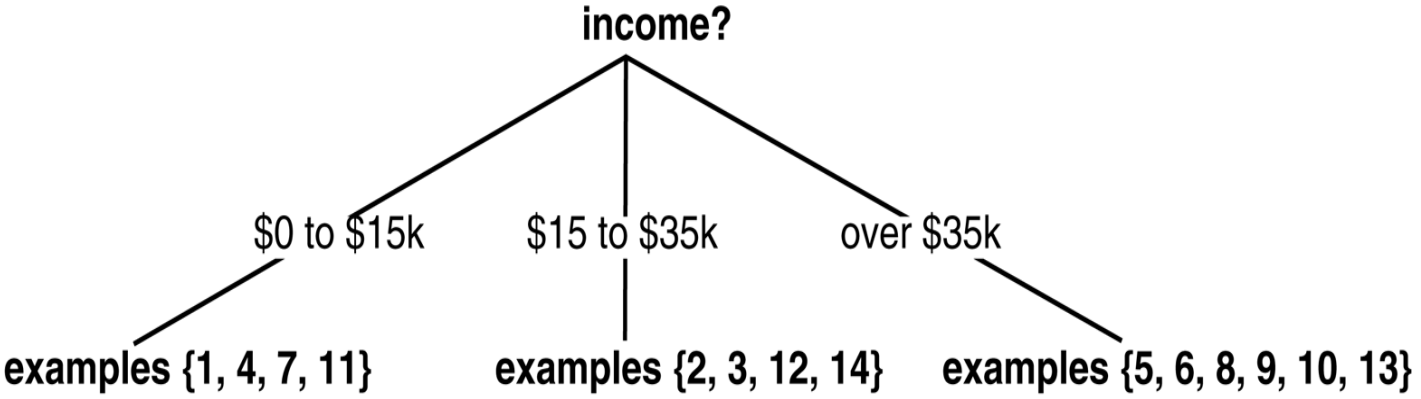
Ví dụ

- Một cây QĐ đơn giản hơn.

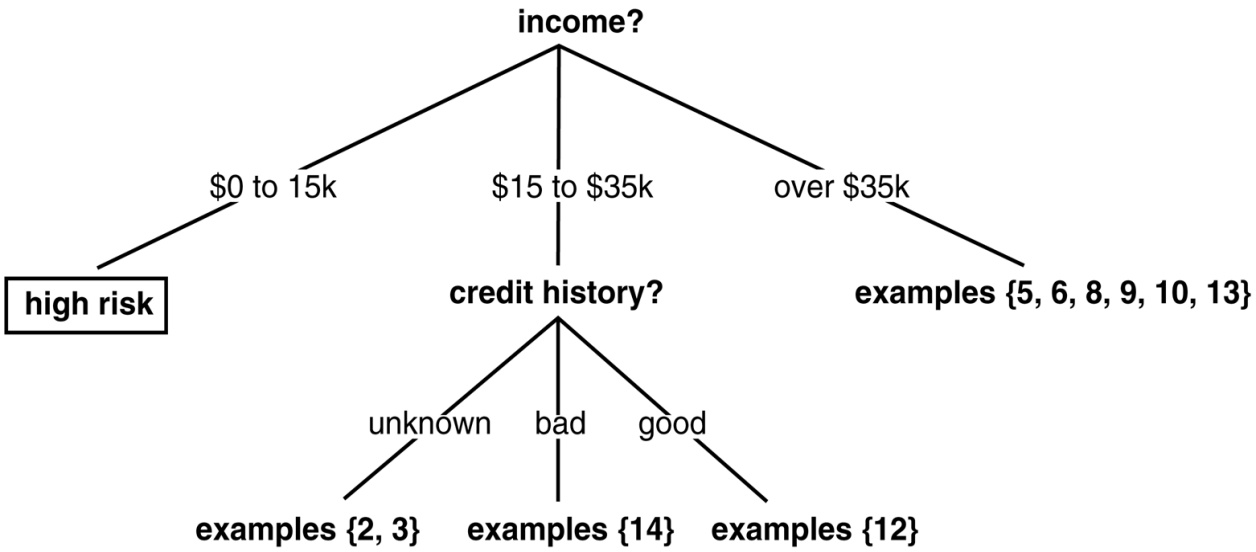


Ví dụ

- Một cây QĐ đang xây dựng.



Một cây



Overfitting trong DT

- **Cây tạo được có thể overfit**
  - Quá nhiều nhánh
  - Độ chính xác kém cho những mẫuchưa biết
- **Lý do overfit**
  - Dữ liệu nhiễu và tách rời khỏi nhóm
  - Dữ liệu huấn luyện quá ít
  - Các giá trị tối đa cục bộ trong tìm kiếm tham lam (greedy search)
- **Cách nào để tránh overfitting**
  - **Rút gọn trước:** ngừng sớm
  - **Rút gọn sau:** loại bỏ bớt các nhánhsau khi xây xong toàn bộ cây

Câu hỏi

- Trình bày và phân tích thuật toán ID3 trong xây dựng mô hình dự đoán.
- Đánh giá, so sánh thuật toán ID3 và C4.5 trong xây dựng cây quyết định.
- Một số ứng dụng của cây quyết định

Bài tập 1

- Cho tập dữ liệu huấn luyện về bài toán dự đoán lựa chọn phương tiện giao thông như sau:

Gender	Car ownership	Travel Cost	Income Level	Transportation mode
Male	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Female	1	Cheap	Medium	Train
Female	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Male	0	Standard	Medium	Train
Female	1	Standard	Medium	Train
Female	1	Expensive	High	Car
Male	2	Expensive	Medium	Car
Female	2	Expensive	High	Car

- Phân tích và xây dựng cây quyết định;

Bài tập 1

- Xác định tập luật dự đoán dựa vào cây quyết định đã xây dựng;
- Sử dụng tập luật đã có để dự đoán người dùng sẽ lựa chọn phương tiện nào

Gender	Car ownership	Travel Cost	Income Level	Transportation Mode
Male	1	Standard	High	?
Male	0	Cheap	Medium	?
Female	1	Cheap	High	?
Male	1	Cheap	Medium	?
Female	0	Cheap	Low	?
Male	0	Expensive	Low	?

Bài tập 2

- Cho tập dữ liệu huấn luyện

#	Outlook	Temperature	Humidity	Wind	Target
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

- Xây dựng cây quyết định để dự đoán việc có "chơi thể thao" hay không dựa trên độ đo Entropy.
- Xây dựng tập luật dự đoán dựa vào câyquyết định đã xây dựng.

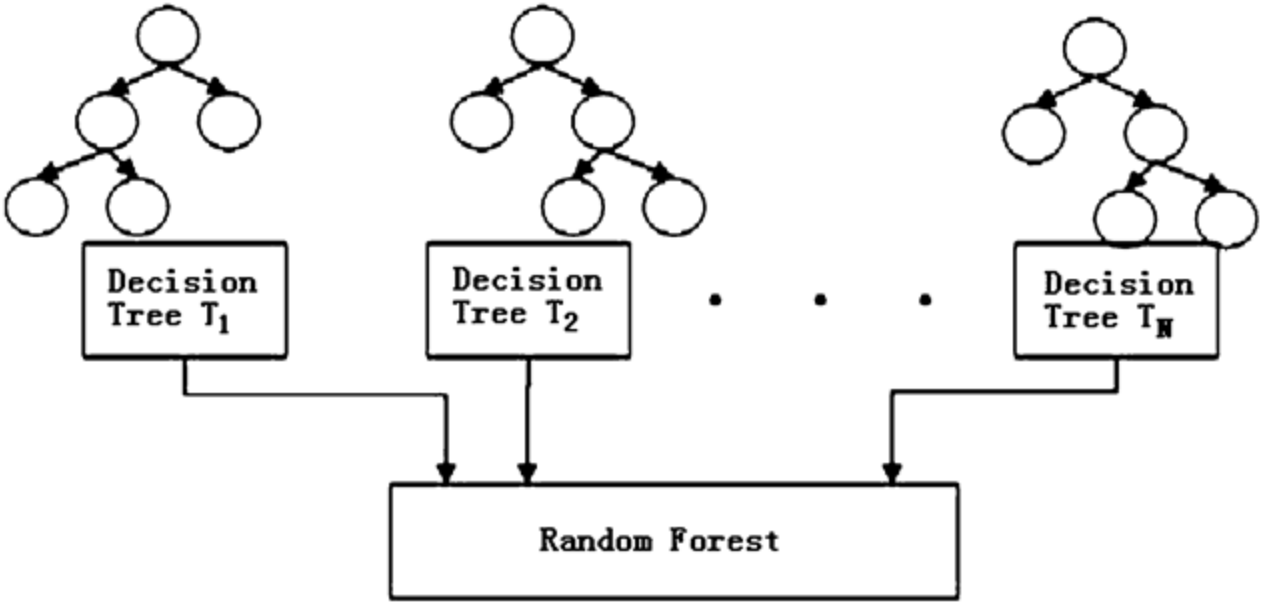
Bài tập 2

Dự đoán giá trị của thuộc tính *Target*

#	Outlook	Temperature	Humidity	Wind	Target
1	Sunny	Mild	Normal	Strong	?
2	Rain	Cool	High	Strong	?
3	Rain	Cool	Normal	Weak	?
4	Rain	Mild	Normal	Strong	?
5	Overcast	Cool	High	Strong	?
6	Sunny	Mild	High	Weak	?
7	Overcast	Cool	Normal	Strong	?
8	Rain	Mild	High	Weak	?
9	Sunny	Cool	Normal	Strong	?
10	Sunny	Mild	High	Strong	?

5. Kỹ thuật Random Forest

Kỹ thuật Rừng ngẫu nhiên được xây dựng trên cơ sở cấu trúc cây quyết định



Random Forest

- RF là kỹ thuật còn khá mới (<15 năm tuổi)
- Đã tạo ra 1 cuộc cách mạng trong Machine Learning:
- RF phức tạp hơn k-nearest neighbors, nhưng lại nhanh hơn rất nhiều khi tính toán với máy tính.
- RF có độ chính xác cao hơn rất nhiều lần so với k-nearest neighbors.
- RF trong chuỗi thuật toán “decision tree”.

Xây dựng mô hình RF

B1) Lựa chọn ngẫu nhiên k thuộc tính từ tập m thuộc tính của dữ liệu, với *k* nhỏ hơn rất nhiều so với *m*.

B2) Từ k thuộc tính, chọn thuộc tính A có khả năng phân chia tốt nhất.

B3) Phát triển cây bằng cách thêm các nhánh tương ứng với từng giá trị của thuộc tính A đã chọn;

B4) Nếu số nút lá trên cây đạt số lượng xác định trước thì qua bước 5,

Ngược lại thì quay lại bước 2

B5) Lặp lại việc xây dựng cây mới từ bước 1 đến bước 4 cho đến khi tạo được rừng với số lượng cây *n* xác định trước.

RF trong phân lớp dữ liệu

1) Từ mẫu dữ liệu đầu vào, sử dụng các thuộc tính đánh giá và các luật đã xây dựng được từ các cây đã xây dựng để đoán nhận lớp. Các lớp đoán nhận được từ các cây được lưu trữ lại.

2) Tính toán mức độ "bình chọn" của mỗi lớp được đoán nhận.

3) Lớp nào có mức độ "bình chọn" cao nhất sẽ là giá trị đoán nhận bởi rừng ngẫu nhiên.

Một số điểm mạnh của RF

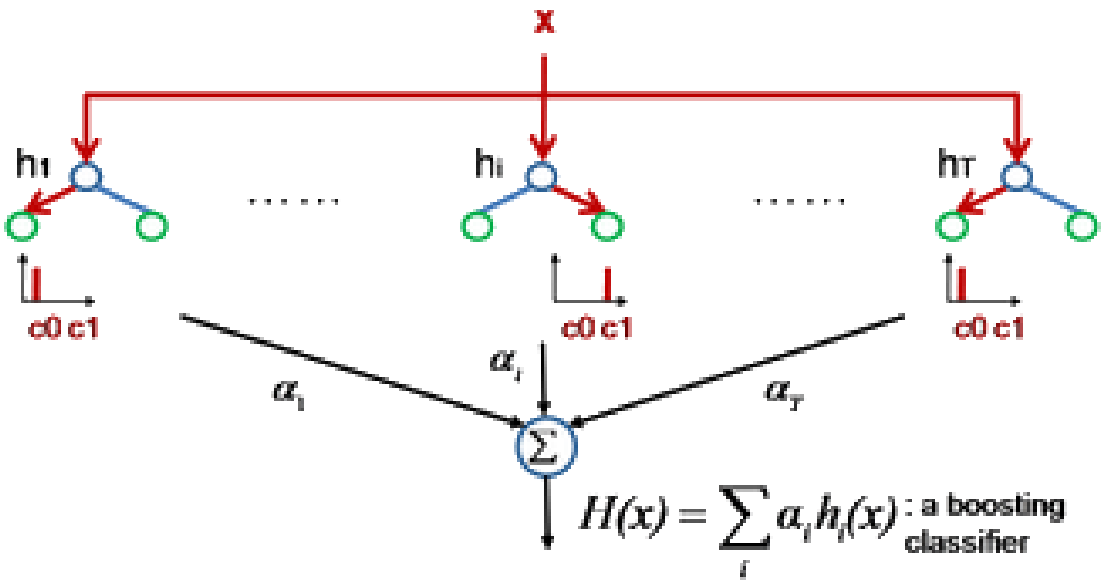
- Rừng ngẫu nhiên giải quyết được cả 2 dạng bài toán phân lớp và quy hồi.
- Rừng ngẫu nhiên rất mạnh trong việc xử lý các tập dữ liệu với số chiều lớn.
- Làm một phương pháp hiệu quả trong việc ước lượng dữ liệu thiếu giá trị thuộc tính và duy trì được độ chính xác khi có một tỷ lệ lớn dữ liệu bị thiếu giá trị.
- Có nhiều phương pháp cho bài toán cân bằng lỗi trong tập dữ liệu khi các lớp không cân bằng.
- Có khă năng mở rộng cho dữ liệu không gán nhãn trước trong việc giải quyết bài toán phân lớp không giám sát và phát hiện ngoại lai.

Một số giới hạn của RF

- RF giải quyết tốt trong bài toán phân loại nhưng chưa tốt trong quy hồi vì nó không đưa ra các các đoán nhận giá trị liên tục. Trong trường hợp của quy hồi, RF không đoán nhận vượt ra ngoài phạm vi dữ liệu huấn luyện, do vậy nó có thể bị over-fit, đặc biệt là nhiễu.
- RF hoạt động như cách tiếp cận blackbox trong mô hình thống kê, khó để điều khiển hoạt động của mô hình của RF.

6. Kỹ thuật Boosting

Phân loại Boosting dựa trên nguyên lý kết hợp nhiều phân loại yếu (weak classifiers) thành phân loại mạnh (Strong classifier)



Khái niệm

- Boosting là thuật toán học quần thể (Ensemble) bằng cách xây dựng nhiều thuật toán học cùng lúc và kết hợp chúng lại.
- Mục đích là để có một cụm hoặc một nhóm các *weak learner* sau đó kết hợp chúng lại để tạo ra một *strong learner* duy nhất.
- **Sự khác nhau giữa strong và weak leaner?**
  - Weak learner phân loại với độ chính xác hầu như không cao. Một ví dụ weak learner là cây quyết định một cấp.
  - Strong leaner có độ chính xác cao hơn nhiều

Phân loại kỹ thuật Boosting

- Hiện nay có nhiều biến thể của Boosting
  - LPBoost (**Linear Programming Boosting)**
  - TotalBoost
  - BrownBoost
    - BrownBoost is a boosting algorithm that may be robust to noisy datasets
  - Xgboost
    - Open-source library, provides the gradient boosting framework
  - MadaBoost
  - LogitBoost
  - AdapBoost (Adaptive Boosting)

Thuật toán Boosting

- Boosting (Schapire 1989)

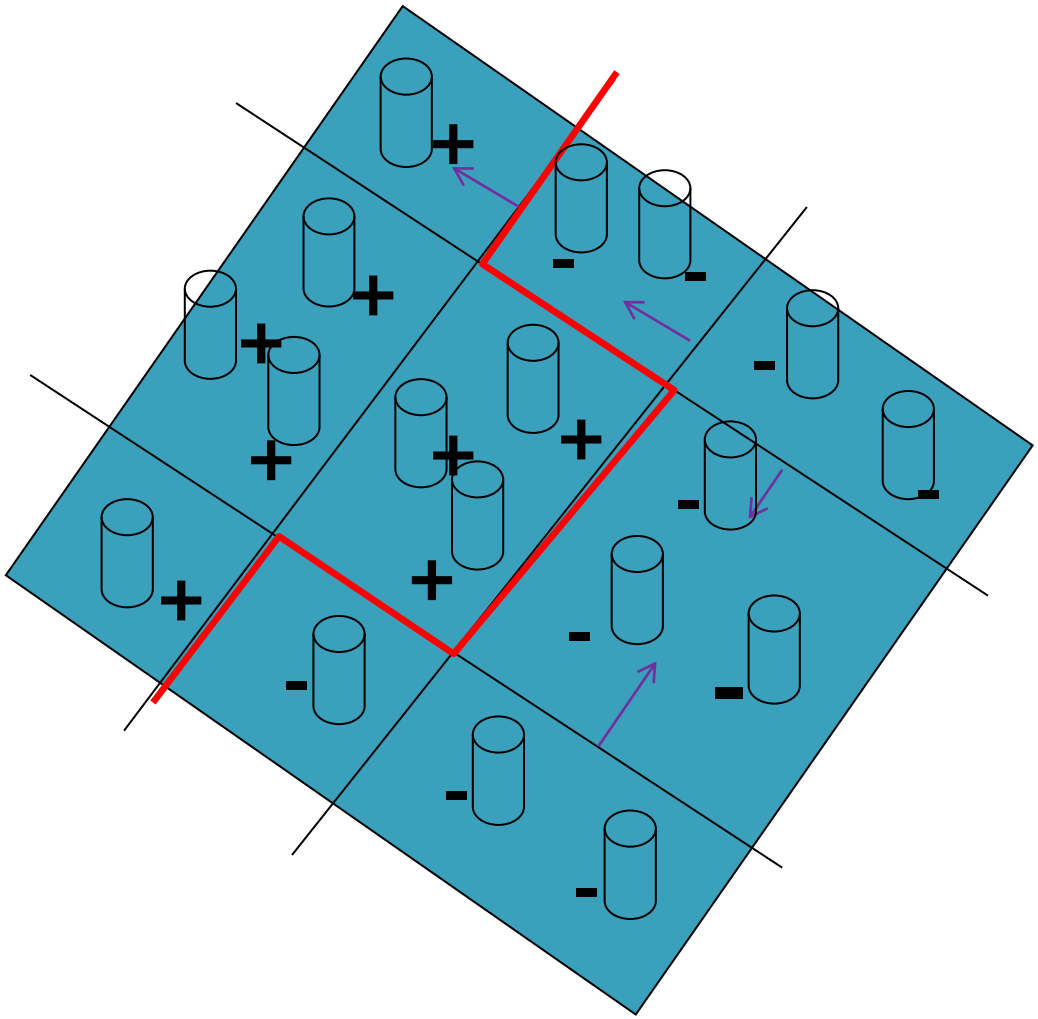
  - Randomly select $n_1 < n$ samples from D without replacement to obtain D_1
    - Train weak learner C_1
  - Select $n_2 < n$ samples from D with half of the samples misclassified by C_1 to obtain D_2
    - Train weak learner C_2
  - Select all samples from D that C_1 and C_2 disagree on
    - Train weak learner C_3
  - Final classifier is vote of weak learners

AdaBoost

- AdaBoost là một thuật toán boosting, theo kiểu học thích ứng.
  - Một classifier nhận vào một tập dữ liệu để học và cố gắng dự đoán hay phân lớp mẫu dữ liệu mới thuộc về phân lớp nào.
- Thuật toán đơn giản và dễ dàng cài đặt.
- Có tốc độ học rất nhanh.
- Các weak learner đơn giản hơn nhiều các strong learner, nhờ vậy thuật toán chạy nhanh hơn.
- AdaBoost là phương pháp có khả năng điều chỉnh các classifier rất tinh tế. Vì mỗi lần lặp AdaBoost lại tinh chỉnh lại các trọng số cho các learner tốt nhất.
- Là thuật toán linh hoạt và đa năng. AdaBoost có thể kết hợp với bất kỳ thuật toán học máy nào và nó có thể làm việc với một lượng lớn dữ liệu khác nhau.

AdaBoost

- AdaBoost is an algorithm for constructing a “strong classifier” by combine many “weak classifier”.



Thuật toán AdaBoost

- AdaBoost is an algorithm for constructing a "strong classifier" by combine many "weak classifier".

$$C(x) = \begin{cases} 1 & if \sum_{t=1}^{T} \alpha_t h_t(x, f, p, \theta) \geq 0 \\ -1 & otherwise \end{cases}$$

C is a strong classifier.

T is a number weak classifier.

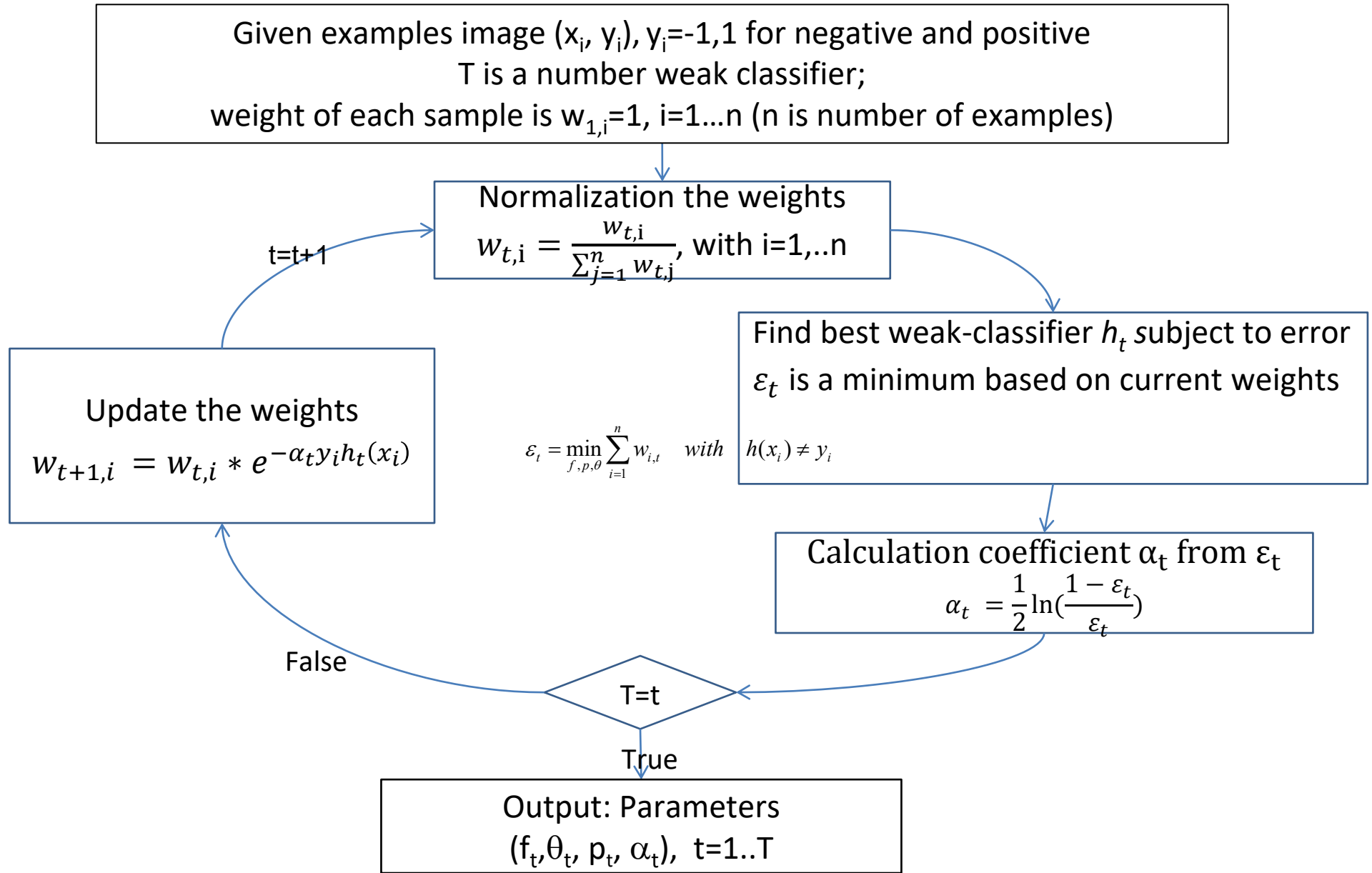
α_t is a gain coefficient of weak classifier h_t.

h_t is a weak classifier

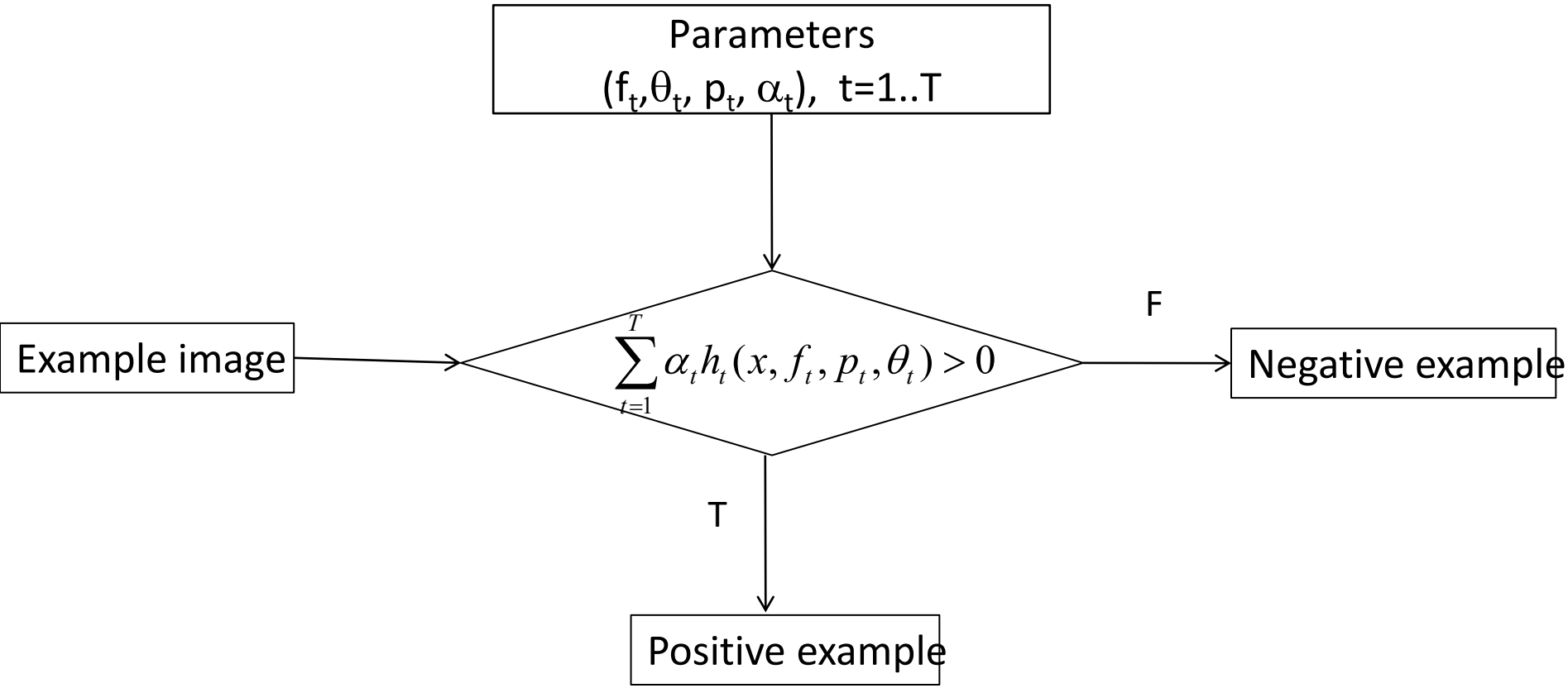
$$h(x, f, p, \theta) = \begin{cases} 1 & if \quad p \cdot f(x) < p \cdot \theta \\ -1 & otherwise \end{cases}$$

θ is threshold, $p \in \{-1, 1\}$ is polarity, f is the order of feature of sample image; x is example image.

Training stage



Classification stage



AdaBoost

- Update the weights:

$$w_{i,t+1} = w_{i,t} * e^{-\alpha_t y_i h_t(x_i)}$$

  - AdaBoost tries to focus on wrongly classified samples.
  - Increase weight of wrongly classified samples.

    $y_i h_t(x_i) < 0$
  - Decrease weight of correctly classified samples.

    $y_i h_t(x_i) > 0$
  - All information about previously selected features is captured in w_t

AdaBoost

- **Minimum error**
  - After each iteration, the classifier is:

$$C_t(x_i) = \alpha_1 h_1(x_i) + \alpha_2 h_2(x_i) + \ldots + \alpha_{t-1} h_{t-1}(x_i) + \alpha_t h_t(x_i)$$
$$= C_{(t-1)}(x_i) + \alpha_t h_t(x_i)$$

  - Error that occur on each example x_i is defined as exponential loss

$$E_t(x_i) = e^{-y_i C_t(x_i)}$$

Total error on training set is:

$$E_t = \sum_{i=1}^{N} e^{-y_i C_t(x_i)} = \sum_{i=1}^{N} e^{-y_i [C_{t-1}(x_i) + \alpha_t h_t(x_i)]} = \sum_{i=1}^{N} e^{-y_i C_{t-1}(x_i)} e^{-y_i \alpha_t h_t(x_i)}$$

$$E_t = \sum_{i=1}^{N} w_{i,t} e^{-y_i \alpha_t h_t(x_i)}$$

AdaBoost

- **Choosing α_t:**

  – $E_t = \sum_{i=1}^{n} w_{i,t} * e^{(-\alpha_t y_i h_t(x_i))}$

  Attempt minimize E_t

  – $\frac{dE_t}{d\alpha} = -\sum_{i=1}^{n} w_{i,t} y_i h(xi) * e^{(-\alpha_t y_i h_t(x_i))} = 0$

  $\Leftrightarrow -\sum_{\substack{i=1 \\ y_i = h(xi)}}^{n} w_{i,t} * e^{-\alpha_t} + \sum_{\substack{i=1 \\ y_i \neq h(xi)}}^{n} w_{i,t} * e^{\alpha_t} = 0$

  $\Leftrightarrow -e^{-\alpha_t}(1 - \varepsilon_t) + e^{\alpha_t}\varepsilon_t = 0$

  $\Leftrightarrow e^{2\alpha_t} = (1 - \varepsilon_t)/\varepsilon_t$

  $\Leftrightarrow \alpha_t = \frac{1}{2} ln(\frac{1-\varepsilon_t}{\varepsilon_t})$

AdaBoost

- Tóm lược

Thuật toán AdaBoost được mô tả gồm các nội dung chính sau:

Đầu vào: Tập dữ liệu huấn luyện $(x_1,y_1),\ldots (x_m,y_m)$ với $x_i \in X$ không gian mẫu, $y_i \in \{-1,+1\}$

Khởi tạo: $D_1(i)=1/m$ với $i=1,\ldots,m$.

For $t=1$ to T // với T là số lượng các weak classifiers

Huấn luyện phân loại yếu với tập giá trị trọng số D_t hiện tại.

Thực hiện các phân loại yếu để đánh giá mức độ "tốt": h_t: X $\rightarrow \{-1,+1\}$

Chọn phân loại yếu h_t tốt nhất với trọng số lỗi là theo công thức

$$\varepsilon_t = \Pr_{i\sim Dt}[h_t(x_i) \neq y_i]$$

Xác định hệ số $\alpha = \frac{1}{2}\ln(\frac{1-\varepsilon_t}{\varepsilon_t})$

Cập nhật lại trọng số D với mọi $i=1,\ldots,m$

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

với Z_t là hệ số chuẩn hóa.

Đầu ra là bộ phân loại mạnh dựa trên kết hợp các phân loại yếu h_t cùng bộ hệ số boosting tương ứng α_t

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

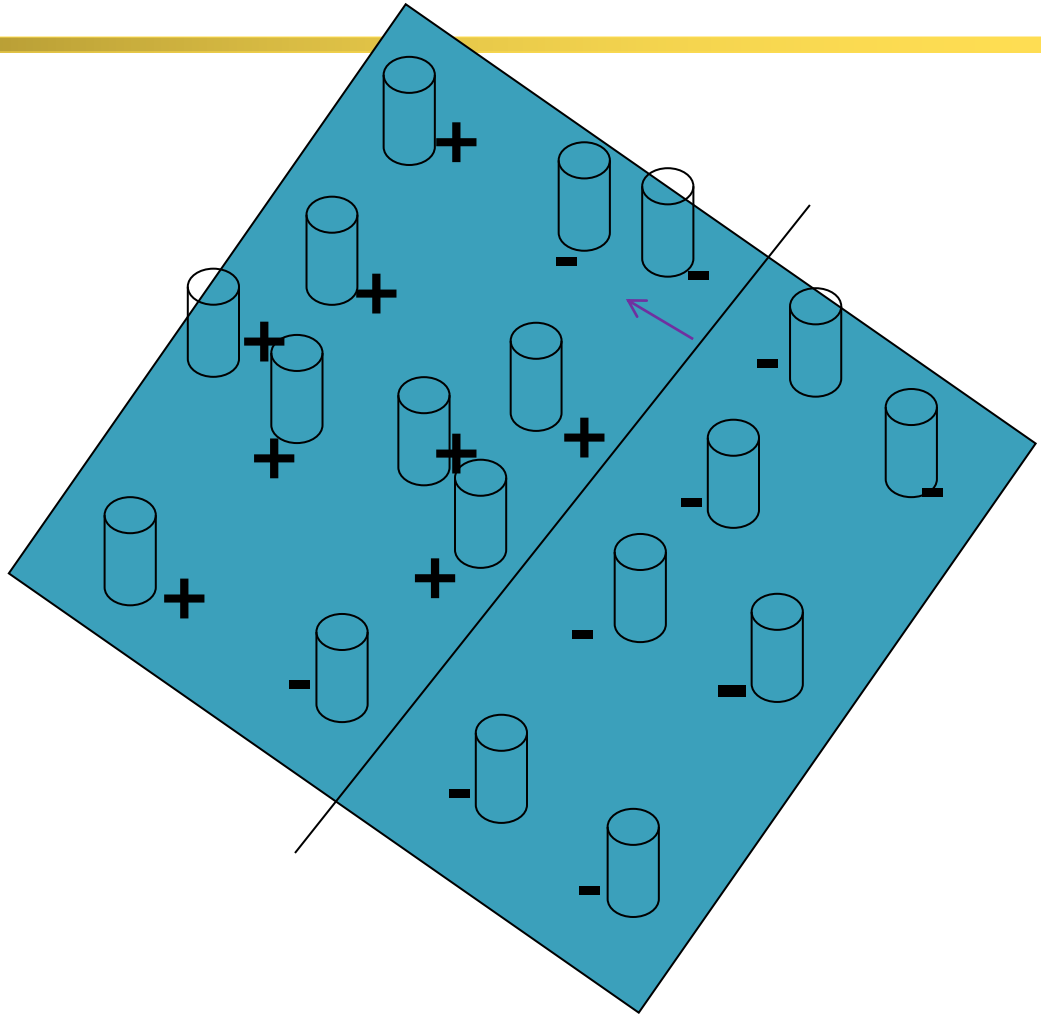
Một số đánh giá cơ bản về AdaBoost:

- AdaBoost là một thuật toán boosting, theo kiểu học thích ứng.
- Một classifier nhận vào một tập dữ liệu để học và cố gắng dự đoán lớp của mẫu dữ liệu mới.
- Thuật toán đơn giản và dễ dàng cài đặt.
- Weak learner phân loại với độ chính xác hầu như không cao. Tuy nhiên khi kết hợp nhiều phân loại yếu thành phân loại mạnh có độ chính xác cao.
- Có tốc độ phân lớp nhanh.
- AdaBoost là phương pháp có khả năng điều chỉnh các phân loại yếu rất tinh tế. Vì mỗi lần lặp AdaBoost lại tinh chỉnh lại các trọng số tập trung cho các mẫu phân loại khó.
- Thuật toán linh hoạt và đa năng. AdaBoost có thể kết hợp với bất kỳ thuật toán học máy nào và có thể làm việc với một lượng lớn dữ liệu khác nhau.

Ví dụ hoạt động AdaBoost

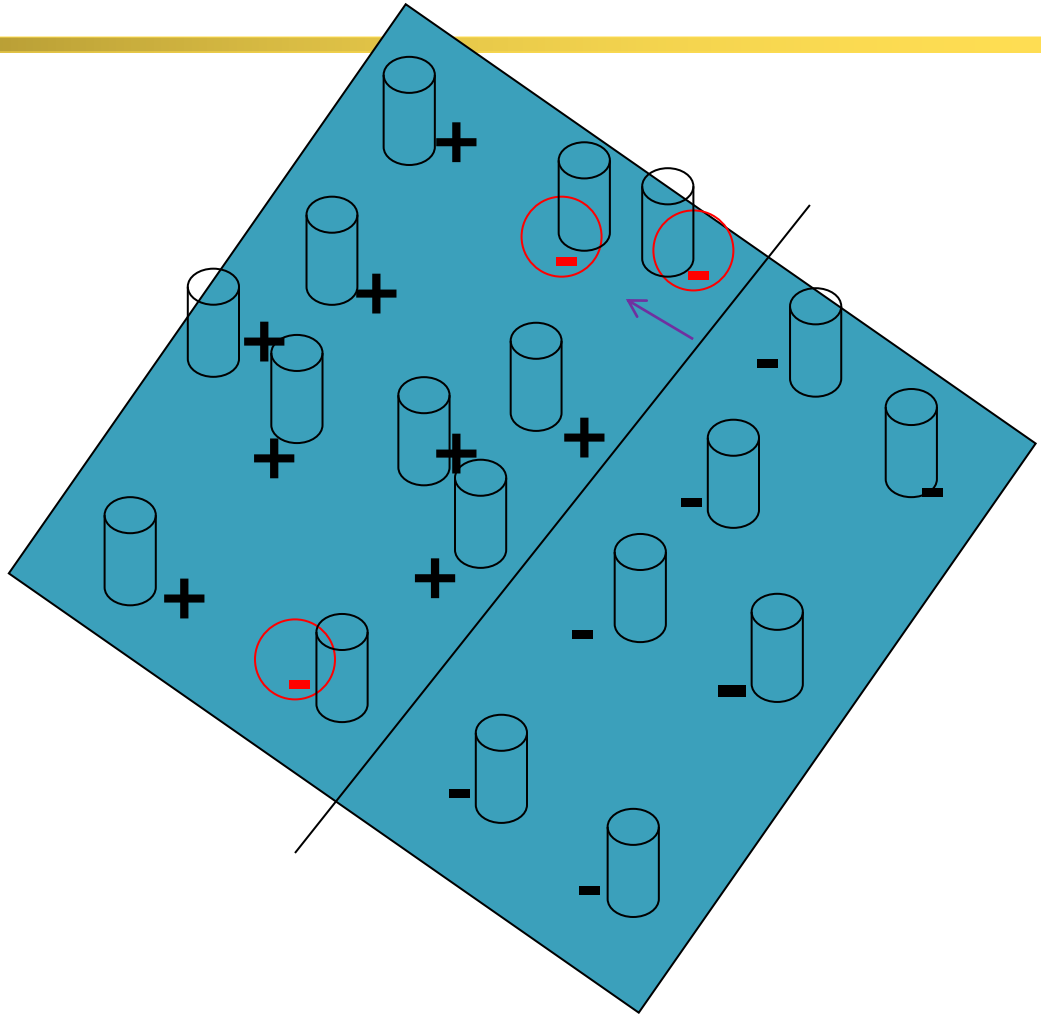
- For example

Choosing a weak classifier



AdaBoost algorithm

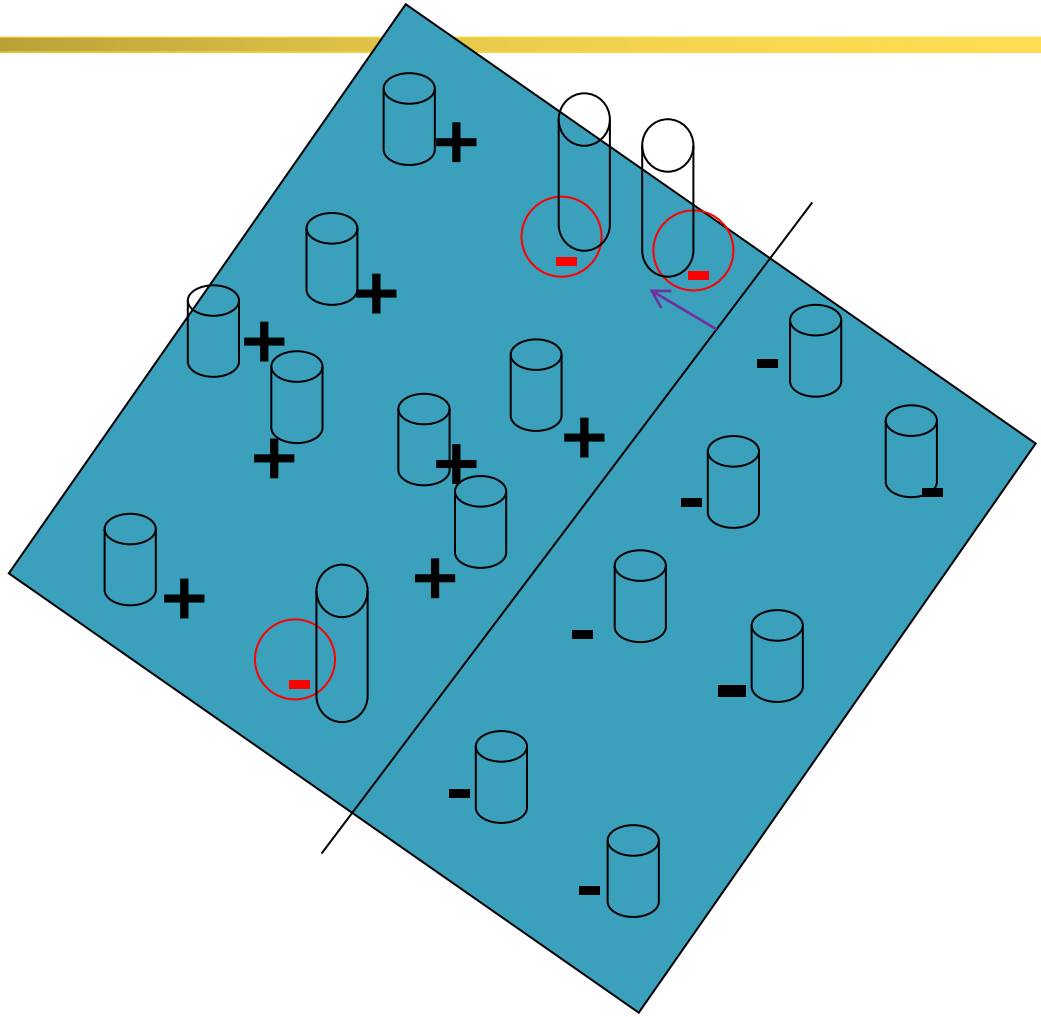
- For example



AdaBoost algorithm

- For example

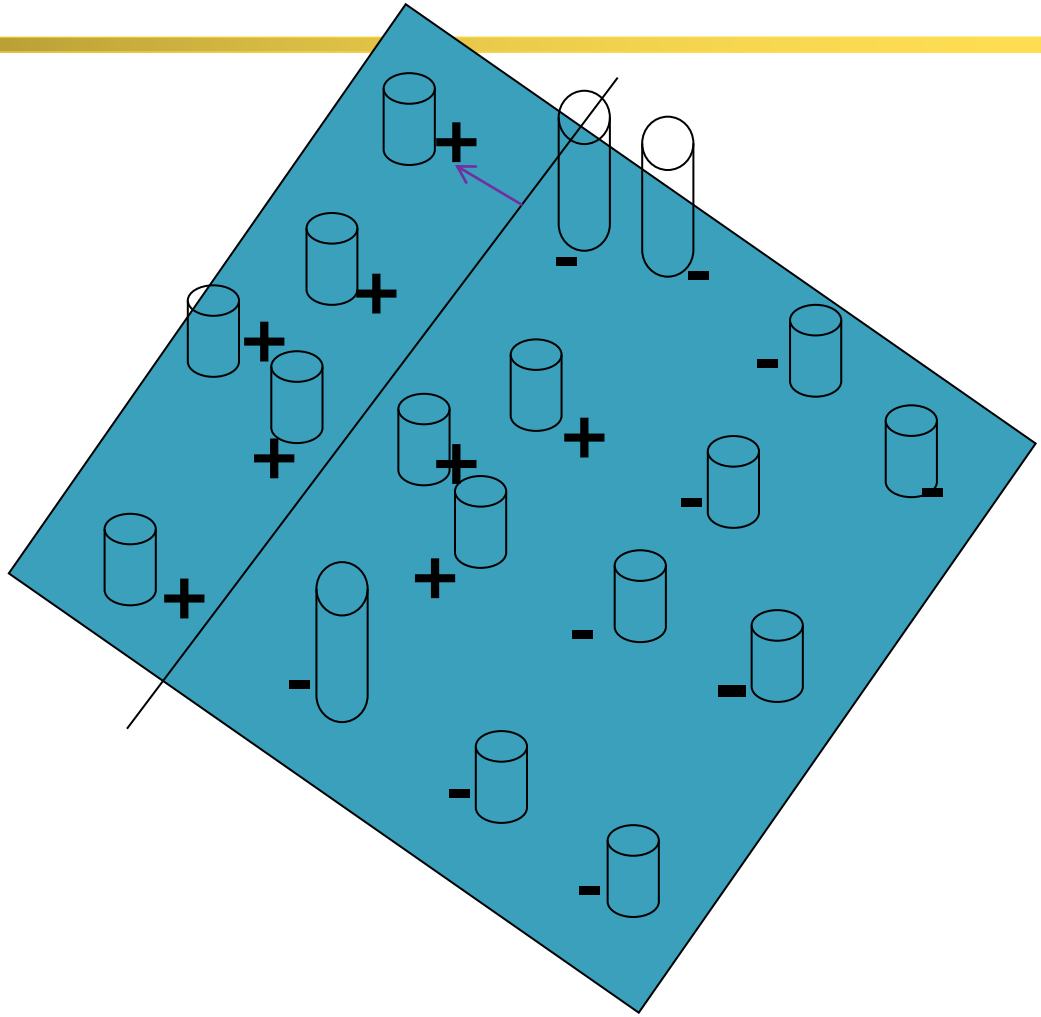
Updating the weights



AdaBoost algorithm

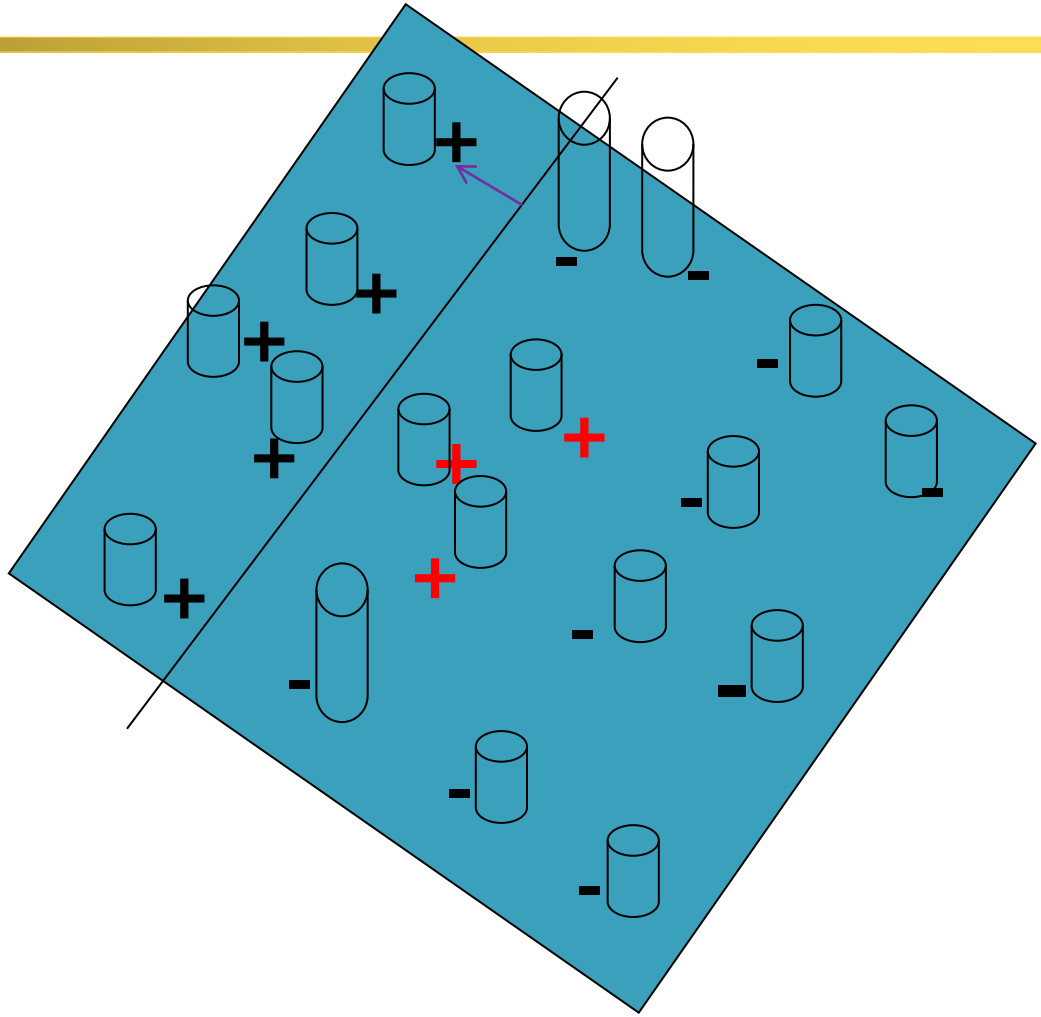
- For example

Choosing a weak classifier



AdaBoost algorithm

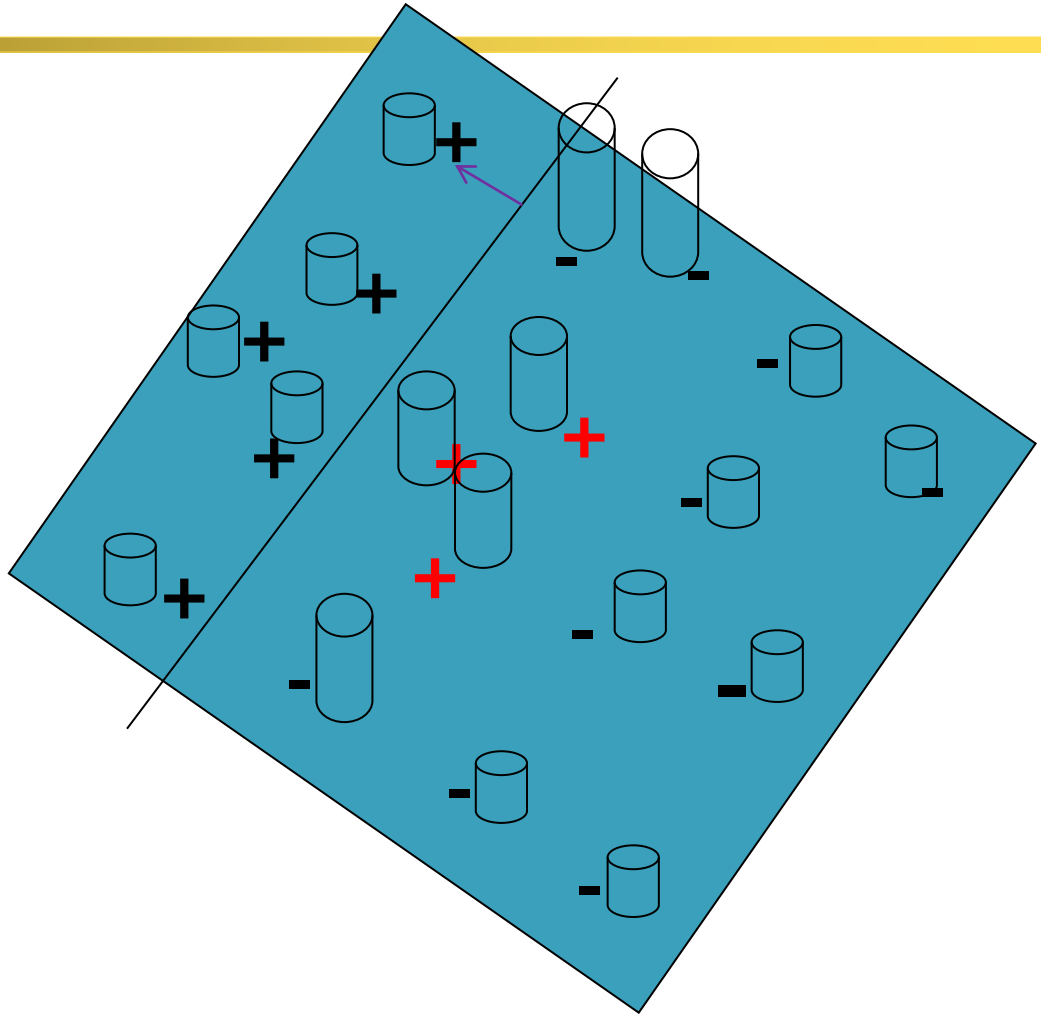
- For example



AdaBoost algorithm

- For example

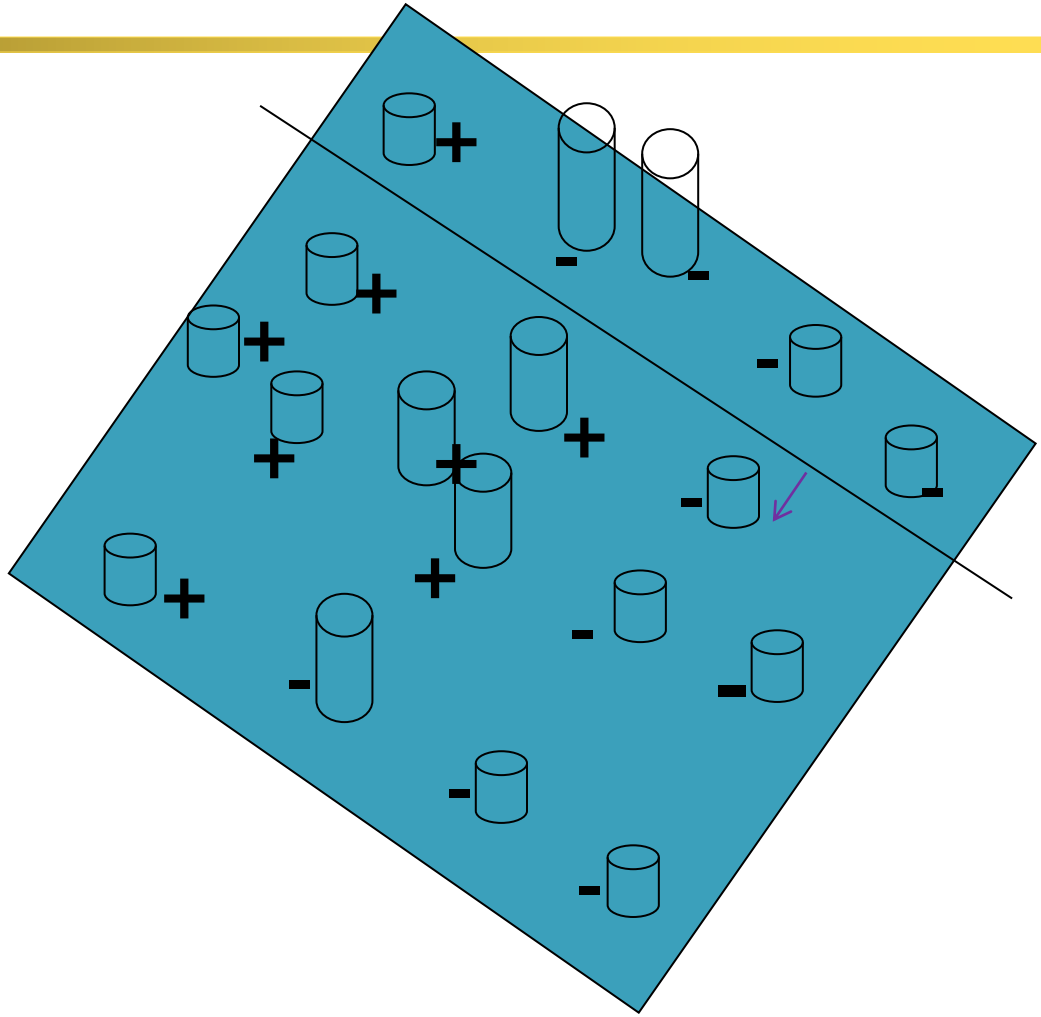
Updating the weights



AdaBoost algorithm

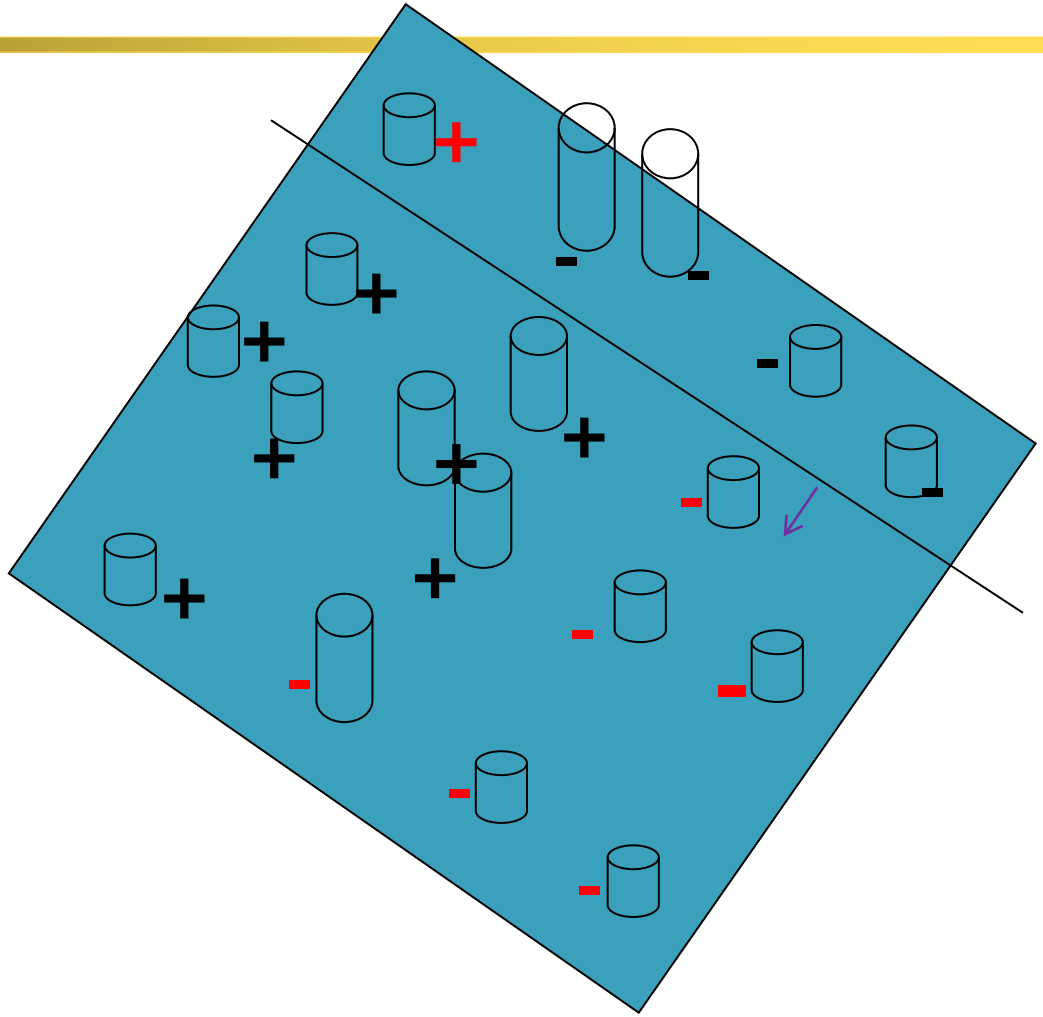
- For example

Choosing a weak classifier



AdaBoost algorithm

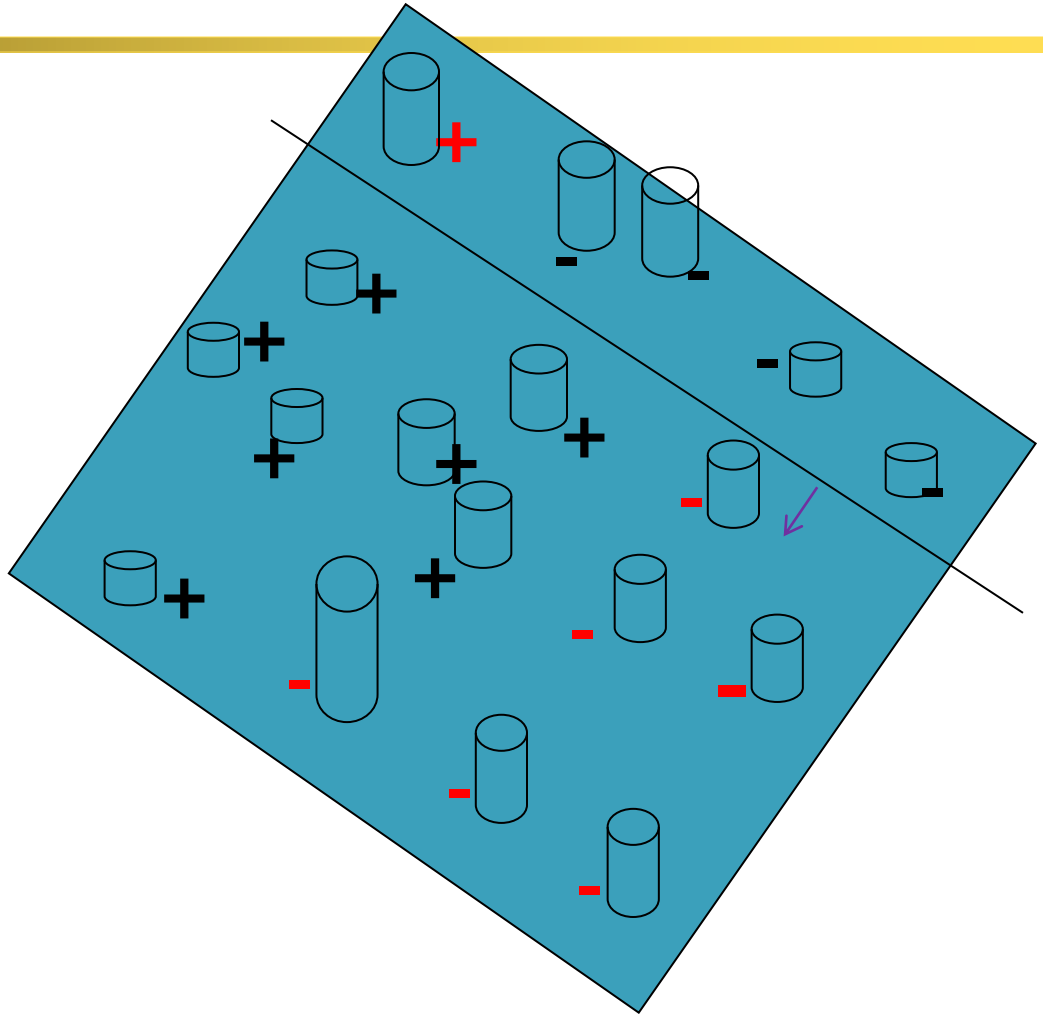
- For example



AdaBoost algorithm

- For example

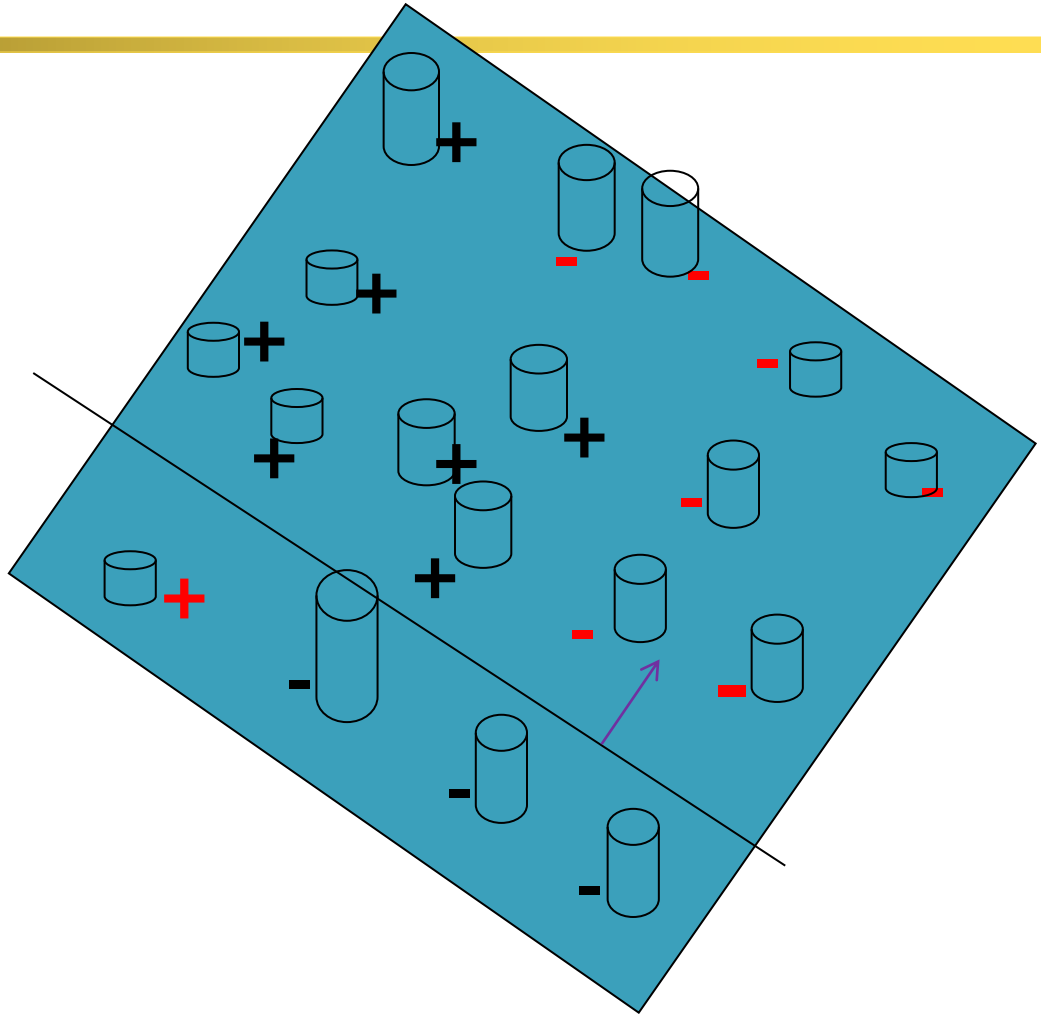
Updating the weights



AdaBoost algorithm

- For example

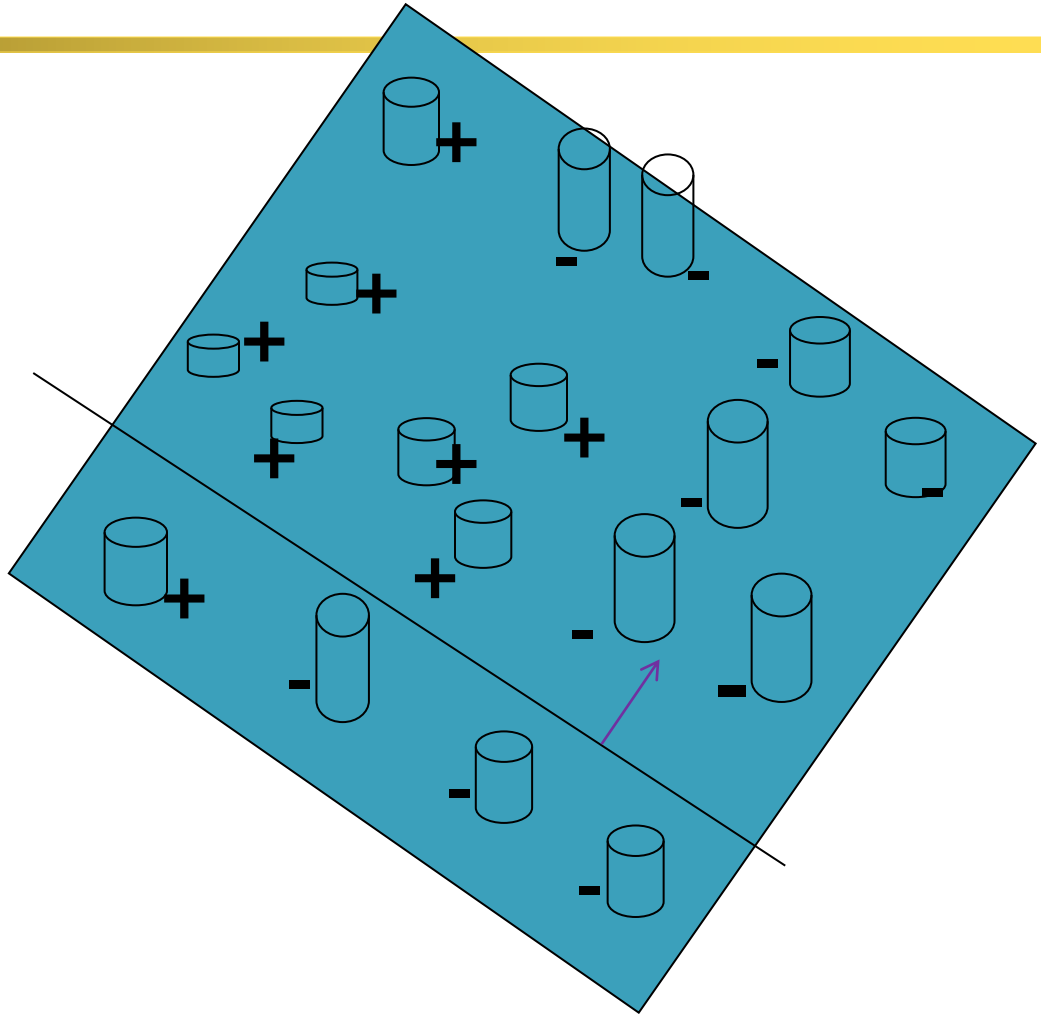
Choosing a weak classifier



AdaBoost algorithm

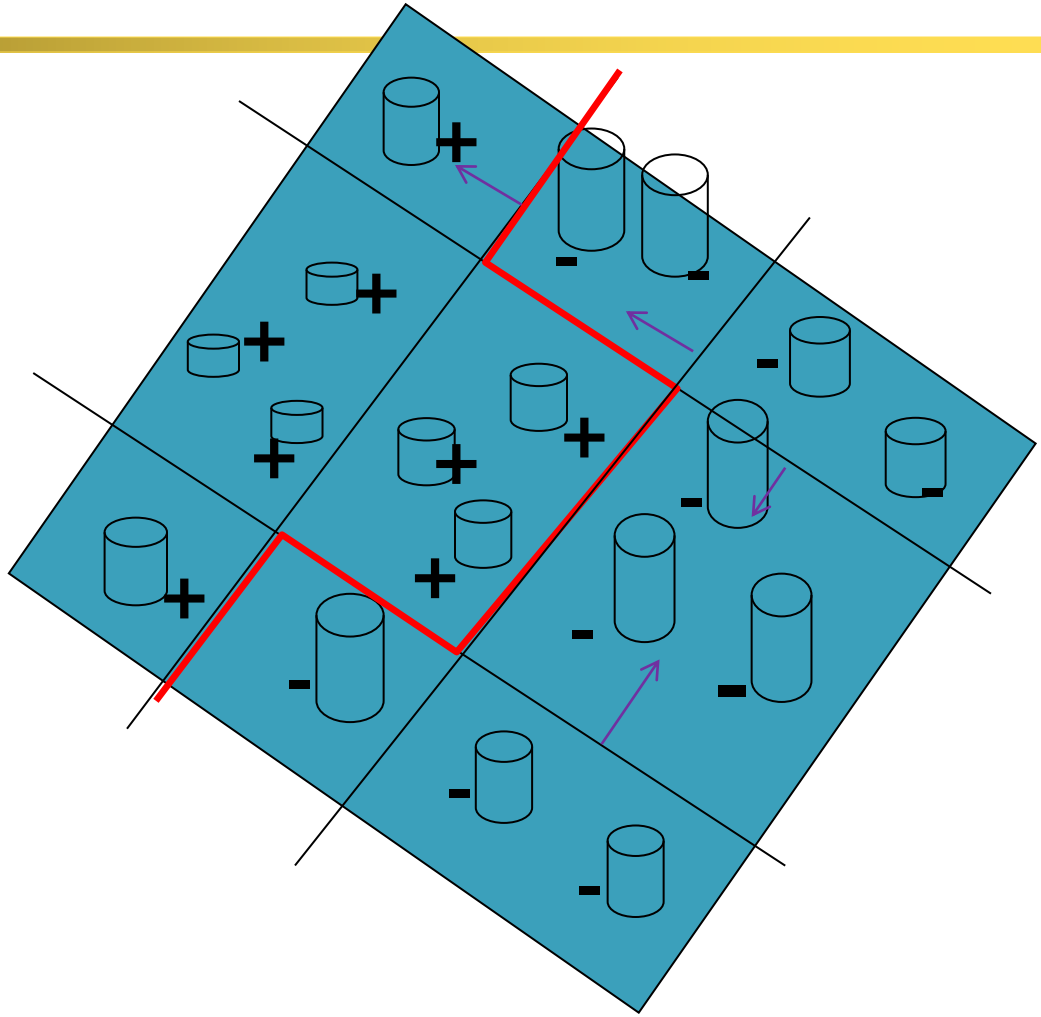
- For example

Updating the weights

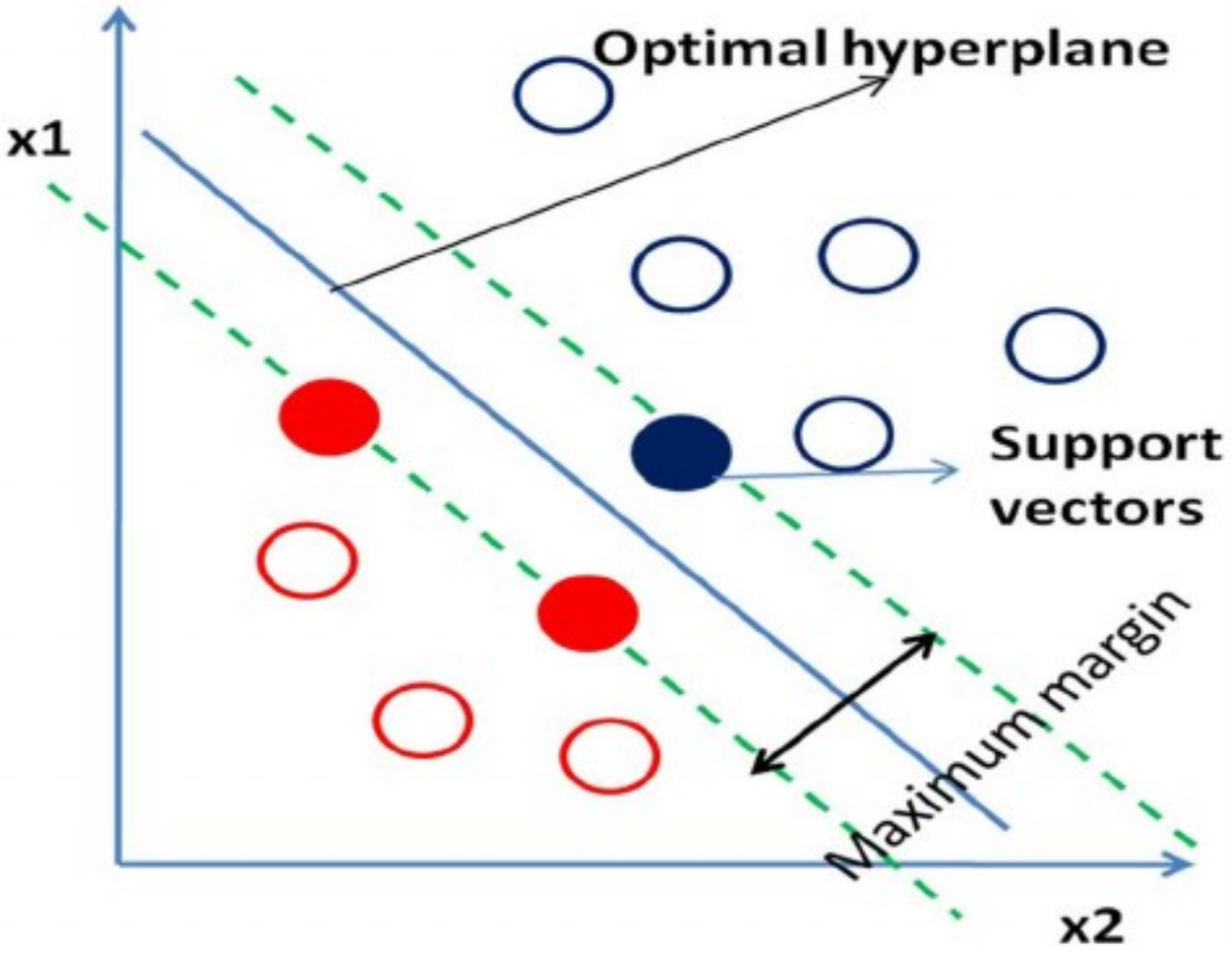


AdaBoost algorithm

- For example



Bộ phân lớp SVM



Thuật toán SVM

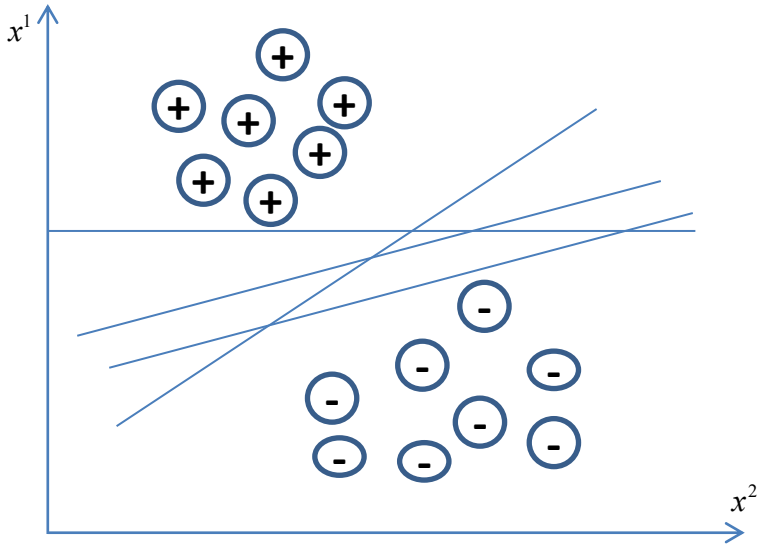
- Thuật toán máy vector hỗ trợ (Support Vector Machine – SVM)
- Được Corters và Vapnik giới thiệu vào năm 1995.
- SVM rất hiệu quả để giải quyết các bài toán với dữ liệu có số chiều lớn (như các vector biểu diễn văn bản).

Thuật toán SVM

- Tập dữ liệu học: D= $\{(X_i, C_i), i=1,...n\}$
  - $C_i \in \{-1,1\}$ xác định dữ liệu dương hay âm
- Tìm một siêu phẳng: $\alpha_{SVM} .d + b$ phân chia dữ liệu thành hai miền.
- Phân lớp một tài liệu mới: xác định dấu của
  - $f(d) = \alpha_{SVM} .d + b$
  - Thuộc lớp dương nếu $f(d) > 0$
  - Thuộc lớp âm nếu $f(d) < 0$

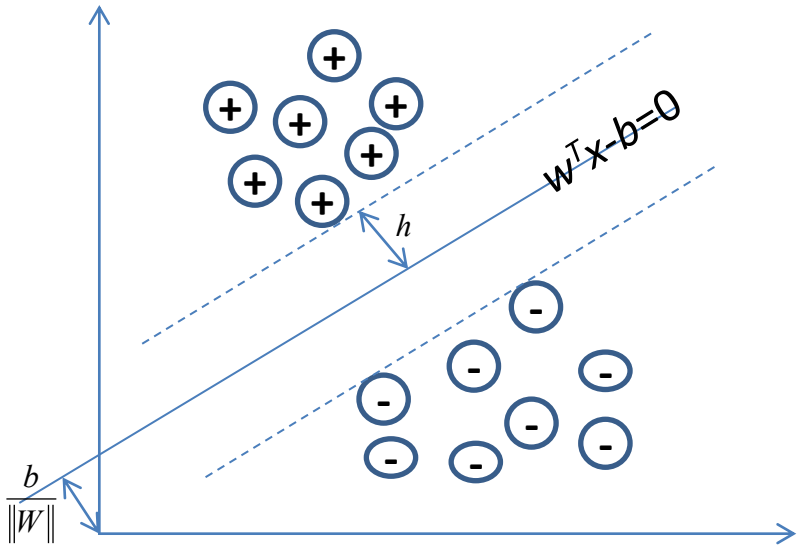
SVM

- Input: $D=\{(x_i,y_i)|\ x\in\Re^p,\ y_i\in\{-1,1\}\}$, i=1,..N, N is a number of examples.



- Objective: How to classify these points by linearly separable function in order to minimum error rate.

Problem statement for SVM



- Find hyper-plane that divides the points having $y_i = 1$ from other having $y_i = -1$ with the maximum-margin (h).

The hyper-plane represent by equation $w^T x-b=0$, w is normal vector of the hyper-plane, $b/||w||$ determines the offset of the hyper-plane from the origin along the w.

Linear classifier

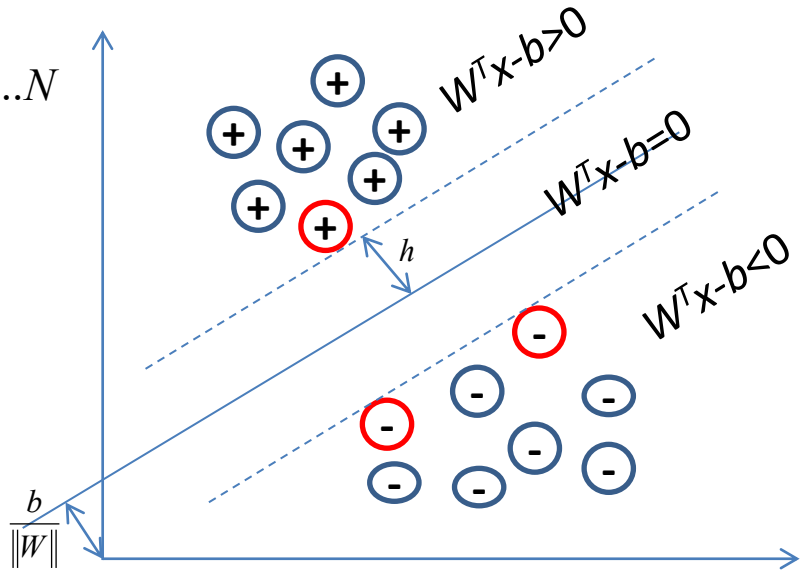
- We have

$$\begin{cases} w^T x_i - b > 0 & with\ y_i = 1 \\ w^T x_i - b < 0 & with\ y_i = -1 \end{cases} \Rightarrow y_i(w^T x_i - b) > 0, \quad i = 1..N$$

- Distance $d((w, b), x_i)$:

$$h_i = y_i \frac{w^T x_i - b}{\|w\|}, \quad i = 1..N$$

$$\Rightarrow h = \min_{i=1..N} h_i = \min_{i=1..N} y_i \frac{w^T x_i - b}{\|w\|}$$



=> Original objective is equivalent

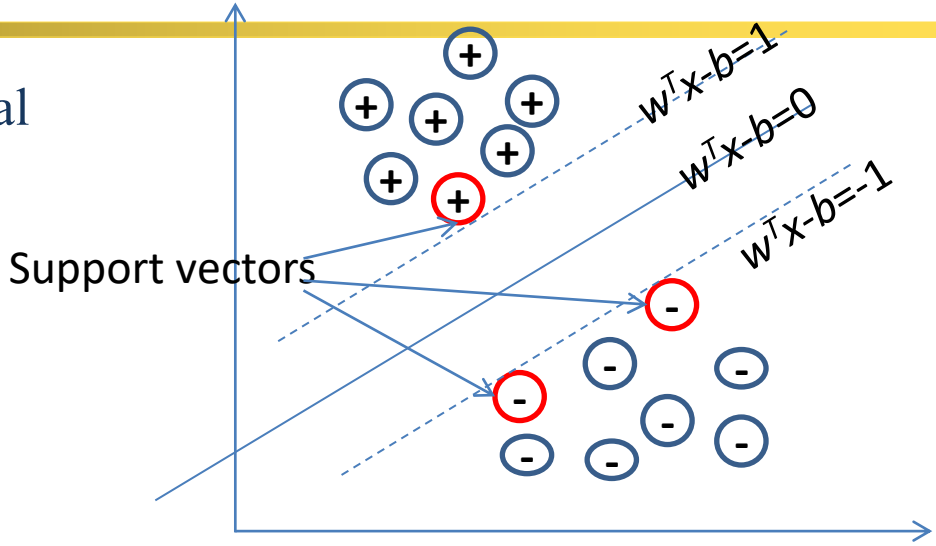
$$\max_{w,b} h \qquad (1)$$

$$s.t. \quad y_i(w^T x_i - b) \geq h\|w\|, \quad \forall i = 1..N$$

Linear classifier

- Karush-Kuhn-Tucker (KTT) conditional

$$\lambda_i[y_i(w^T x_i - b) - 1] = 0, \quad \forall i = 1..n$$



=> 1) If $\lambda_i = 0 \Rightarrow$ x_i not attend to find (w, b)

2) If $\lambda_i <> 0 \Rightarrow y_i(w^T x_i - b) = 1$

$$\Rightarrow \quad b = \begin{cases} w^T x_i - 1 & with \quad y_i = 1 \\ w^T x_i + 1 & with \quad y_i = -1 \end{cases}$$

x_i is support vector

Linear classifier with soft margin

- If the data set cannot linear separable we can use “soft margin”

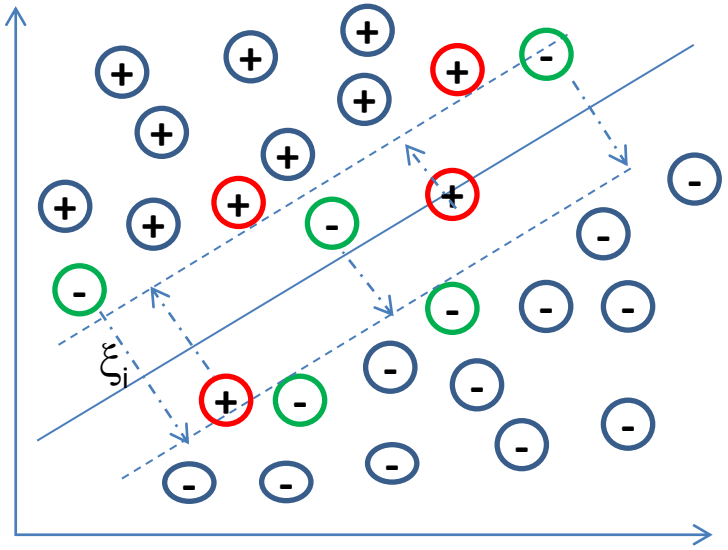
$$\min_{w,b} \quad \frac{1}{2}\|w\|^2 \qquad (3)$$

$$s.t. \quad y_i(w^T x_i - b) \geq 1, \quad \forall i = 1..n$$

Adds some constrains

$$\min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

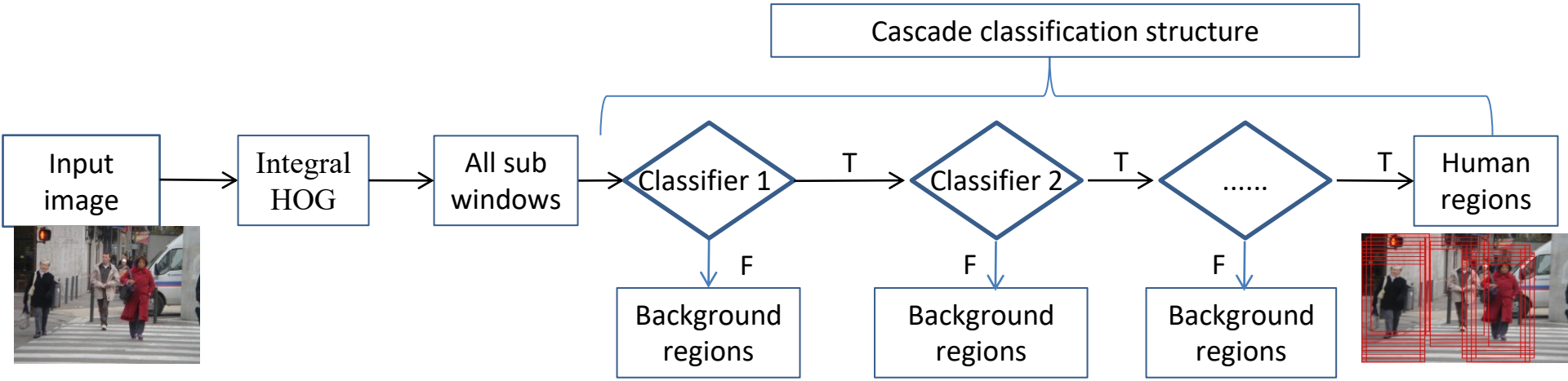
$$s.t. \quad \begin{cases} y_i(w^T x_i - b) \geq 1 - \xi_i, \\ \xi_i \geq 0 \end{cases} \quad \forall i = 1..n$$



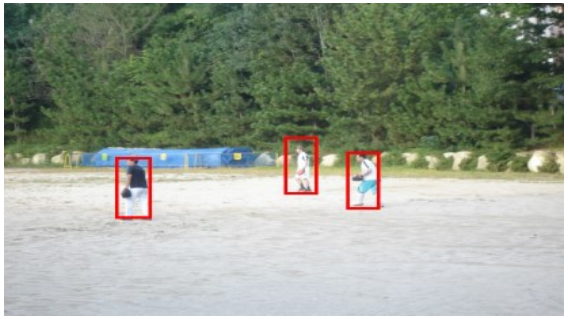
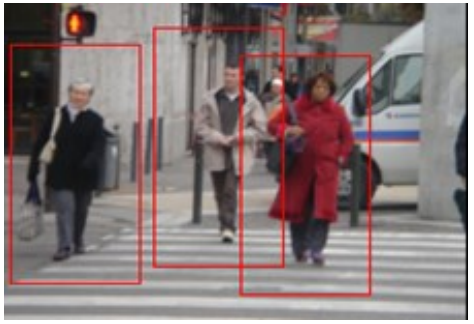
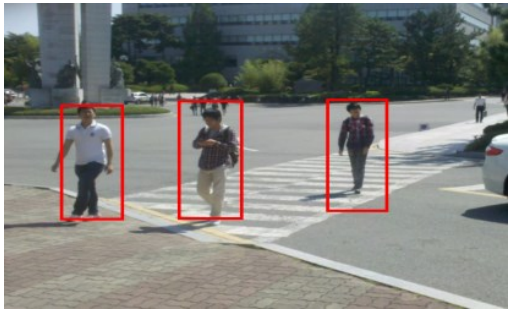
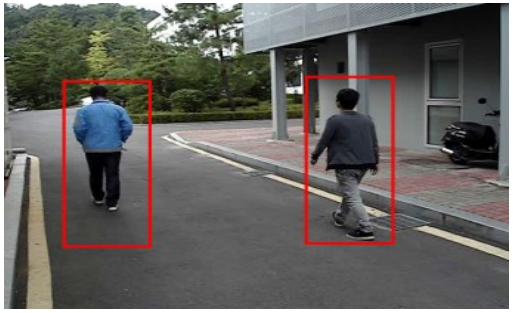
Where ξ_i is a slack variables of example x_i, C is a trade-off parameter.

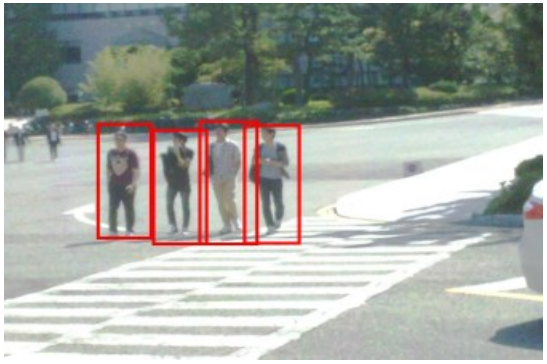
Ví dụ ứng dụng

- Quá trình nhận dạng người dùng boosting



Ví dụ

Ví dụ

- Samples database
  - Used image database on INRIA (1208 person images for training and 2334 person images for testing).
  - Training set:
    - 1208 positive examples (human).
    - 2000 negatives examples (non-human).
  - Testing set:
    - 2334 positives examples (human).
    - 5000 negative examples (non-human).
- Testing and Comparison :
  - AdaBoost and SVM.
  - Various HOG descriptor.

Q&A