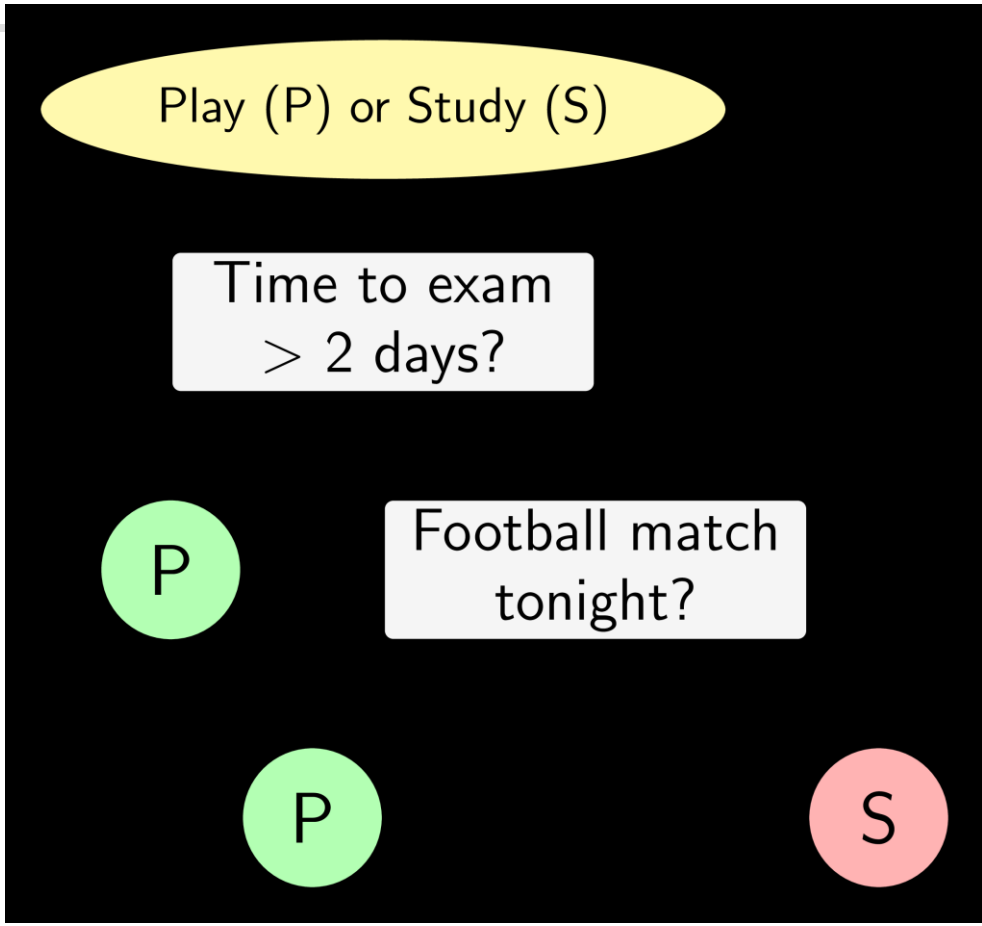
CSE: **Faculty of Computer Science and Engineering**
Thuyloi University

Cây quyết định (Decision tree)

TS. Nguyễn Thị Kim Ngân

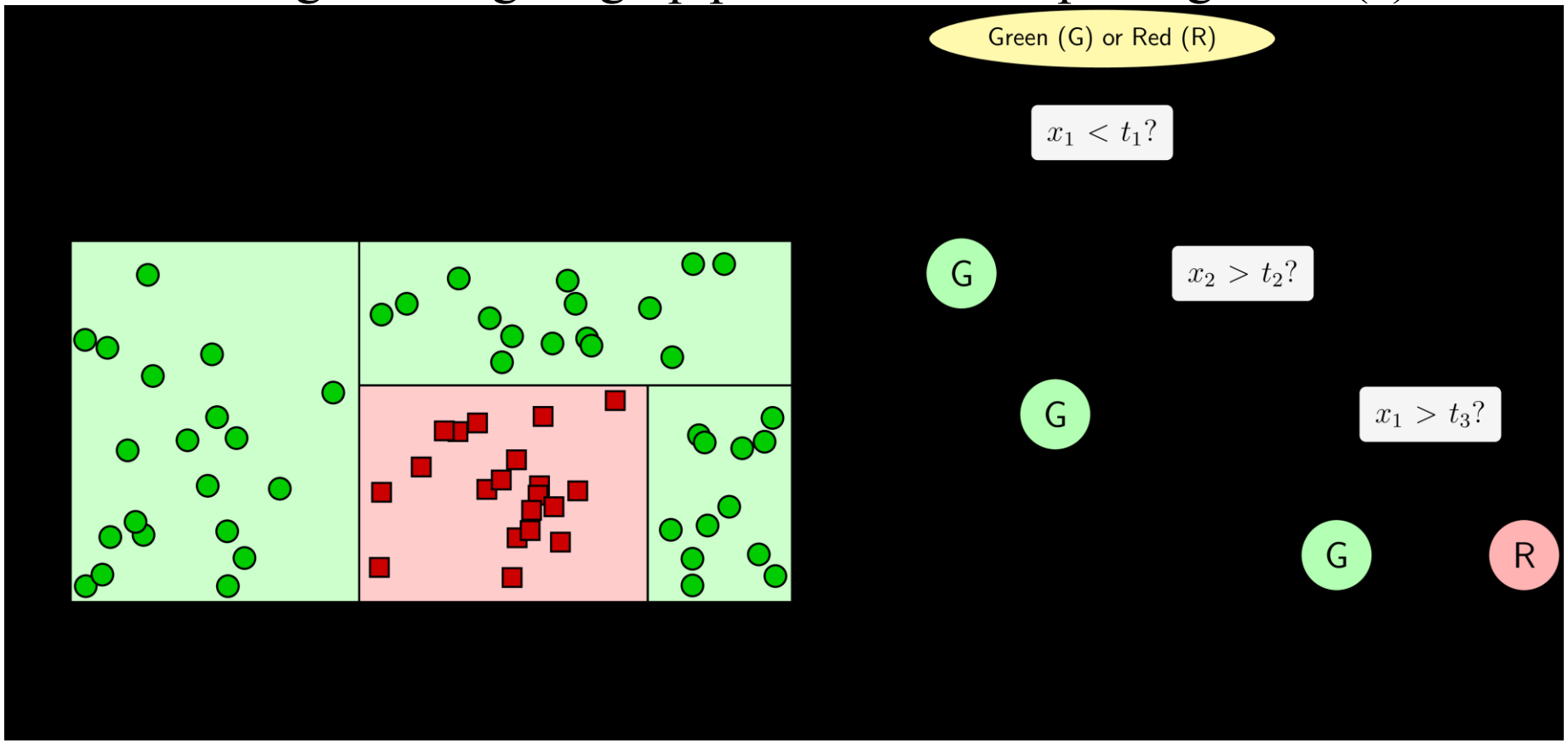
Giới thiệu

- Sắp đến kỳ thi, một SV tự đặt ra quy tắc *học* hay *chơi* của mình như sau:
  - Nếu còn nhiều hơn hai ngày, sv sẽ đi chơi
  - Nếu còn không quá hai ngày và đêm hôm đó có một trận bóng đá, SV sẽ sang nhà bạn chơi và cùng xem bóng đêm đó
  - Chỉ học trong các trường hợp còn lại



Giới thiệu

- Tìm ranh giới đơn giản giúp phân chia hai lớp trong Hình (a)

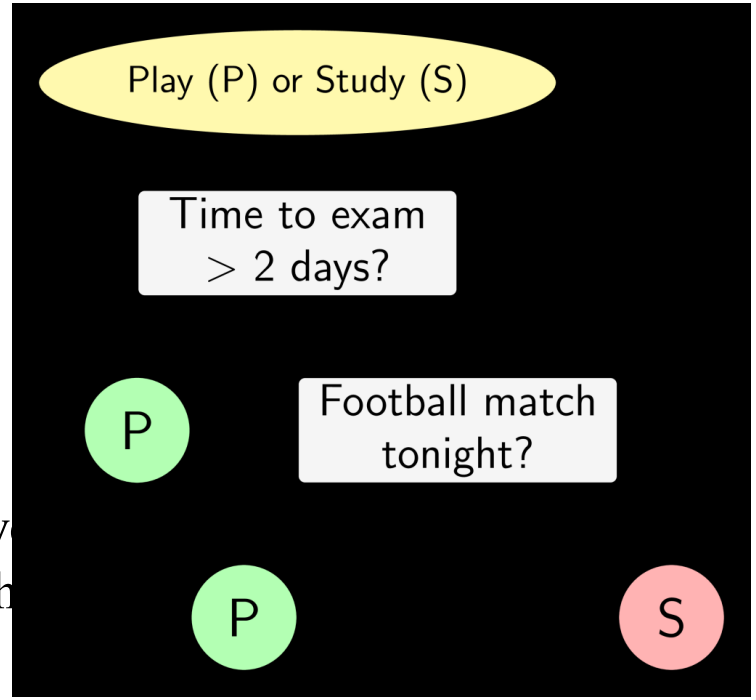


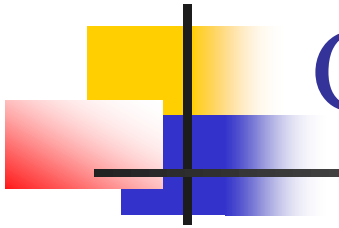
Cây quyết định

- Cây quyết định là một mô hình có giám sát (supervised learning), có thể được áp dụng cho cả bài toán phân lớp (classification) và hồi quy (regression)
- Các thuộc tính của cây quyết định có thể là:
  - Thuộc tính rời rạc và không có thứ tự (categorical). Ví dụ, *mưa, nắng* hay *xanh, đỏ*, …
  - Thuộc tính liên tục (*numeric*)

Cây quyết định

- Cây quyết định
  - Cấu trúc cây giống như biểu đồ luồng
  - Mỗi nút trong thể hiện một sự kiểm tra trên một thuộc tính
  - Mỗi nhánh đại diện cho một kết quả của sự kiểm tra
  - Các nút lá đại diện cho các nhãn lớp hoặc phân phối lớp
- Việc tạo cây quyết định gồm 2 giai đoạn
  - Xây dựng cây quyết định
    - Tại bước khởi tạo, nút gốc bao gồm tất cả các mẫu huấn luy
    - Các mẫu được phân chia đệ quy dựa trên thuộc tính được ch
  - Tỉa cây
    - Phát hiện và loại bỏ những nhánh nhiễu hoặc ngoại lệ
- Sử dụng cây quyết định: Phân lớp một mẫu chưa biết
  - Kiểm tra các giá trị thuộc tính của mẫu dựa trên cây quyết định





Cây quyết định

- Thuật toán cơ bản (thuật toán tham lam)
  - Cây quyết định được xây dựng theo cách chia để trị từ trên xuống (top-down)
  - Tại vị trí khởi tạo, tất cả các mẫu thuộc nút gốc
  - Các thuộc tính được phân loại (nếu giá trị của thuộc tính là liên tục, thì phải được rời rạc hoá trước)
  - Các mẫu được phân chia đệ quy dựa vào các thuộc tính được chọn
  - Thuộc tính kiểm tra được lựa chọn dựa vào kinh nghiệm (heuristic) hoặc độ đo thống kê (statistical measure) (vid dụ, information gain)
- Điều kiện dừng phân chia
  - Tất cả các mẫu của nút xem xét thuộc cùng một lớp
  - Không có thuộc tính nào để phân chia - biểu quyết đa số được sử dụng để gán nhãn phân loại cho lá
  - Không còn mẫu nào

Thuật toán ID3 (Iterative Dichotomiser 3)

Ý tưởng

- chúng ta cần xác định thứ tự của thuộc tính cần được xem xét tại mỗi bước.
- tại mỗi bước, một thuộc tính *tốt nhất* sẽ được chọn ra dựa trên một tiêu chuẩn nào đó.
- Với mỗi thuộc tính được chọn, ta chia dữ liệu vào các *child node* tương ứng với các giá trị của thuộc tính đó rồi tiếp tục áp dụng phương pháp này cho mỗi *child node*.
- Việc chọn ra thuộc tính *tốt nhất* ở mỗi bước như thế này được gọi là cách chọn *greedy* (*tham lam*). Cách chọn này có thể không phải là tối ưu, nhưng trực giác cho chúng ta thấy rằng cách làm này sẽ gần với cách làm tối ưu.

Thuật toán ID3

Algorithm GenDecTree(Sample S, Attlist A)

1. Tạo một nút N
2. Nếu tất cả các mẫu thuộc cùng lớp C thì N được gán nhãn C; dừng thuật toán;
3. Nếu A là rỗng thì N được gán nhãn C là nhãn phổ biến nhất trong S; dừng thuật toán;
4. Chọn a∈A, có độ đo information gain cao nhất; Gán nhãn N theo a;
5. Với mỗi giá trị v của a:
   a. Phát triển 1 nhánh từ N với điều kiện a=v;
   b. Đặt S_v là tập con của S với a=v;
   c. Nếu S_v là rỗng thì gắn một lá có nhãn phổ biến nhất trong S;
   d. Ngược lại gắn một nút được tạo bởi GenDecTree(S_v, A-a)

Thuật toán ID3 (Iterative Dichotomiser 3)

Ta cần đi tìm một phép đo *chất lượng* của một cách phân chia:

- phép phân chia tốt:
  - một phép phân chia là tốt nhất nếu dữ liệu trong mỗi *child node* hoàn toàn thuộc vào một class– khi đó *child node* này có thể được coi là một *leaf node*
  - Nếu dữ liệu trong các *child node* vẫn lẫn vào nhau theo tỉ lệ lớn, ta coi rằng phép phân chia đó chưa thực sự tốt.
- Ta cần có một hàm số đo *độ tinh khiết* (*purity*), hoặc *độ vẩn đục* (*impurity*) của một phép phân chia.
  - Hàm số này sẽ cho giá trị thấp nhất nếu dữ liệu trong mỗi *child node* nằm trong cùng một class (tinh khiết nhất),
  - cho giá trị cao nếu mỗi *child node* có chứa dữ liệu thuộc nhiều class khác nhau.
- Một hàm số có các đặc điểm này và được dùng nhiều trong lý thuyết thông tin là hàm *entropy*.

Thuật toán ID3 (Iterative Dichotomiser 3)

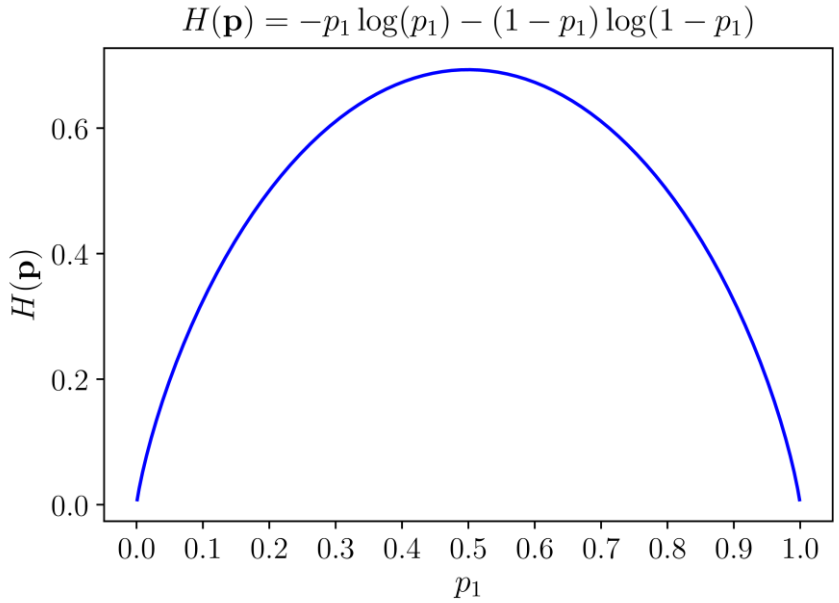
Hàm số entropy

- Cho một phân phối xác suất của một biến rời rạc x có thể nhận n giá trị khác nhau $x_1, x_2, \dots, xn$.
  - Giả sử rằng xác suất để x nhận các giá trị này là $p_i = p(x = xi)$ $với$ $0 \leq pi \leq 1, \sum_{i=1}^{n} p_i = 1$. Ký hiệu phân phối này là $p = (p_1, p_2, \dots, pn)$.
- Entropy của phân phối này được định nghĩa là

$$H(\mathbf{p}) = -\sum_{i=1}^{n} p_i \log(p_i)$$

Thuật toán ID3 (Iterative Dichotomiser 3)

- Xét một ví dụ với n=2.
  - Trong trường hợp p là *tinh khiết* nhất, tức một trong hai giá trị pi bằng 1, giá trị kia bằng 0, entropy của phân phối này là H(p)=0.
  - Khi p là *vẩn đục* nhất, tức cả hai giá trị pi=0.5, hàm entropy đạt giá trị cao nhất.



Thuật toán ID3 (Iterative Dichotomiser 3)

Thuật toán ID3

- Tìm các cách phân chia hợp lý (thứ tự chọn thuộc tính hợp lý) sao cho hàm mất mát cuối cùng đạt giá trị càng nhỏ càng tốt.
  - việc này đạt được bằng cách chọn ra thuộc tính sao cho nếu dùng thuộc tính đó để phân chia, entropy tại mỗi bước giảm đi một lượng lớn nhất.

Thuật toán ID3 (Iterative Dichotomiser 3)

Xét một bài toán với C class khác nhau:

- GS ta đang làm việc với một *non-leaf node* với các điểm dữ liệu tạo thành một tập S với số phần tử là $|S| = N$.
- Giả sử thêm rằng trong số N điểm dữ liệu này, $N_c, c = 1,2, \dots, C$ điểm thuộc vào class c. Xác suất để mỗi điểm dữ liệu rơi vào một class c được xấp xỉ bằng $\frac{N_c}{N}$(maximum likelihood estimation).
- entropy tại $$H(S) = -\sum_{c=1}^{C} \frac{N_c}{N} \log\left(\frac{N_c}{N}\right)$$

Thuật toán ID3 (Iterative Dichotomiser 3)

- GS thuộc tính được chọn là x. Dựa trên x, các điểm dữ liệu trong S được phân ra thành K child node $S_1, S_2, \ldots, SK$ với số điểm trong mỗi child node lần lượt là $m_1, m_2, \ldots, mK$. Ta định nghĩa

$$H(x, S) = \sum_{k=1}^{K} \frac{m_k}{N} H(S_k)$$

- Ta định nghĩa *information gain* dựa trên thuộc tính x

$$G(x, S) = H(S) - H(x, S)$$

- tại mỗi node, thuộc tính được chọn được xác định dựa trên:

$$x^* = \arg\max_x G(x, S) = \arg\min_x H(x, S)$$

Thuật toán ID3 (Iterative Dichotomiser 3)

Ví dụ: xây dựng cây quyết định từ bảng dữ liệu ở dưới để dự đoán liệu đội bóng có c

id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	no

Thuật toán ID3 (Iterative Dichotomiser 3)

- Có bốn thuộc tính thời tiết:
  - *Outlook* nhận một trong ba giá trị: sunny, overcast, rainy.
  - *Temperature* nhận một trong ba giá trị: hot, cool, mild.
  - *Humidity* nhận một trong hai giá trị: high, normal.
  - *Wind* nhận một trong hai giá trị: weak, strong.
- (Tổng cộng có 3×3×2×2=36 loại thời tiết khác nhau, trong đó 14 loại được thể hiện trong bảng.)

Thuật toán ID3 (Iterative Dichotomiser 3)

- Entroy tại *root node* của bài toán là:

$$H(S) = -\frac{5}{14}\log\left(\frac{5}{14}\right) - \frac{9}{14}\log\left(\frac{9}{14}\right) \approx 0.65$$

id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	no

Thuật toán ID3 (Iterative Dichotomiser 3)

- Entroy tại *root node* của bài toán là:

$$H(S) = -\frac{5}{14}\log\left(\frac{5}{14}\right) - \frac{9}{14}\log\left(\frac{9}{14}\right) \approx 0.65$$

- Xét thuộc tính *outlook*

id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes

Thuật toán ID3 (Iterative Dichotomiser 3)

id	outlook	temperature	humidity	wind	play
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes

id	outlook	temperature	humidity	wind	play
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
10	rainy	mild	normal	weak	yes
14	rainy	mild	high	strong	no

Thuật toán ID3 (Iterative Dichotomiser 3)

- Chúng ta tính được

$$H(\mathcal{S}_s) = -\frac{2}{5}\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right) \approx 0.673$$

$$H(\mathcal{S}_o) = 0$$

$$H(\mathcal{S}_r) = -\frac{3}{5}\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right) \approx 0.673$$

$$H(outlook, \mathcal{S}) = \frac{5}{14}H(\mathcal{S}_s) + \frac{4}{14}H(\mathcal{S}_o) + \frac{5}{14}H(\mathcal{S}_r) \approx 0.48$$

Thuật toán ID3 (Iterative Dichotomiser 3)

- Xét thuộc tính *temperature*

$$H(\mathcal{S}_h) = -\frac{2}{4}\log\left(\frac{2}{4}\right) - \frac{2}{4}\log\left(\frac{2}{4}\right) \approx 0.693$$

$$H(\mathcal{S}_m) = -\frac{4}{6}\log\left(\frac{4}{6}\right) - \frac{2}{6}\log\left(\frac{2}{6}\right) \approx 0.637$$

$$H(\mathcal{S}_c) = -\frac{3}{4}\log\left(\frac{3}{4}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right) \approx 0.562$$

$$H(temperature, \mathcal{S}) = \frac{4}{14}H(\mathcal{S}_h) + \frac{6}{14}H(\mathcal{S}_m) + \frac{4}{14}H(\mathcal{S}_c) \approx 0.631$$

Thuật toán ID3 (Iterative Dichotomiser 3)

- Xét thuộc tí

$$H(humidity, S) \approx 0.547.$$

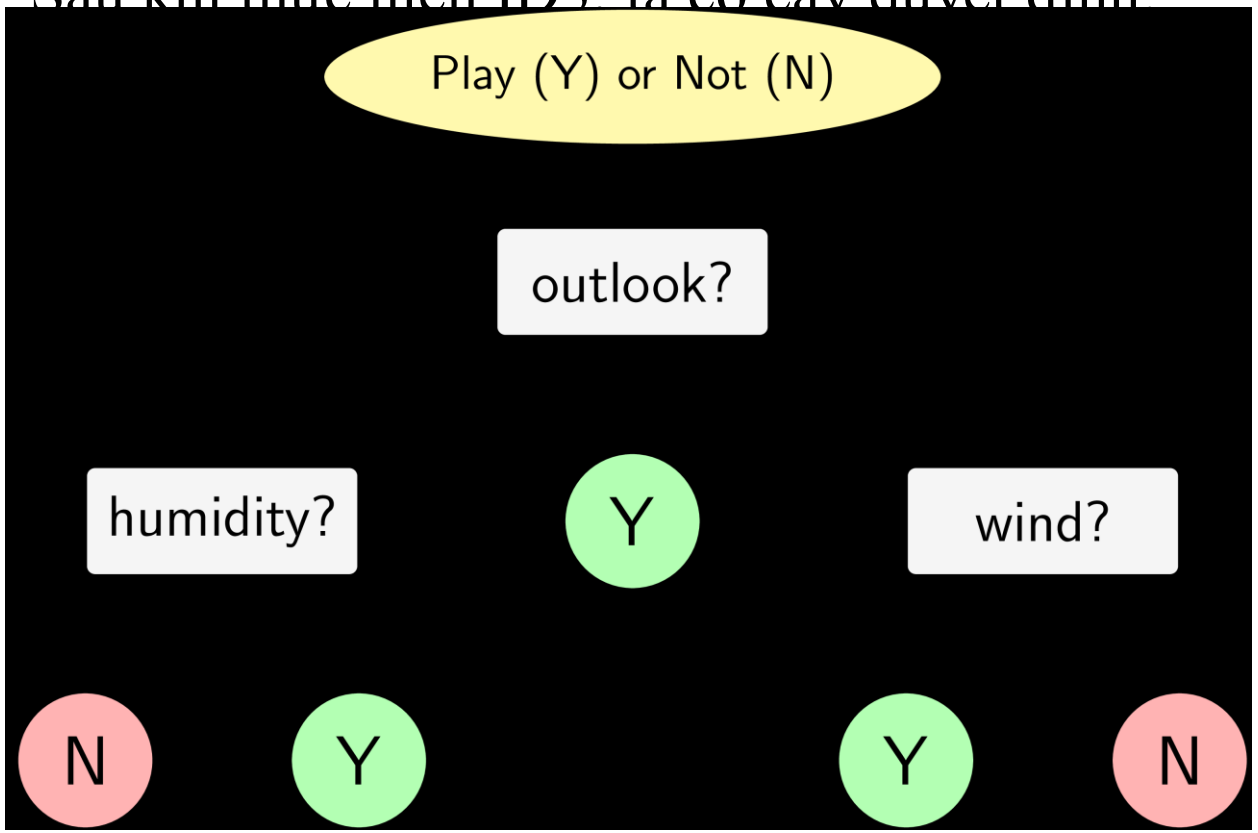
- Xét thuộc tính *wind*

$$H(wind, S) \approx 0.618$$

- Như vậy, thuộc tính cần chọn ở bước đầu tiên là *outlook* vì H(outlook,S) đạt giá trị nhỏ nhất (information gain là lớn nhất).

Thuật toán ID3 (Iterative Dichotomiser 3)

- Sau khi thực hiện ID3, ta có cây quyết định:



Thuật toán ID3 (Iterative Dichotomiser 3)

Điều kiện dừng

- nếu ta tiếp tục phân chia các node *chưa tinh khiết*, ta sẽ thu được một tree mà mọi điểm trong tập huấn luyện đều được dự đoán đúng.
- Khi đó, tree có thể sẽ rất phức tạp (nhiều node) với nhiều leaf node chỉ có một vài điểm dữ liệu. Như vậy, nhiều khả năng overfitting sẽ xảy ra.
- Để tránh overfitting, chúng ta phải có điều kiện dừng.

Thuật toán ID3 (Iterative Dichotomiser 3)

- Tại một node, nếu một trong số các điều kiện sau đây xảy ra, ta không tiếp tục phân chia node đó và coi nó là một leaf node:
  - nếu node đó có entropy bằng 0, tức mọi điểm trong node đều thuộc một class.
  - nếu node đó có số phần tử nhỏ hơn một ngưỡng nào đó. Trong trường hợp này, ta chấp nhận có một số điểm bị phân lớp sai để tránh overfitting. Class cho leaf node này có thể được xác định dựa trên class chiếm đa số trong node.
  - nếu khoảng cách từ node đó đến root node đạt tới một giá trị nào đó. Việc hạn chế *chiều sâu của tree* này làm giảm độ phức tạp của tree và phần nào giúp tránh overfitting.
  - nếu tổng số leaf node vượt quá một ngưỡng nào đó.
  - nếu việc phân chia node đó không làm giảm entropy quá nhiều (information gain nhỏ hơn một ngưỡng nào đó).

Thuật toán ID3 (Iterative Dichotomiser 3)

Pruning

- Pruning là một kỹ thuật regularization để tránh overfitting cho decision tree nói chung.
- Trong pruning:
  - một decision tree sẽ được xây dựng tới khi mọi điểm trong training set đều được phân lớp đúng.
  - Sau đó, các leaf node có chung một non-leaf node sẽ được *cắt tỉa* và non-leaf node đó trở thành một leaf-node, với class tương ứng với class chiếm đa số trong số mọi điểm được phân vào node đó.

Thuật toán ID3 (Iterative Dichotomiser 3)

Việc cắt tỉa cây quyết định này có thể được xác định dựa vào các cách sau:

- Dựa vào một validation set:
  - Trước tiên, training set được tách ra thành một training set nhỏ hơn và một validation set.
  - Decision tree được xây dựng trên training set cho tới khi mọi điểm trong training set được phân lớp đúng.
  - Sau đó, đi ngược từ các leaf node, cắt tỉa các sibling node của nó và giữ lại node *bố mẹ* nếu độ chính xác trên validation set được cải thiện.
  - Khi nào độ chính xác trên validation set không được cải thiện nữa, quá trình pruning dừng lại.

Thuật toán ID3 (Iterative Dichotomiser 3)

- Dựa vào toàn bộ data set:
  - sử dụng toàn bộ dữ liệu trong tập này cho việc xây dựng decision tree.
  - Một ví dụ cho việc này là cộng thêm một đại lượng regularization vào hàm mất mát.
  - Đại lượng regularization sẽ lớn nếu số leaf node là lớn. Cụ thể:
    - giả sử decision tree cuối cùng có K leaf node, tập hợp các điểm huấn luyện rơi vào mỗi leaf node lần lượt là $S_1, ..., S_K$. Khi đó, regularized loss của ID3 có thể được tính tương tự như:

$$\mathcal{L} = \sum_{k=1}^{K} \frac{|\mathcal{S}_k|}{|\mathcal{S}|} H(\mathcal{S}_k) + \lambda K$$