Trường Đại học Công nghệ thông tin
Đại học quốc gia Thành phố Hồ Chí Minh

Báo cáo Bài tập 0

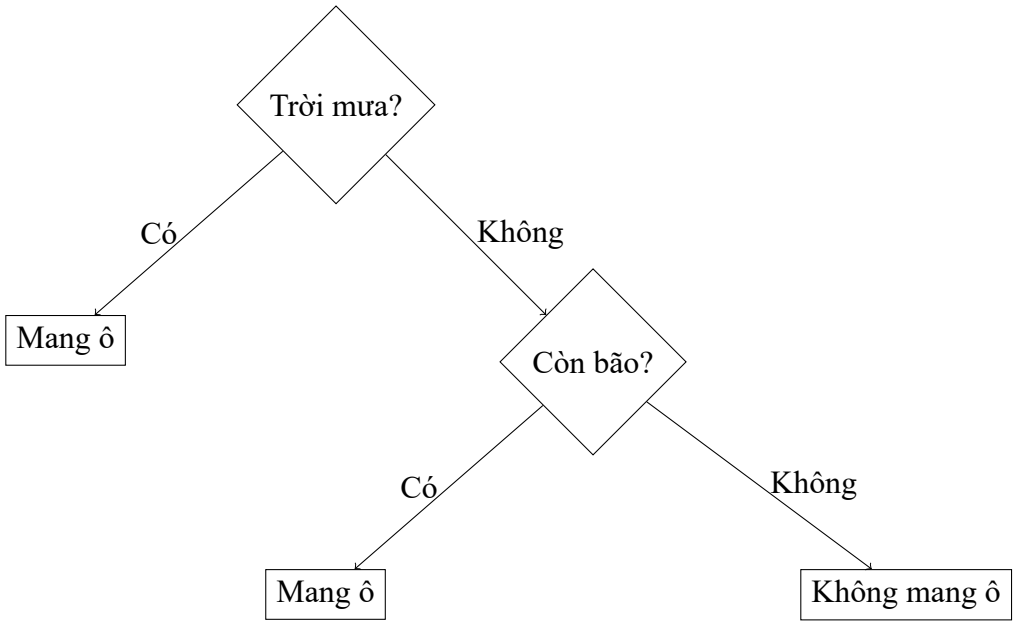
Họ và tên: Phan Đức Anh
MSSV: 23520071
Giảng viên: Lương Ngọc Hoàng
Lớp tham gia: CS115.Q15

1 Giới thiệu - Decision Trees

1.1 Decision Tree *thực chất* là gì?

Decision Trees (Cây quyết định), về bản chất là một flowchart mà máy tính có thể tự học được. [2]

1.2 Ví dụ trực quan



Hình 1: Ví dụ Decision Tree để quyết định có mang ô hay không?

1.3 Mục tiêu

Mục tiêu của cây quyết định là liên tục đặt câu hỏi để phân chia tập dữ liệu thành các nhóm con ngày càng **thuần khiết** hơn, nghĩa là tất cả phần tử thuộc nhóm con này đều thuộc về cùng một lớp.[1]

Ví dụ: Với ngữ cảnh cụ thể là khách hàng có mua sản phẩm không, chỉ với một câu hỏi ”Người dùng có mua sản phẩm không?” thì ta đã có thể phân chia dữ liệu thành hai nhóm hoàn toàn riêng biệt (hay còn gọi là **thuần khiết**): nhóm người dùng mua sản phẩm và nhóm người dùng không mua sản phẩm.

2 Decision Trees hoạt động như thế nào?

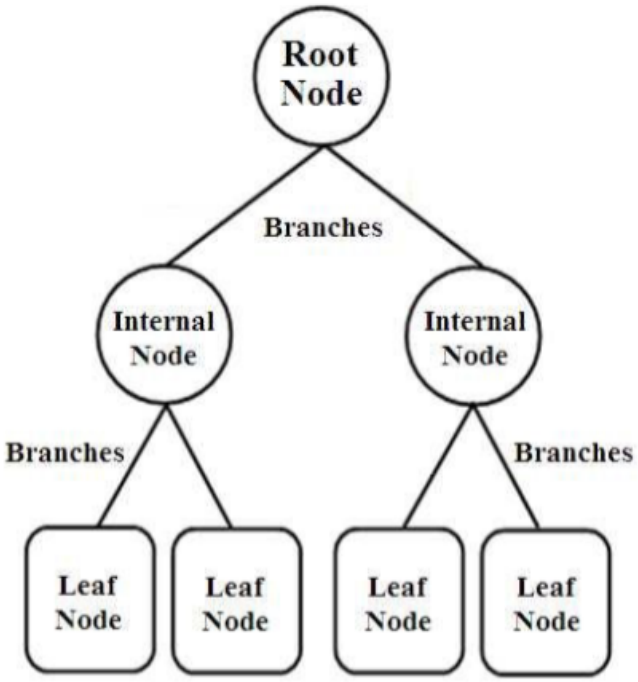
Làm sao để Decision Trees đặt được câu hỏi tốt nhất ở mỗi giai đoạn?

- Quá trình học của Decision Trees gồm nhiều bước mà tại đó tập dữ liệu được chia ra thành các tập con **thuần khiết** hơn như trong Figure 2.
- Root node (node gốc) đại diện toàn bộ tập dữ liệu.
- Thuật toán sẽ xác định đặc trưng trong tập dữ liệu và ngưỡng thích hợp để chia tập dữ liệu thành các **nhóm con** thuần khiết nhất dựa trên một vài tiêu chí cụ thể gọi là **split criteria (tiêu chuẩn chia tách)**.
- Quá trình này tiếp tục theo phương pháp đệ quy, mỗi tập dữ liệu sẽ được chia nhỏ hơn nữa tại mỗi **nhóm con**.
- Điều này sẽ tiếp tục cho tới khi một số điều kiện dừng được thỏa mãn, điển hình là khi các **nhóm con** đã đủ thuần khiết hoặc không còn đặc trưng nào để chia tách nữa hoặc khi độ sâu của cây (được định nghĩa trước) đã đạt giá trị tối đa.
- Node cuối cùng (không thể chia tách nữa) gọi là **node lá** - đại diện cho dự đoán cuối cùng của mô hình hoặc nhãn của lớp đó.
- Các thuật toán xây dựng Decision Trees khác nhau như CART, ID3, C4.5, CHAID,...

(Toàn bộ nội dung trong mục này được tham khảo từ [1])
Thành công của Decision Trees phụ thuộc vào đâu? Làm sao để xây dựng được một cây quyết định tốt?

- Chất lượng dữ liệu.
- Độ sâu của cây.
- Tiêu chí chia tách.
- Phương pháp cắt tỉa cây.
- Lựa chọn đặc trưng, tăng cường dữ liệu, trích xuất đặc trưng.

(Toàn bộ nội dung trong mục này được tham khảo từ [1])



Hình 2: Cấu trúc của Decision Tree

3 Thuật toán xây dựng và tiêu chuẩn split

3.1 Các tiêu chuẩn chia tách (split criteria)

Khái niệm: Là phương pháp mà cây dùng để quyết định đặt câu hỏi thế nào để chia dữ liệu hiện tại thành các nhóm con với **độ bất thuần khiết Gini** thấp nhất. Đồng nghĩa với, các nhóm con tạo ra chứa các phần tử thuộc cùng một lớp càng nhiều càng tốt.[1]

Các thuật toán cây quyết định khác nhau sử dụng các tiêu chuẩn chia tách khác nhau, bao gồm:

3.1.1 Gini Index

Gini Index hay còn gọi là **Gini Impurity** là tiêu chuẩn chia tách được sử dụng trong thuật toán **CART (Classification and Regression Trees)**. [1]
Gini Impurity là con số đo lượng độ **bất thuần khiết** của tập dữ liệu.
Gini Index đo lường xác suất khi một mẫu dữ liệu **được lựa chọn ngẫu nhiên** sẽ bị phân loại sai.
Ví dụ: Nếu có 10 quả bóng, 3 quả màu xanh, 7 quả màu đỏ, vậy ta có hai trường hợp sẽ đoán sai:

- Chọn quả bóng màu xanh (xác suất 30%), đoán sang màu đỏ - xác suất đoán sai là 70% $\Rightarrow$ Xác suất đoán sai là $0.3 * 0.7 = 0.21$
- Chọn quả bóng màu đỏ (xác suất 70%), đoán sang màu xanh - xác suất đoán sai là 30% $\Rightarrow$ Xác suất đoán sai là $0.7 * 0.3 = 0.21$
- Tổng xác suất đoán sai là $0.21 + 0.21 = 0.42$ - đây chính là **Gini Impurity** của tập dữ liệu này.

Công thức:

$$Gini(S) = 1 - \sum_{i=1}^{n} p_i^2$$

Trong đó, S là tập các mẫu dữ liệu, n là số lớp (classes) trong dữ liệu, p_i là tỉ lệ mẫu thuộc lớp i.
Công thức này cho biết xác suất phân loại sai một mẫu ngẫu nhiên từ tập S dựa trên phân phối của các lớp trong tập S.

Vậy công thức này sẽ hoạt động thế nào với ví dụ trên?

- Gọi p_i là tỉ lệ của mẫu thuộc lớp i trong tập S (Xanh: 0.3 và Đỏ: 0.7).
- Xác suất bốc được 1 quả bóng thuộc lớp i và đoán đúng nó cũng thuộc lớp i (Ví dụ chọn quả xanh và đoán đúng nó là xanh):
  - Xác suất bốc được quả xanh và đoán đúng nó là xanh: 0.3 * 0.3 = 0.09
  - Xác suất bốc được quả đỏ và đoán đúng nó là đỏ: 0.7 * 0.7 = 0.49
  - Tổng xác suất đoán đúng là 0.09 + 0.49 = 0.58
- p_i^2 là xác suất mà ta đoán đúng.

- Vậy $1 - p_i^2 = 1 - 0.58 = 0.42$ là xác suất mà ta đoán sai, chính là **Gini Impurity** của tập dữ liệu này.

Vậy cách mà Gini Index hoạt động trong thuật toán như thế nào?

- Nó sẽ xem xét tất cả cách chia tách dữ liệu có thể.
- Với mỗi cách chia, nó sẽ tính **Gini Impurity** của các nhóm con được tạo ra.
- Nó tính điểm **Gini Impurity** trung bình có trọng số của các nhóm con này và chọn cách chia có điểm số tổng thể này là thấp nhất.

Ví dụ: Giả sử có một tập dữ liệu có 100 mẫu (node) với Gini Impurity là 0.5, ta có hai cách chia là cách chia A và cách chia B. Với cách chia A.

- Nhóm con trái (60 mẫu) có Gini Impurity là 0.3 và nhóm con phải (40 mẫu) có Gini Impurity là 0.4.
- Điểm số tổng thể được tính như sau:

$$Gini_A = \frac{60}{100} * 0.3 + \frac{40}{100} * 0.4 = 0.18 + 0.16 = 0.34$$

Với cách chia B.

- Nhóm con trái (50 mẫu) có Gini Impurity là 0.1 và nhóm con phải (50 mẫu) có Gini Impurity là 0.2.
- Điểm số tổng thể được tính như sau:

$$Gini_B = \frac{50}{100} * 0.1 + \frac{50}{100} * 0.2 = 0.05 + 0.1 = 0.15$$

Vậy thuật toán sẽ chọn cách chia B vì điểm số tổng thể của nó là thấp nhất.

(Toàn bộ nội dung trong mục này được tham khảo từ [1])

3.1.2 Information Gain (IG) và Entropy

Information Gain (IG) là tiêu chuẩn chia tách được sử dụng trong thuật toán **ID3 (Iterative Dichotomiser 3)** và **C4.5**, được dựa trên khái niệm của **Entropy** trong lý thuyết thông tin.[1]

Nhưng tổng kết lại, Entropy là gì?

Entropy là thước đo mức độ **hỗn loạn** hoặc mức độ **bất định** trong một hệ thống.

- Entropy cao = Hỗn loạn tối đa = Thông tin tối thiểu.
  - Một đồng xu dự đoán sấp/ngửa có xác suất 50/50, ta không biết gì cả, mọi kết quả đều có tính khả thi như nhau, việc dự đoán là gần như vô nghĩa.
- Entropy thấp = Hỗn loạn tối thiểu = Thông tin
  - Một đồng xu bị gian lận có 70/30 tỉ lệ ra mặt sấp/ngửa, ta biết rất nhiều, việc dự đoán trở nên khả thi hơn.

Suy cho cùng, **IG** là một thước đo mức độ **lợi nhuận về mặt thông tin** khi chia tập dữ liệu thành các nhóm con dựa trên một đặc trưng cụ thể.

Công thức tính Entropy:

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

Trong đó, S là tập các mẫu dữ liệu, n là số lớp (classes) trong dữ liệu, p_i là tỉ lệ mẫu thuộc lớp i.
Công thức này cho biết mức độ bất định của một tập dữ liệu dựa trên phân phối của các lớp trong tập dữ liệu đó (Sự **hỗn loạn** của toàn bộ tập dữ liệu ở trạng thái hiện tại).

Công thức tính Information Gain:

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

Trong đó, S là tập các mẫu dữ liệu (node hiện tại), A là đặc trưng dùng để chia tách dữ liệu (Attribute).

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

$Values(A)$ là tập hợp các giá trị duy nhất của đặc trưng A. Ví dụ: Nếu A = ”Màu sắc” thì $Values(A)$ là Đỏ, Xanh.
S_v là tập con của S mà đặc trưng A có giá trị v. Ví dụ: Nếu v = ”Đỏ” thì $S_v <=> S_{red}$ là tập các giá trị trong S có màu sắc là Đỏ.
$|S_v|$ là số lượng mẫu trong tập con S_v.
$|S|$ là số lượng mẫu trong tập S.
$|S_v|/|S|$ là tỉ lệ mẫu trong tập con S_v so với tập S.

Công thức này cho biết **sự suy giảm về mức độ bất định** khi chia tập dữ liệu thành các nhóm con dựa trên đặc trưng A.

Ví dụ: Với tập dữ liệu có 100 hộp quà, 50 hộp có quà và 50 hộp không quà

$$E(S) = -(0.5 * log_2(0.5) + 0.5 * log_2(0.5)) = 1.0$$

Định nghĩa cách chia A = ”Màu sắc” với Values(A) = Đỏ, Xanh. Giả sử có 40 hộp đỏ và 60 hộp xanh.
Đầu tiên, v1 = ”Đỏ”. S_{v1} = 40 (35 có quà, 5 không có quà)

$$E(S_{v1}) = -(5/40 * log_2(\frac{5}{40}) + \frac{35}{40} * log_2(\frac{35}{40})) \cong 0.5435$$

Tiếp theo, v2 = ”Xanh”. S_{v2} = 60 (15 có quà, 45 không có quà)

$$E(S_{v2}) = -(15/60 * log_2(\frac{15}{60}) + \frac{45}{60} * log_2(\frac{45}{60})) \cong 0.811$$

Vậy ta có:

$$IG(S, A) = 1.0 - (\frac{40}{100} * 0.5435 + \frac{60}{100} * 0.811) = 1.0 - (0.2174 + 0.4866) = 0.296$$

Vậy, việc chia tách dựa trên đặc trưng ”Màu sắc” mang lại **lợi ích về mặt thông tin** hay còn gọi là **sự suy giảm Entropy** là 0.296.

Và, (0.2174 + 0.4866) = 0.704 là **mức độ bất định còn lại** sau khi chia tách dữ liệu dựa trên đặc trưng ”Màu sắc”.

Phụ lục: Giải thích công thức Entropy

Tại sao lại có log_2 ***trong công thức?***

- Giả sử ta có 8 cái hộp, chỉ có một hộp có quà. Cả 8 hộp đều có khả năng chứa quà như nhau.
  - Xác suất để mở trúng hộp có quà là $1/8 = 0.125$.
- Công việc của ta sẽ là đặt câu hỏi để loại trừ bớt số hộp cần mở. Chiến lược loại tốt nhất là chia đôi.
  - Câu hỏi 1: Hộp có quà có nằm trong nửa bên trái không? (Loại bỏ được 4 hộp)
  - Câu hỏi 2: Trong 4 hộp còn lại, hộp có quà có nằm trong nửa bên trái không? (Loại bỏ được 2 hộp)
  - Câu hỏi 3: Trong 2 hộp còn lại, hộp có quà có nằm trong nửa bên trái không? (Tìm ra hộp có quà)
- Vậy ta cần 3 câu hỏi để tìm ra hộp có quà <=> Đây là 3 lần ta chia đôi số hộp ban đầu hay nói cách khác là chia đôi dữ liệu trong ngữ cảnh bài toán.
- Suy ra, số câu hỏi cần đặt cho 8 hộp quà là $log_2(8) = 3$.
- Tương đương, số lần phân chia dữ liệu cho **n** mẫu dữ liệu là $log_2(n)$.

Nhưng trong công thức là $log_2(p_i)$***?***

- Ta có $p_i = 1/n$ => $log_2(p_i) = log_2(1/n) = -log_2(n)$.
- Vậy $-log_2(p_i) = log_2(n)$ - chính là số lần ta cần chia đôi dữ liệu để thu được một mẫu thuộc lớp i.

Vậy $-log_2(p_i)$ ***đóng vai trò gì trong công thức trên?***

- $-log_2(p_i)$ đo lường **lượng thông tin** mà ta nhận được khi một sự kiện với xác suất p_i xảy ra.

- Với 1 sự kiện rất hiếm $\rightarrow p_i$ nhỏ $\rightarrow -log_2(p_i)$ lớn $\rightarrow$ việc xảy ra nó mang nhiều thông tin hơn.
- Với 1 sự kiện rất phổ biến $\rightarrow p_i$ lớn $\rightarrow -log_2(p_i)$ nhỏ $\rightarrow$ việc xảy ra nó mang ít thông tin hơn
- Ví dụ: Sự kiện thiên nga đen rất hiếm $\rightarrow$ việc nhìn thấy nó mang nhiều thông tin hơn so với việc nhìn thấy một con chim sẻ (rất phổ biến).

Tại sao $-log_2(p_i)$ ***chỉ để tính số lần phân chia lại đại diện cho lượng thông tin?***

- Bởi vì, theo định nghĩa, **lượng thông tin** chính là **số lần phân chia hoàn hảo**.
- Đoán 1 số từ 1 đến 16.
- Khi không biết gì, có 16 khả năng.
- Nếu đặt câu hỏi là "Số đó có phải là 7 không?" - chỉ loại được 1 khả năng, xác suất loại bỏ được 15 khả năng (khi số 7 là số đúng) chỉ có $1/15 = 0.0667$ - rất thấp.
- Nếu đặt câu hỏi là "Số đó có lớn hơn 8 không" - Chỉ với 1 câu trả lời "Có/Không" ta có thể loại 1 nửa số khả năng -> vậy câu hỏi này đem lại **1 bit thông tin**.
- '1 bit' đại diện cho 1 câu trả lời cho 1 câu hỏi "Có/Không" mà có thể loại bỏ một nửa số khả năng hiện có (1 lần chia đôi hoàn hảo).

(Toàn bộ phụ lục chú giải này tham khảo từ [3]).

3.2 Thuật toán xây dựng Decision Tree

3.2.1 Iterative Dichotomiser 3 (ID3)

- 1. Tính **Entropy** cho tập dữ liệu gốc.
- 2. Với mỗi đặc trưng trong tập dữ liệu, tính **Information Gain (IG)** khi chia tách dữ liệu sử dụng đặc trưng đó.
- 3. Chọn đặc trưng có độ giảm bất định nhiều nhất **Information Gain cao nhất** làm đặc trưng để chia tách dữ liệu tại node hiện tại.

- 4. Chia nhánh dựa vào thuộc tính đã chọn, đối với mỗi giá trị của thuộc tính, tách dữ liệu ra thành các nhóm con nhỏ hơn.
- 5. Lặp lại quy trình bước 2, 3, 4 cho mỗi nhóm con, sử dụng tập dữ liệu con tương ứng.
- 6. Dừng lại khi thỏa một số điều kiện dừng hoặc nhánh đó chỉ còn toàn bộ dữ liệu cùng 1 nhãn (Ví dụ: Tất cả hộp đều có quà).

Tài liệu

[1] Ibomoiye Domor Mienye et al. "A Survey of Decision Trees: Concepts, Algorithms, and Applications." In: *ResearchGate* (2024). URL: https://www.researchgate.net/publication/381564302_A_Survey_of_Decision_Trees_Concepts_Algorithms_and_Applications.

[2] GeeksforGeeks. *Decision Tree Implementation in Python*. Accessed: 2025-02-01. 2023. URL: https://www.geeksforgeeks.org/machine-learning/decision-tree-implementation-python/.

[3] Claude E. Shannon. "A Mathematical Theory of Communication." In: *Bell System Technical Journal* 27 (1948), pp. 379–423. URL: https://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf.

4 Triển khai code

Google Colab Link