TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH

KHOA ĐÀO TẠO CHẤT LƯỢNG CAO

NGÀNH CÔNG NGHỆ THÔNG TIN

🙡🙣



BÁO CÁO ĐỒ ÁN 2

Đề Tài: Tìm hiểu về Decision Tree và thuật toán ID3

GVHD: TS. Trần Nhật Quang

SVTH: Hứa Văn Lâm 16110133

Nguyễn Trọng Đạt 16110049

Lớp: 16110ST3

**TP. Hồ Chí Minh, tháng 05 năm 2019**

**MỤC LỤC**

[**GIỚI THIỆU** 1](#_Toc9535418)

[**1.** **Giới thiệu** 1](#_Toc9535419)

[**2.** **Cấu trúc Decision Tree** 1](#_Toc9535420)

[**ITERATIVE DICHOTOMISER 3 (ID3)** 3](#_Toc9535421)

[**1.** **Giới thiệu ID3** 3](#_Toc9535422)

[**2.** **Hàm số Entropy** 3](#_Toc9535423)

[**3.** **Ví dụ** 3](#_Toc9535424)

[**LẬP TRÌNH VỚI PYTHON** 8](#_Toc9535425)

[**1.** **Thiết kế lớp** 8](#_Toc9535426)

[**2.** **Cài đặt và kiểm thử** 9](#_Toc9535427)

[**a.** **Dataset[3]** 9](#_Toc9535428)

[**b.** **F1-SCORE** 9](#_Toc9535429)

[**c.** **Các thư viện sử dụng** 9](#_Toc9535430)

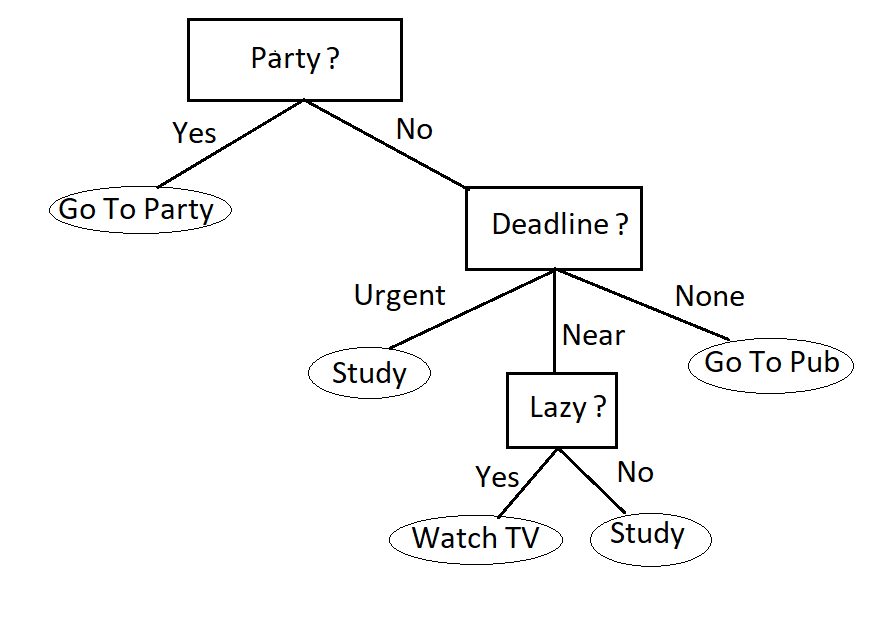
[**d.** **Kiểm thử[4]** 10](#_Toc9535431)

[**e.** **Đánh giá** 12](#_Toc9535432)

[**TÀI LIỆU THAM KHẢO** 14](#_Toc9535433)

# **GIỚI THIỆU**

## **Giới thiệu**



Hình 1: Ví dụ về quyết định được đưa ra trên các câu hỏi.

Nhìn vào hình trên ta thấy những ô vuông chứa Party? Deadline? Lazy? rồi rẽ nhánh ra nhiều hướng khác. Các nhánh đó kết thúc tại vòng tròn hay thêm các ô vuông khác. Vậy đó là gì ???

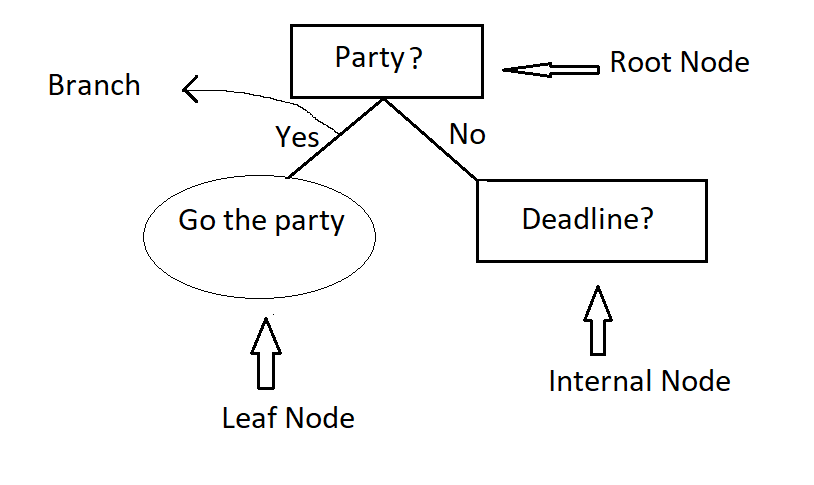
Thực chất đây có thể coi là 1 Decision Tree dựa trên các câu hỏi của một người. Để dễ hiểu hơn thì tôi sẽ hỏi chính bản thân tôi:

* Nếu như có Party thì tôi đi Party còn không có thì xem Deadline gần tới chưa. Nếu là None thì Go To Pub, còn nếu Deadline là Near thì xem mình có Lazy không nếu có thì Watch TV còn không thì Study. Nếu Deadline là Urgent thì tôi phải Study ngay lập tức.

Vậy Decision Tree là cây mà mỗi node biểu diễn một thuộc tính. Các nhánh (branch) biểu diễn cho một quy luật và các lá biểu diễn cho kết quả (là giá trị cụ thể hay 1 nhánh kế tiếp).

1. **Cấu trúc Decision Tree**

Cấu trúc của Decision Tree gồm 4 thành phần: node gốc (Root node), node lá (Leaf node), node con (Internal node), nhánh (Branch). Node gốc là nơi bắt đầu của Decision Tree, nhánh biễu diễn các kết quả của các node, đích của nhánh có thể là một node lá hoặc là một node con.



Hình 2: Cấu trúc của Decision Tree.

Để tạo một Decision Tree, ta sẽ cần dùng thuật toán ID3 một trong những thuật toán tạo Decision Tree.

**ITERATIVE DICHOTOMISER 3 (ID3)**

1. **Giới thiệu ID3**

Giải thuật ID3 (gọi tắt là ID3) Được phát triển đồng thời bởi Quinlan trong AI và Breiman, Friedman, Olsen và Stone trong thống kê. ID3 là một giải thuật học đơn giản nhưng tỏ ra thành công trong nhiều lĩnh vực. ID3 biểu diễn các khái niệm ở dạng các decision tree. Biểu diễn này cho phép chúng ta xác định phân loại của một đối tượng bằng cách kiểm tra các giá trị của nó trên một số thuộc tính nào đó.[1]

1. **Hàm số Entropy**

Entropy H(s) là thước đo xem dữ liệu có sạch không (dữ liệu giống nhau hết là sạch).

2019-05-21_165851

Chú thích:

+ S: dataset.

+ p(c): tỉ lệ số phần tử giữa lớp c với lớp S.

+ c: lớp con của dataset.

+ C: tập hợp các output.

Dựa vào Entropy ta tìm node cho cây bằng cách tính Entropy của mỗi lớp con trong dataset. Lớp nào có Entropy nhỏ nhất thì ta lấy làm node và vòng lặp như vậy tới khi hết dataset thì thôi.

1. **Ví dụ**

Mình sẽ dùng tập dữ liệu nổi tiếng trong ngành máy học - dữ liệu thời tiết (weatherdataset) - đưa ra “yes” hoặc “no” dựa vào thời tiết.[2]



* Tạo cây:

B1: Gọi dataset có các lớp con là outlook, temp, humidly, windy.

B2: Tính Entropy cho mỗi lớp:

+ Liệt kê lớp outlook.

|  |  |  |
| --- | --- | --- |
| Sunny | 2 Yes | 3 No |
| Overcast | 4 Yes |  |
| Rainy | 3 Yes | 2 No |

Tính Entropy:

H(Sunny) = -(2/5)\*Log2(2/5) - (3/5)\*Log2(3/5)

H(Overcast) = -(4/4)\*Log2(4/4)

H(Rainy) = -(3/5)\*Log2(3/5) - (2/5)\*Log2(2/5)

=> Tổng Entropy của outlook:

H(outlook) = (5/14)\*H(Sunny) + (4/14)\*H(Overcast) + (5/14)\*H(Rainy) = 0.69

+ Liệt kê lớp temp

|  |  |  |
| --- | --- | --- |
| Hot | 2 Yes | 2 No |
| Mild | 4 Yes | 2 No |
| Cool | 3 Yes | 1 No |

Tính Entropy:

H(Hot) = -(2/4)\*Log2(2/4) - (2/4)\*Log2(2/4)

H(Mild) = -(4/6)\*Log2(2/5) - (3/5)\*Log2(4/6)

H(Cool) = -(3/4)\*Log2(3/4) - (1/4)\*Log2(1/4)

=> Tổng Entropy của temp:

H(temp) = (4/14)\*H(Hot) + (6/14)\*H(Mild) + (4/14)\*H(Cool) = 0.91

+ Liệt kê lớp humidity

|  |  |  |
| --- | --- | --- |
| High | 3 Yes | 4 No |
| Normal | 6 Yes | 1 No |

Tính Entropy

H(High) = -(3/7)\*Log2(3/7) - (4/7)\*Log2(4/7)

H(Normal) = -(6/7)\*Log2(6/7) - (1/7)\*Log2(1/7)

=> Tổng Entropy của humidity:

H(humidity) = (7/14)\*H(High) + (7/14)\*H(Normal) = 0.79

+ Liệt kê lớp windy

|  |  |  |
| --- | --- | --- |
| Fasle | 6 Yes | 2 No |
| True | 3 Yes | 3 No |

Tính Entropy

H(Fasle) = -(6/8)\*Log2(6/8) - (2/8)\*Log2(2/8)

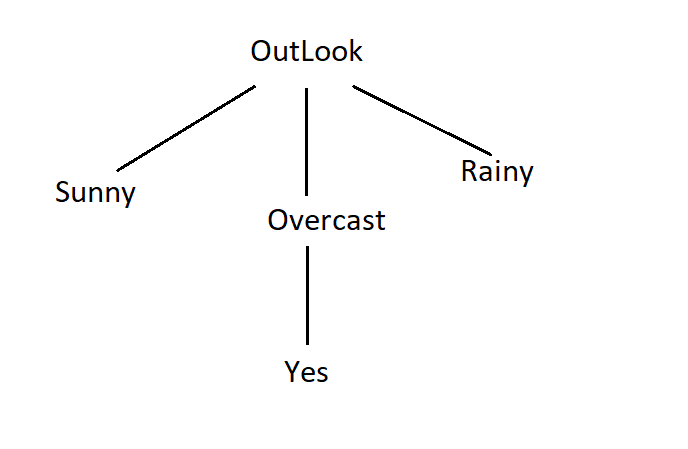
H(True) = -(3/6)\*Log2(3/6) - (3/6)\*Log2(3/6)

=> Tổng Entropy windy:

H(windy) = (7/14)\*H(Fasle) + (7/14)\*H(True) = 0.89

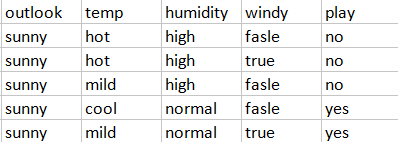
B3: Chọn node

Vì Entropy outlook là nhỏ nhất nên node đầu tiên cũng như node root là outlook.



Tại sao Overcast lại là có nhánh Yes luôn? do Overcast output của nó toàn dữ liệu trùng nên nó được cho là dữ liệu sạch.

Tương tự như vậy ta thực hiện 3 bước trên để tìm node tiếp theo với dataset bây giờ là dataset Sunny.



B1: Dataset Sunny có các lớp: temp, humidity, windy

B2: Tính Entropy cho mỗi lớp:

+ Liệt kê temp

|  |  |  |
| --- | --- | --- |
| Hot |  | 2 No |
| Mild | 1 Yes | 1 No |
| Cool | 1 Yes |  |

+ Tính Entropy

H(Hot) = - (2/2)\*Log2(2/2)

H(Mild) = -(1/2)\*Log2(1/2) - (1/2)\*Log2(1/2)

H(Cool) = -(1/1)\*Log2(1/1)

=> Tổng Entropy của temp:

H(temp) = (2/5)\*H(Hot) + (2/5)\*H(Mild)+ (1/5)\*H(Cool) = 0.4

+ Liệt kê humidity

|  |  |  |
| --- | --- | --- |
| High |  | 3 No |
| Normal | 2 Yes |  |

+ Tính Entropy

H(High) = - (3/3)\*Log2(3/3)

H(Normal) = -(2/2)\*Log2(2/2)

=> Tổng Entropy của humidity:

H(temp) = (3/5)\*H(High) + (2/5)\*H(Normal) = 0

+ Liệt kê lớp windy

|  |  |  |
| --- | --- | --- |
| Fasle | 1 Yes | 2 No |
| True | 1 Yes | 1 No |

+ Tính Entropy

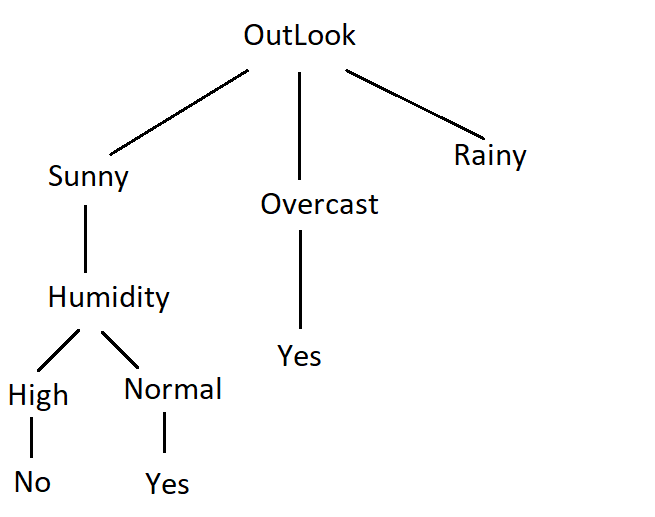
H(Fasle) = -(1/3)\*Log2(1/3) - (2/3)\*Log2(2/3)

H(True) = -(1/1)\*Log2(1/1) - (1/1)\*Log2(1/1)

=> Tổng Entropy windy :

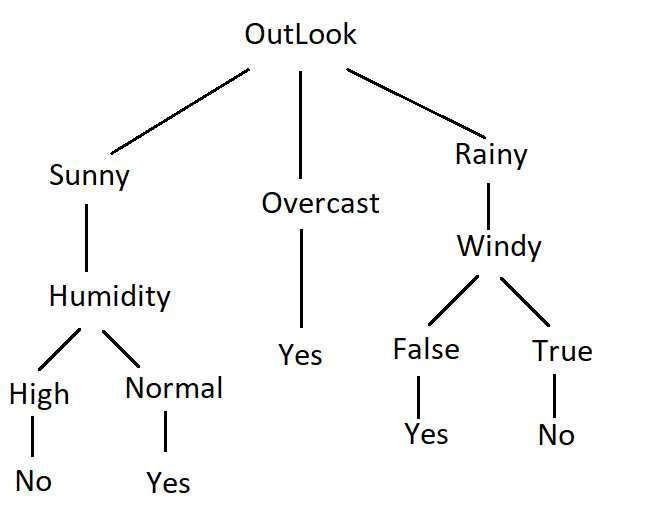
H(windy) = (3/5)\*H(Fasle) + (2/5)\*H(True) = 0.55

Như vậy từ node Sunny ta rẽ thêm nhánh là humidity do lớp humidity toàn dữ liệu sạch nên nhánh này coi như kế thúc.



Cứ tiếp tục với các dataset Temp và dataset Windy bằng 3 bước trên thì ta sẽ hoàn thành 1 cái cây.

Cây đã hoành thành:



Tại sao lại ko có Temp? Vậy khi nào dừng? Khi các nhánh của nó là dữ liệu sạch thì nhánh đó coi như đã xong và đó cũng là điều kiện dừng lại khi tạo cây.

**LẬP TRÌNH VỚI PYTHON**

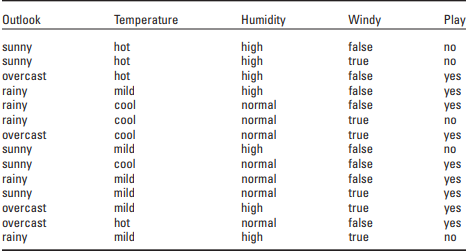
1. **Thiết kế lớp**

|  |  |  |  |
| --- | --- | --- | --- |
| Stt | Tên phương thức | Mục đích | Tên file, stt dòng chứa khai báo |
| 1 | printTree(root) | In ra cây | Testid3vs1.py  Dòng 11 |
| 2 | id3(df\_train,root) | Trả về một cây | Testid3vs1.py  Dòng 26 |
| 3 | entropycategorys(df\_train) | Trả về cột có số entropy nhỏ nhất | Testid3vs1.py  Dòng 53 |
| 4 | entropycategoryR(df\_train,index) | Trả về số entropy của mỗi cột | Testid3vs1.py  Dòng 61 |
| 5 | entropycategoryC(df\_train\_index) | Trả về số entropy của mỗi cụm dữ liệu trong một cột | Testid3vs1.py  Dòng 73 |
| 6 | socum(df\_train\_Cindex) | Trả về mảng cụm dữ liệu có trong một cột | Testid3vs1.py  Dòng 92 |
| 7 | kttontai(array,i) | Kiểm tra sự tồn tại của một điểm dữ liệu trong một mảng | Testid3vs1.py  Dòng 100 |
| 8 | test(df\_test,root) | Trả về dữ liệu dự đoán dựa trên cây quyết định | Testid3vs1.py  Dòng 107 |
| 9 | f1\_score(y,y\_pred) | Tính f1-score | Testid3vs1.py  Dòng 125 |
| 10 | getData() | Lấy dữ liệu | Testid3vs1.py  Dòng 151 |
| 11 | main(): | Thực thi chương trình | Testid3vs1.py  Dòng 161 |

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên lớp | Mục đích | Tên file, stt dòng chứa khai báo |
| 1 | Node() | Tạo cây theo hướng cấu trúc | Testid3vs1.py  Dòng 6 |

1. **Cài đặt và kiểm thử**
   1. **Dataset[3]**

Đây là một bảng dữ liệu được sử dụng rất nhiều trong các bài giảng về decision tree. Bảng dữ liệu này mô tả mối quan hệ giữa thời tiết trong 14 ngày (bốn cột đầu, không tính cột id) và việc một đội bóng có chơi bóng hay không (cột cuối cùng). Nói cách khác, ta phải dự đoán giá trị ở cột cuối cùng nếu biết giá trị của bốn cột còn lại.



* 1. **F1-SCORE**

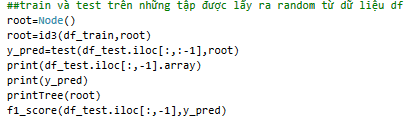
Là hàm số hài hòa của precision và recall

F1 có giá trị từ [0:1], nếu với giá trị dần tiến về 1 thì cho thấy độ chính sách của thuật toán càn lớn, ngược lại dần về 0 thì thuật toán càng không tốt.

Trong đó

* TP là số điểm mà có giá trị dự đoán (positive) giống với giá trị thật sự (positive).
* TN là số điểm mà có giá trị dự đoán (negative) giống với giá trị thực sự (negative).
* FP là số điểm mà có giá trị dự đoán (positive) khác với giá trị thực sự (negative).
* FN là số điểm mà có giá trị dự đoán (negative) khác với giá trị thực sự (positive).
  1. **Các thư viện sử dụng**
* Pandas được sử dụng để thao tác với dữ liệu: cắt, ghép, chọn lọc,… dữ liệu.
* Sklearn: ở đây sử dụng thư viện này để tách dữ liệu thành train và test.
* Math: thêm thư viện này nhằm mục đích phục vụ cho tính toán hàm logarit.
* Collections: sử dụng thư viện này để có được phương thứ deque() dễ dàng in cây.
  1. **Kiểm thử[4]**
* Với tập dữ liệu được tách bởi sklearn.

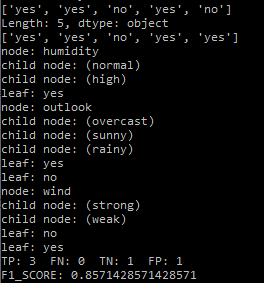
Đầu vào:



Giải thích:

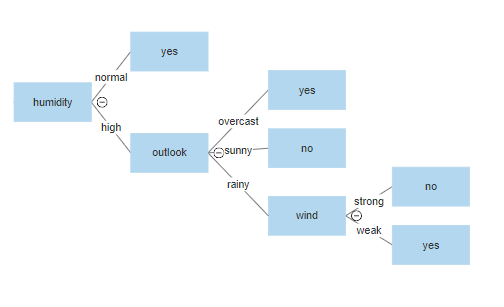
* Khởi tạo root.
* Tính lại root với dữ liệu df\_train.
* Dự đoán kết quả vơi cây.
* In dữ liệu thực tế.
* In dữ liệu dự đoán.
* In cây quyết định.
* Tính f1-score.

Kết quả:



Giải thích:

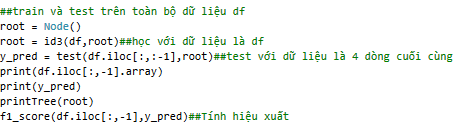
* Dòng đầu tiên là giá trị thực tế.
* Dòng thứ hai là giá trị dự đoán.
* Các dòng tiếp theo là cây quyết định.



Hình này mô tả cho cây ở trên.

* Hai dòng cuối là F1.
  + ở đây f1-score=0.85 bởi vì có sự khác biệt giữa giá trị dự đoán với dữ liệu thực tế, cụ thể: giá trị thứ 5 của mảng giá trị thực tế là ”no” còn dữ liệu dự đoán là “yes”.
* Với tập dữ liệu train và test là toàn bộ dữ liệu.

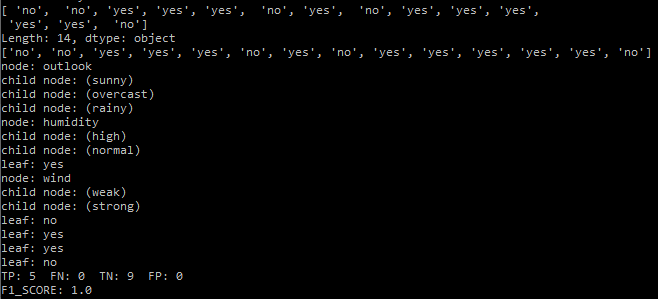
Đầu vào:



Giải thích:

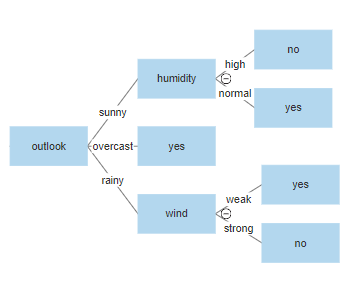
* Khởi tạo root.
* Tính lại root với dữ liệu df\_train.
* Dự đoán kết quả vơi cây.
* In dữ liệu thực tế.
* In dữ liệu dự đoán.
* In cây quyết định.
* Tính f1-score.

Kết quả:



Giải thích:

* Dòng đầu tiên là giá trị thực tế.
* Dòng thứ hai là giá trị dự đoán.
* Các dòng tiếp theo là cây quyết định.



Hình này mô tả cho cây ở trên.

* Hai dòng cuối là F1.
  + ở đây f1-score=1 bởi không có sự sai lệch nào giữa giá trị dự đoán và giá trị thực tế.
  1. **Đánh giá**

Từ giá trị f1-score từ kiểm thử chúng tôi rút ra kết luận:

* Dữ liệu đầu vào của thuật toán này không cần nhiều, mà cần đủ trường hơp (đủ các trường hợp để đưa ra kết luận), và nên phân biệt (cùng điều kiện nhưng khác kết quả), nếu dữ liệu không phân biệt thì nên sử dụng tập data lớn.
* Với việc tính entropy nhiều lần thì việc học tốn khá nhiều thời gian (trên code tự tạo).
* Phân loại không tốt đối với dữ liệu không phân biệt (cùng điều kiện nhưng khác kết quả).

**TÀI LIỆU THAM KHẢO**

1. <https://sinhvientot.net/cay-quyet-dinh-va-giai-thuat-id3/>
2. <https://viblo.asia/p/cay-quyet-dinh-decision-tree-RnB5pXWJ5PG>
3. Ian H. Witten, Eibe Frank, Mark A. Hall, Christopher J. Pal.(2016). Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann
4. <https://1upnote.me/post/2018/10/ds-ml-decision-tree-id3/>