#Viet_AI #AI_Community

CÂY QUYẾT ĐỊNH – DECISION TREE

1. Ý tưởng

Đối với các tổ chức tín dụng, mỗi quyết định cho vay đều được cân nhắc rất kĩ vì nếu việc không thể nhận được số vốn đã cấp xảy ra thì sẽ mang đén rất nhiều rủi ro. Công việc phân tích rủi ro tín này dù không dễ nhưng những khó khăn cũng đã được khắc phục phần nào nhờ vào ứng dụng của các thuật AI. Để mô phỏng lại quá trình này, chúng ta cùng xem xét bộ data bên dưới và cách ứng dụng thuật toán cây quyết định vào việc giải quyết bài toán.

Đây là bộ data về tình hình tài chính của các trường hợp đã cho vay trong quá khứ, bao gồm 8 trường hợp cùng các thuộc tính như: khoản tiết kiệm, tài sản hiện có, thu nhập và biến rủi ro tín dụng, thuộc tính này sẽ là có nếu như trường hợp này nhiệm vụ của bài toán là dựa trên các , hãy xác định khả năng hoàn vốn của những người đi vay tiếp theo qua thuộc tính rủi ro tín dụng của họ.

ID	Khoản tiết kiệm	Tài sản hiện có	Thu nhập ($)	Rủi ro tín dụng
1	Trung bình	Cao	70000	Không
2	Thấp	Thấp	50000	Có
3	Cao	Trung bình	30000	Có
4	Trung bình	Trung bình	50000	Không
5	Thấp	Trung bình	100000	Không
6	Cao	Cao	30000	Không
7	Thấp	Thấp	30000	Có
8	Trung bình	Trung bình	70000	Không

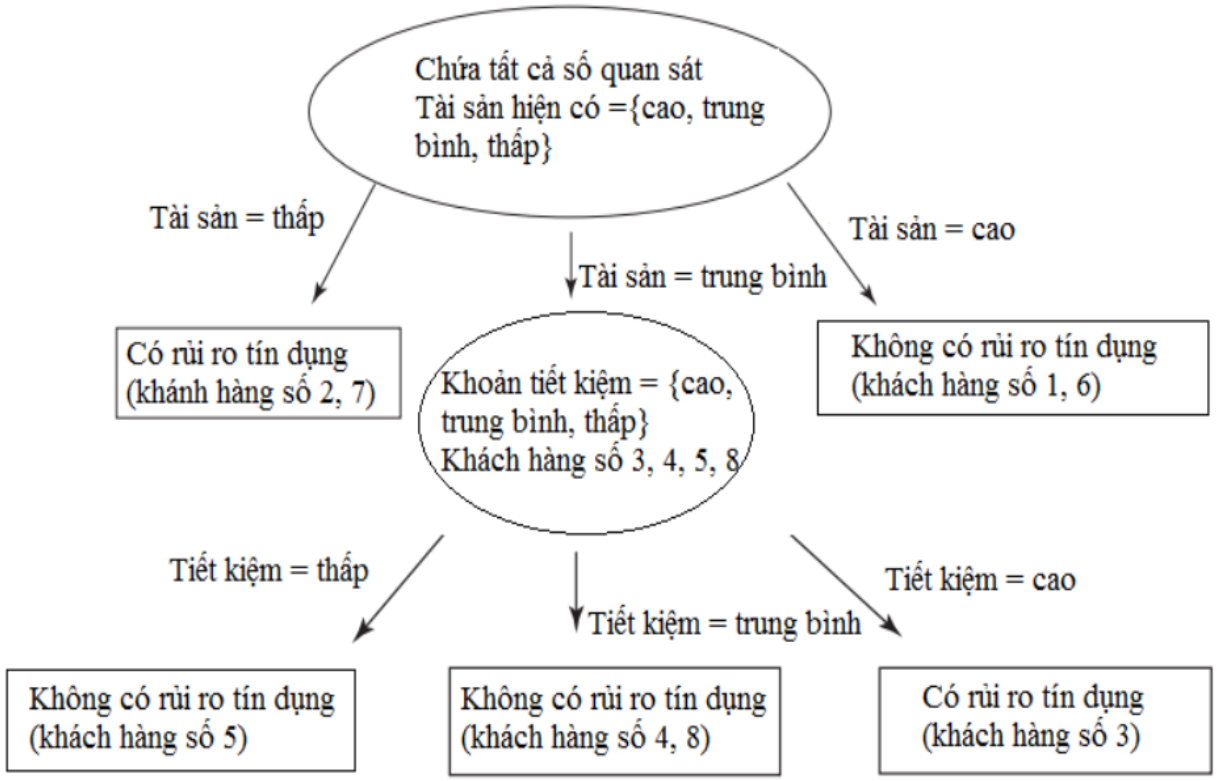
Qua quan sát, chúng ta được những kết quả về những trường hợp có rủi ro tín dụng cao như sau:

Trường hợp	Rủi ro tín dụng
Tài sản hiện có: Thấp	Có
Tài sản hiện có: Cao	Không
Tài sản hiện có: Trung bình, khoản tiết kiệm: Thấp	Không

Tài sản hiện có: Trung bình, khoản tiết kiệm: Trung bình	Không
Tài sản hiện có: Trung bình, khoản tiết kiệm: Cao	Có

Với cách này, ta có thể phân tích rủi ro tín dụng của các trường hợp dựa trên nét tương đồng giữa những đặc điểm của chúng và các trường hợp khác. Nghĩa là ta có thể rút ra các quy luật để dự đoán cho các trường hợp tiếp theo, ví dụ: những trường với tài sản hiện có là thấp thì khả năng cao sẽ có rủi ro tín dụng hay những người có tài sản và khoản tiết kiệm ở mức trung bình thì khả năng cao rủi ro tín dụng sẽ là không.

Đây cũng chính là ý tưởng của thuật toán cây quyết định, với output là một quy luật được biểu diễn như sau:



2. Khái niệm:

Theo wikipedia: "Trong lĩnh vực máy học, **cây quyết định** là một kiểu mô hình dự báo (*predictive model*), nghĩa là một ánh xạ từ các quan sát về một sự vật/hiện tượng tới các kết luận về giá trị mục tiêu của sự vật/hiện tượng."

Hiểu theo cách đơn giản hơn thì **trên thực tế**, v*iệc ra quyết định bị ảnh hưởng bởi cách chúng ta trả lời những câu hỏi liên quan đến vấn đề.* ***Cây quyết định*** *chính là phương pháp* ***mô phỏng lại quá trình đó*** *trên máy tính và tiến hành đưa ra các quyết định, dự đoán phù hợp"*

Với, các thành phần như sau:

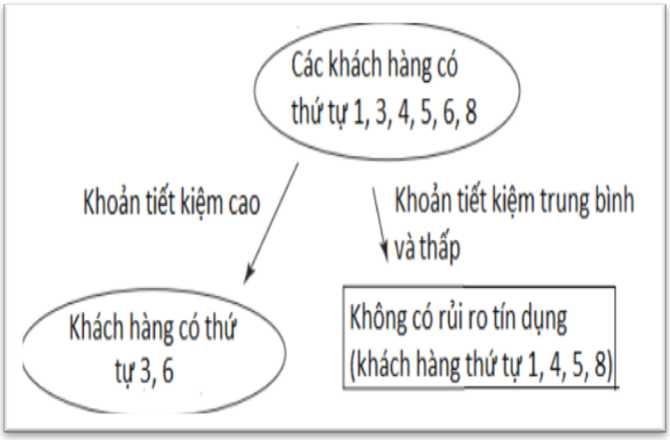
Trên thực tế	Quá trình thực hiện trên máy tính	Trong cây quyết định được gọi là…
các câu hỏi xoay quanh vấn đề	các thuộc tính được xem xét	internal node.
Câu hỏi đầu tiên	Thuộc tính được xem xét đầu tiên	Root node
Đáp án, quyết định cuối cũng	Giá trị dự đoán biến mục tiêu	Leaf node
Thước đo khả năng đáp án, quyết định có độ chính xác cao	Thước đo khả năng thuộc tính có thể phân loại mạnh biến mục tiêu chính xác	Độ tinh khiết (trường hợp ngược lại được gọi là độ vẩn đục)

3. Cách thực hiện:

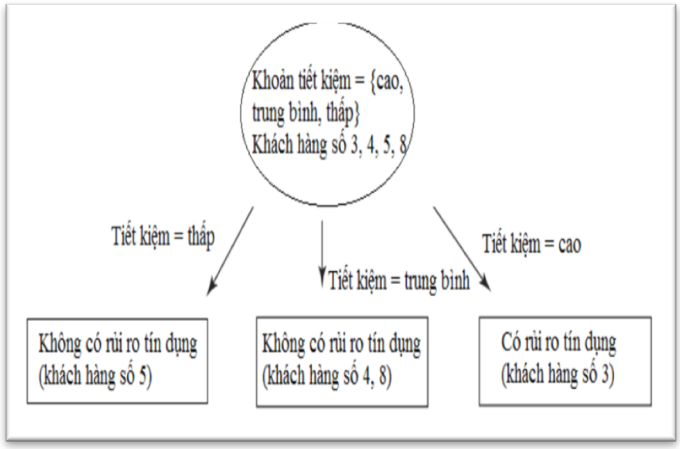
Vậy làm như thế nào mà cây quyết định có thể thực hiện được công việc này, mọi thứ xoay quanh việc chọn thuộc tính và phân nhánh chúng. Việc chia nhánh của cây quyết định gồm 2 loại:

- loại thứ nhất (nổi bật với thuật toán CART) chỉ có thể phân thành 2 nhánh
- loại thứ hai có số nhánh bằng với số lượng các giá trị duy nhất của thuộc tính với các thuật toán như ID 3, C4.5.

Ví dụ với bài toán ban đầu, khi xem xét thuộc tính "khoản tiết kiệm" bằng loại thứ nhất: ta chỉ có thể chia làm 2 nhánh, thấp và trung binh, cao hoặc trung bình và thấp, cao hoặc cao và thấp trung bình; trong khi ở loại thứ 2 ta hoàn toàn có thể chia là 3 nhánh.



*Hình **Error! No text of specified style in document.**2.1: Loại thứ nhất*



Hình 1.1: Loại thứ hai

3.1 Thuật toán CART

Đối với trường hợp thứ nhất, ta xem xét thuật toán CART. Việc nên chọn thuộc tính nào trước, thuộc tính nào sau sẽ có tác động lớn đến kết quả bài toán. Vì chọn đúng một thuộc tính có khả năng phân nhánh mạnh mẽ đồng nghĩa với việc chỉ bằng một bước, ta có thể chia bộ dữ liệu "cồng kềnh" thành những phần "tinh gọn" hơn. Ví dụ với bài toán ban đầu ta đề cập, chỉ cần 2 bước là hoàn thành bài toán nếu ta phân nhánh thuộc tính "tài sản hiện có" trước và thuộc tính "khoản tiết kiệm" sau. Việc hoàn thành sẽ khó khăn và trải qua nhiều bước hơn nếu ta chọn "khoản tiết kiệm" hoặc "thu nhập" làm root node.

Thuật toán này có thể tính toán bằng 2 chỉ số, goodness of best split hoặc GINI. Mặc dù cả 2 chỉ số đều có sự tương đồng và chưa có minh chứng nào về việc chỉ số nào hiệu quả hơn, tuy nhiên, trong bài viết, tôi sẽ chủ yếu đề cập đến GINI vì nó là một trong 2 cách được thư viện sklearn sử dụng.

3.1.1 Ý tưởng thuật toán CART:

Bước 1: Xác định node đầu tiên

+ Tìm tập hợp các trường hợp có thể xảy ra đối với node đầu tiên

+ Xác định trường hợp có chỉ số GINI thấp nhất và lấy đó làm node đầu tiên

Bước 2: Tìm node tiếp theo

+ Loại trường hợp trên

+ Tiếp tục tinh toán tương tự như bước 1 nhưng là đối với tập điểm dữ liệu của từng node

Bước 3: Quay lại bước 2

3.1.2 Chỉ số GINI và ý nghĩa:

Công thức GINI: $GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$ Với, Gini(i) = $GINI(t) = 1 - \sum_{j} [p(j \mid t)]^2$

Không riêng GINI, các chỉ số khác dùng trong thuật toán của cây quyết định như goodness of best split hay entropy,… đều là những chỉ số dùng để biểu trưng cho khả năng phân loại của thuộc tính, nghĩa là chúng biểu trưng cho độ tinh khiết hay vẩn đục của thuộc tính. Các chỉ số hầu như đều hình thành từ 2 giá trị chính: xác suất xảy ra của nhánh và xác suất xảy ra giá trị của biến mục tiêu tại nhánh đó.

3.1.3 Thuật toán

Bước 1:

ID	Khoản tiết kiệm	Tài sản hiện có	Thu nhập ($)	Rủi ro tín dụng
1	Trung bình	Cao	70000	Không
2	Thấp	Thấp	50000	Có
3	Cao	Trung bình	30000	Có
4	Trung bình	Trung bình	50000	Không
5	Thấp	Trung bình	100000	Không
6	Cao	Cao	30000	Không
7	Thấp	Thấp	30000	Có
8	Trung bình	Trung bình	70000	Không

- Tìm tập hợp các trường hợp có thể xảy ra đối với node đầu tiên

Cách phân	Node bên trái	Node bên phải
1	Khoản tiết kiệm = Thấp	Khoản tiết kiệm = {Cao, Trung bình}
2	Khoản tiết kiệm = Trung bình	Khoản tiết kiệm = {Cao, Thấp}
3	Khoản tiết kiệm = Cao	Khoản tiết kiệm = {Thấp, Trung bình}
4	Tài sản hiện có = Thấp	Tài sản hiện có = {Cao, Trung bình}
5	Tài sản hiện có = Trung bình	Tài sản hiện có = {Cao, Thấp}
6	Tài sản hiện có = Cao	Tài sản hiện có = {Thấp, Trung bình}
7	Thu nhập ≤ 30000	Thu nhập ≥ 30000
8	Thu nhập ≤ 50000	Thu nhập ≥ 50000
9	Thu nhập ≤ 70000	Thu nhập ≥ 70000

Trường hợp 1	Rủi ro tín dụng = Có	Rủi ro tín dụng = Không	Tổng
Node trái	2	1	3
Node phải	1	4	5

GINI (khoản tiết kiệm = Thấp) = $1 - [(2/3)^2 + (1/3)^2] = 4/9$

GINI (khoản tiết kiệm = Trung bình, Cao) = $1 - [(1/5)^2 + (4/5)^2] = 8/25$

⇨ GINI (trường hợp 1) = $(3/8) * (4/9) + (5/8) * 8/25 = 11/30 \approx 0.36$

Tương tự, ta có:

GINI (trường hợp 2) ≈ 0.3

GINI (trường hợp 3) ≈ 0.45

GINI (trường hợp 4) ≈ 0.2

GINI (trường hợp 5) ≈ 0.44

GINI (trường hợp 6) ≈ 0.38

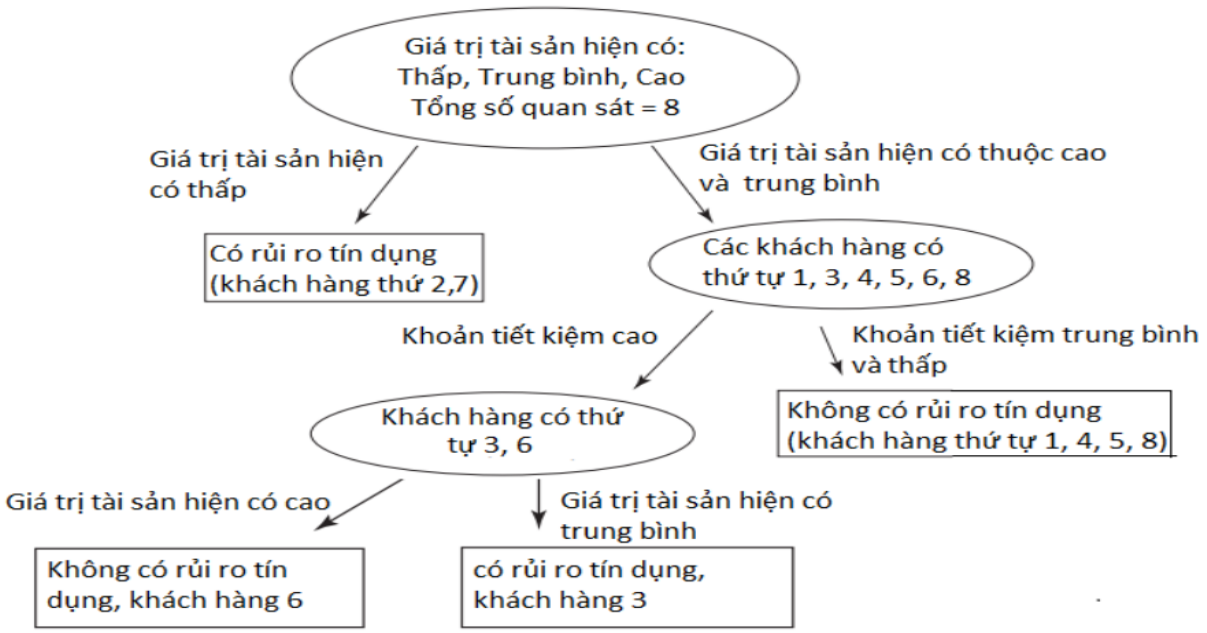
GINI (trường hợp 7) ≈ 0.37

GINI (trường hợp 8) ≈ 0.46

GINI (trường hợp 9) ≈ 0.3

Vì vậy, ta chọn trường hợp thứ tư làm root node

- Tương tự cho bước 2 và ta sẽ có kết quả như sau:



3.2 Thuật toán C4.5

Đối với trường hợp thứ hai, nơi ta có thể phân nhiều nhánh với mỗi nhánh là một giá trị duy nhất của thuộc tính. Ở trường hợp này, ta sẽ xem xét thuật toán C4.5. Thuật toán này sử dụng chỉ Gain Ratio.

3.2.1 Ý tưởng thuật toán:

Bước 1: Xác định node đầu tiên

+ Tìm tập hợp các trường hợp có thể xảy ra đối với node đầu tiên

+ Xác định trường hợp có chỉ số Gain Ratio nhỏ nhất và lấy đó làm node đầu tiên

Bước 2: Tìm node tiếp theo

+ Loại trường hợp trên

+ Tiếp tục tinh toán tương tự như bước 1 nhưng là đối với tập điểm dữ liệu của từng node

Bước 3: Quay lại bước 2

3.2.2 Chỉ số Gain Ratio:

Cách tính Gain Ratio:

1. Tính chênh lệch giữa Entropy của biến mục tiêu và tính Entropy của từng thuộc tính

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

2. Xác định Information Gain của từng thuộc tính:

$$\text{Information Gain:} \quad GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

3. Xác định Split INFO của từng thuộc tính:

$$SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \log_2 \frac{n_i}{n}$$

4. Tính Gain Ratio của từng thuộc tính”

$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO}$$

3.2.3 Thuật toán:

+ Tìm tập hợp các trường hợp có thể xảy ra đối với node đầu tiên

Cách phân nhánh		
Khoản tiết kiệm = Thấp	Khoản tiết kiệm = Trung bình	Khoản tiết kiệm = Cao
Tài sản hiện có = Thấp	Tài sản hiện có = Trung bình	Tài sản hiện có = Cao
Thu nhập ≤ 30000	Thu nhập > 30000	
Thu nhập ≤ 50000	Thu nhập > 50000	
Thu nhập ≤ 70000	Thu nhập > 70000	

+ Xác định trường hợp có chỉ số Gain Ratio nhỏ nhất và lấy đó làm node đầu tiên

Bước 1: Tính Entropy của biến mục tiêu:

$$\text{Entropy(rủi ro tín dụng)} = -5/8*\log_2(5/8) - 3/8*\log_2(3/8) = 0.9544$$

Và Entropy của thuộc tính:

$$\text{Entropy(khoản tiết kiệm = cao)} = -1/2*\log_2(1/2) - 1/2*\log_2(1/2) = 1$$

$$\text{Entropy(khoản tiết kiệm = trung bình)} = -3/3*\log_2(3/3) - 0/3*\log_2(0/3) = 0$$ do 3 khách hàng có khoản tiết kiệm trung bình đều không có rủi ro tín dụng, do đó nhánh này sẽ dẫn đến pure node

$$\text{Entropy(khoản tiết kiệm = thấp)} = -1/3*\log_2(1/3) - 2/3*\log_2(2/3) = 0.9183$$

$$\text{Entropy(khoản tiết kiệm)} = (2/8)*(1) + (3/8)*(0) + (3/8)*(0.9183) = 0.5944$$

Bước 2. Tính Information Gain:

$$\text{Information Gain (khoản tiết kiệm)} = \text{Entropy(rủi ro tín dụng)} - \text{Entropy(khoản tiết kiệm)}$$
$$= 0.9544 - 0.5944 = 0.36$$

Bước 3. Xác định Split INFO:

$$\text{SplitInfo (khoản tiết kiệm)} = -(2/8)*\log_2(2/8) - (3/8)*\log_2(3/8) - (3/8)*\log_2(3/8) = 1.56$$

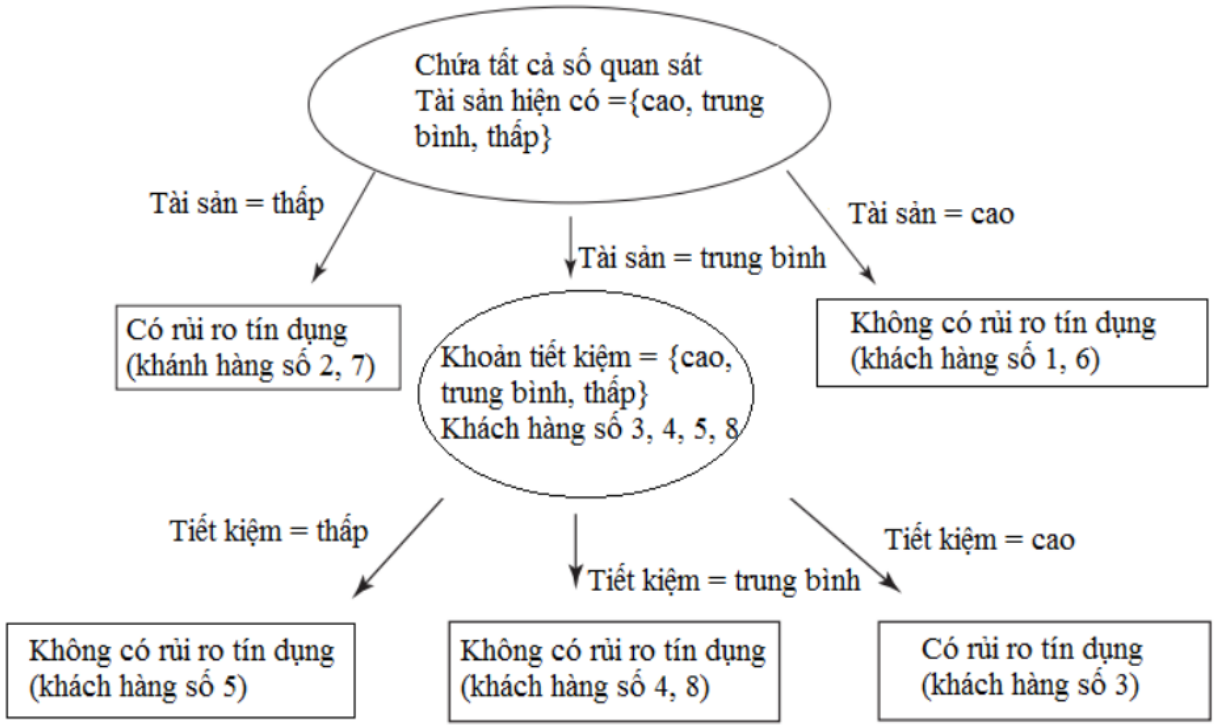
Bước 4. Tính Gain Ratio:

$$\text{GainRatio (khoản tiết kiệm)} = \frac{\text{Information Gain (khoản tiết kiệm)}}{\text{SplitInfo (khoản tiết kiệm)}} = 0.23$$

Tương tự, ta có bảng sau:

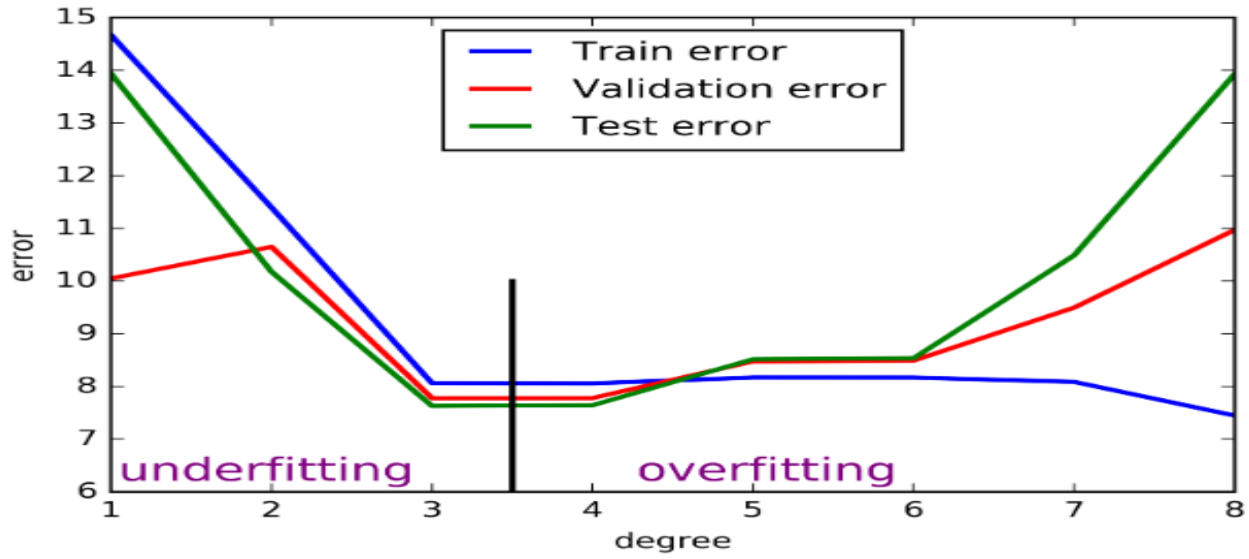
Cách phân	Node phân nhánh	Information Gain	SplitInfo	GainRatio
1	Khoản tiết kiệm = Thấp Khoản tiết kiệm = Trung bình Khoản tiết kiệm = Cao	0.36	1.56	0.231
2	Tài sản hiện có = Thấp Tài sản hiện có = Trung bình Tài sản hiện có = Cao	0.5487	1.5	0.366
3	Thu nhập ≤ 30000 Thu nhập > 30000	0.1588	0.9544	0.166
4	Thu nhập ≤ 50000 Thu nhập > 50000	0.3475	0.9544	0.364
5	Thu nhập ≤ 70000 Thu nhập > 70000	0.0923	0.544	0.170

Nhận thấy Gain Ratio ở cách phân thứ 2 là cao nhất, ta chọn đó làm root node, tương tự như vậy, ta có kết quả sau:

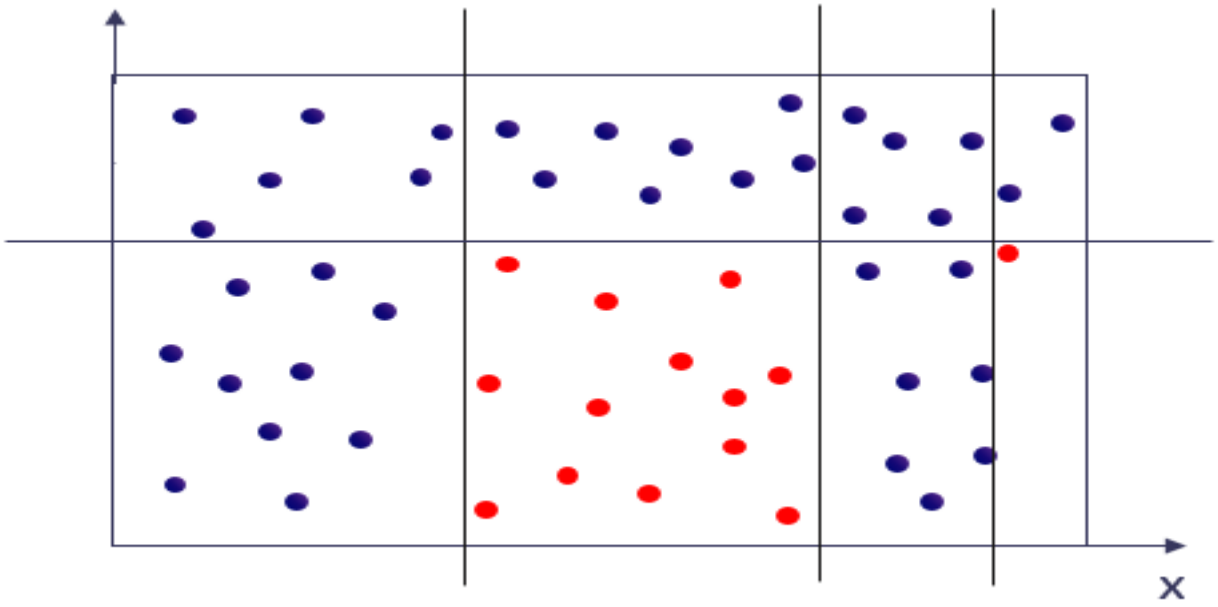


4. **Overfitting trong Cây quyết định > Điều kiện dừng**

Chúng ta đều hình dung được rằng với một mô hình có số bậc càng lớn thì càng dễ dẫn đến overfitting, cây quyết định cũng vậy và số lớp trong cây quyết định cũng tương tự như số bậc của mô hình.



Hình bên dưới là mô phỏng cách phân nhánh của cây quyết định trên bộ dữ liệu. Với 4 đường kẻ chia bộ dữ liệu thành 8 phần. Với 4 đường kẻ chính là 4 lớp của cây quyết định còn 8 phần đó chính 8 leaf node. Như trong hình vẽ, ta có thể có một điều kiện dừng là hạn chế số lớp tối đa là 3 vì lớp thứ 4 phục vụ cho một tập thiểu số các điểm dữ liệu hoặc các điểm noise.



Dưới đây là các điều kiện dừng trong cây quyết định và nếu sử dụng mô hình thông qua sklearn thì chúng sẽ được cài đặt thông qua các hyperparameter:
- nếu mọi điểm trong node đều thuộc một class.
- 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).
- 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.

5. Ưu và nhược điểm của Cây quyết định

Ưu điểm:
- Có thể phù hợp cho cả Phân loại và Hồi quy
- Không cần một lượng dữ liệu quá lớn
- Nâng cao tính khách quan vì ít sử dụng HyperParameter
- Kết quả dễ dàng hiểu và diễn giải

Khuyết điểm
- Chỉ hoạt động tốt trong trường hợp dữ liệu không quá nhiều chiều và các thuộc tính không liên quan với nhau: Vì trường hợp quá nhiều chiều sẽ làm tăng số lớp của cây quyết định hay việc các thuộc tính nếu có liên quan với nhau thì các điểm dữ liệu sẽ tập trung theo đường chéo, điều này cũng làm tăng số lượng lớp của cây quyết định
- Dễ bị thiên vị lớp có chiếm đa số trong biến mục tiêu
- Các bước phân nhánh chỉ được tính theo xác suất xảy ra cao nhất tại nhánh đó chứ không tính đến độ hiệu quả của cả mô hình: điều này dẫn đến sự hình thành của thuật toán Random Forest, bằng cách tạo ra hàng loạt cây quyết định nhằm nâng cao độ khách quan.

6. Tài liệu tham khảo:

[1]. Vũ Khắc Tiệp (2019). *Machine Learning cơ bản*.

[2]. **Bigdatauni.com**. (sd). https://bigdatauni.com/vi/tin-tuc/thuat-toan-cay-quyet-dinh-p-3-c4-5-entropy.html