

Credit Scoring and Decision Trees

Nguyen Hoang Huy

AI Academy Vietnam

October 2, 2020

- 1 Giới thiệu
 - Bài toán chấm điểm tín dụng
 - Tổng quan về chấm điểm tín dụng
- 2 Các phương pháp học máy cho chấm điểm tín dụng
 - Cây quyết định
 - Rừng ngẫu nhiên
 - Hồi quy Logistic
- 3 Kết quả

Bài toán chấm điểm tín dụng

- Dữ liệu chấm điểm tín dụng trong hệ thống Tima Connect:
 - 1 Dữ liệu lấy trực tiếp từ người dùng thông qua App
 - 2 Dữ liệu từ bảo hiểm xã hội
 - 3 Dữ liệu từ thuê bao điện thoại (Telco)
 - 4 Dữ liệu từ Facebook
- Dữ liệu từ App:
 - 1 Họ tên, ngày sinh, giới tính, số chứng minh thư
 - 2 Trạng thái hôn nhân, số con
 - 3 Địa chỉ, tọa độ
 - 4 Nghề nghiệp, lương
 - 5 Khoản vay, thời gian vay
- Dữ liệu bảo hiểm xã hội:
 - 1 Số chứng minh thư
 - 2 Có bảo hiểm xã hội không
 - 3 Số tiền đóng tiền bảo hiểm xã hội trong 12 tháng

Bài toán chấm điểm tín dụng

- Dữ liệu Telco:

1. Họ tên, ngày sinh, giới tính, số chứng minh thư
2. Số điện thoại, loại thuê bao, loại gói cước, gói dữ liệu
3. Số tiền còn lại ở tài khoản chính, khuyến mãi, nội mạng
4. Thông tin về nợ của tài khoản (chỉ với thuê bao trả sau)
5. Dữ liệu tháng (gồm tháng hiện tại và ba tháng trước đó):
6. Số tiền thanh toán ở tài khoản chính, phụ, tổng
7. Lịch sử giao dịch cuộc gọi và tin nhắn: Thời điểm, Số điện thoại người được gọi/nhắn tin, Thời lượng, Cước

- Dữ liệu Facebook:

1. Tên, tuổi, giới tính
2. Trạng thái hôn nhân, gia đình, nghề nghiệp
3. Số bạn bè, tương tác với bạn bè
4. Số ảnh, ngày đăng bài gần nhất, số năm dùng facebook, likes, posts, comments, ...

Một số tập dữ liệu chấm điểm tín dụng khác

- Tập dữ liệu Kaggle:

<https://www.kaggle.com/c/GiveMeSomeCredit>; Biến đầu ra (biến phụ thuộc) “SeriousDlqin2yrs”: đối tượng đã qua 90 ngày quá nợ hoặc nợ xấu; 10 biến dự báo (biến độc lập) bao gồm:

- “Revolving utilization of unsecured Lines”: Tổng số dư trên các thẻ tín dụng và khoản vay tín dụng tối đa không kể bất động sản và nợ trả góp như vay mua ô tô chia cho tổng hạn mức tín dụng và được tính theo %
- “Age”: Tuổi của người vay tính theo năm
- “Number of time 30–59 days past due not worse”: Số lần người vay quá hạn 30-59 ngày nhưng không phải nợ xấu trong 2 năm qua
- “Debt ratio”: Nợ hàng tháng phải trả, tiền trợ cấp, chi phí sinh hoạt chia cho tổng thu nhập tính theo %
- “Monthly income”: Thu nhập hàng tháng
- “Number of open credit lines and loans”: Số lượng các khoản

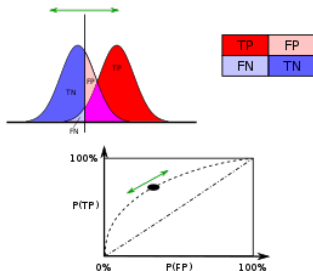
Một số tập dữ liệu chấm điểm tín dụng khác

- “Number of times 90 days late”: Số lần người vay quá hạn 90 ngày hoặc hơn
- “Number of real estate loans or lines”: Số lần vay bất động sản và cầm cố gồm cả các hạn mức tín dụng thế chấp bất động sản (giống như cầm cố lần hai)
- “Number of times 60–89 days past due not worse”: Số lần người vay quá hạn 60-89 ngày nhưng không nợ xấu trong hai năm qua
- “Number of dependents”: Số người phụ thuộc trong gia đình bao gồm cả chính bản thân họ và vợ chồng con cái, ...
- Tập dữ liệu HMEQ (Home Equity Loans):
<http://www.creditriskanalytics.net/datasets-private.html>
- Tập dữ liệu bankloan.sav, trên trang các tập dữ liệu mẫu của IBM:
https://www.ibm.com/support/knowledgecenter/en/SSLVMB_sub/spss/tutorials/data_files.html

Bài toán chấm điểm tín dụng

- Credit scoring problem: maximizing profit, predicting profit
- Assume that net interest is $r = 5\%(/month) = 62.88\%(/year)$, how to calculate the profit? \Rightarrow **focusing on default labels (set = 1)**, see Wang et al. (2015), credit scoring competitions, codes, papers

$$\text{mean profit} = \frac{m}{k} (Acc_1 + (k-1)r \times Acc_0 - 1) = \frac{m}{4} (Acc_1 + 0.15 \times Acc_0 - 1)$$



- $Auc \approx 1 \Rightarrow FP$ smaller, it also means reach more customers

Giới thiệu

- The selected criterion defines the trained classifier
- Almost papers, competitions, banks use: AUC, Gini coefficient, F-score with default label = 1, Average accuracy for each label, see Thomas et al. (2017),
<https://www.kaggle.com/GiveMeSomeCredit>
- The last years have seen the development of many credit scoring models for assessing the credit-worthiness of loan applicants.
- Traditional credit scoring methodology has involved the use of statistical and mathematical programming techniques such as, see Marques et al. (2013).
 - Discriminant analysis, such as Linear discriminant analysis
 - Linear and logistic regression
 - Linear and quadratic programming
 - Decision trees

Giới thiệu

- However, the importance of credit grant decisions for financial institutions has caused growing interest in using a variety of computational intelligence techniques such as
 - Evolutionary algorithms, see Marques et al. (2013)
 - Neural networks, see Blanco et al. (2013), Hussain (2014)
 - Support vector machines, see Baesens et al. (2003), clustered support vector machines, see Terry (2015)
 - Ensemble learning classifiers, such as random forests, see Koutanaei (2015)
 - Regularization techniques such as Lasso logistic regression, LASSO logistic regression ensemble, see Thomas (2011), Wang et.al (2015)
- A very common situation in real-life credit scoring applications is that the data collected are usually highly unbalanced or skewed

Giới thiệu

- Standard learning algorithms are in favor of the majority class, and they will usually show poor performance on the minority class examples
- Compared to a large number of studies in credit scoring, there is relatively only a little research focusing on scoring imbalanced data
- Cost-sensitive learning and re-sampling approaches are two general methods applied to alleviate the class imbalanced problem
- Cost-sensitive learning algorithms which incorporate costs into certain classifier usually assume that
 - The misclassification costs are known beforehand
 - Require special knowledge of the classifiers themselves
- However, these two conditions are not easily satisfied in practice

Giới thiệu

- Hence, most credit scoring algorithms have adopted a re-sampling approach, see Wang (2009)
- Over-sampling outperforms under-sampling in most cases, especially with logistic regression model, see Garcia et al. (2013)
- On the other hand, big data become a commonplace for major credit institutions
- How to exploit efficiently the huge amount data to train a satisfactory model within reasonable time and common computing facilities has become more and more imperative for banks and other credit companies

Cây quyết định

- Dùng cấu trúc cây để đưa ra một hàm phân lớp cần học (hàm mục tiêu có giá trị rời rạc)

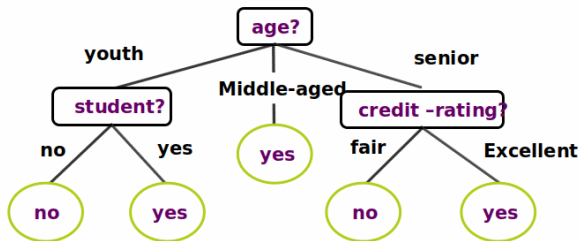
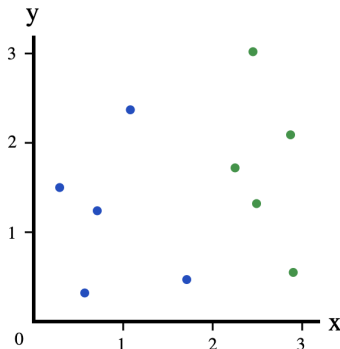


Figure 1: Cây ra quyết định chỉ ra khách hàng nào thường mua máy tính

- Một cây quyết định có thể được biểu diễn (diễn giải) bằng một tập các luật IF-THEN (dễ đọc và dễ hiểu)
- Được áp dụng thành công trong rất nhiều các bài toán ứng dụng thực tế

Ví dụ về cây quyết định

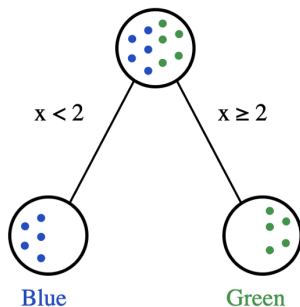
- Cho tập dữ liệu sau:



- Có điểm dữ liệu mới với giá trị thuộc tính $x = 1$, màu của điểm này nên là gì? (nên phân vào lớp nào?)

Ví dụ về cây quyết định

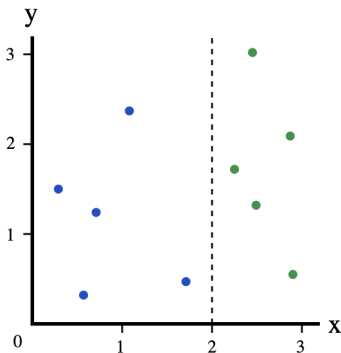
- Đó là cây ra quyết định đơn giản với một node phân loại kiểm tra xem $x < 2$.



- Nếu kiểm tra $x < 2$, chúng ta lấy nhánh trái và gán nhãn màu xanh da trời, nếu kiểm tra không đúng ($x \geq 2$), chúng ta lấy nhánh phải và gán nhãn màu xanh lá cây.

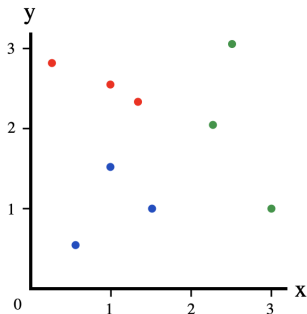
Ví dụ về cây quyết định

- Cây ra quyết định thường được sử dụng để gán nhãn (trong ví dụ này mỗi nhãn của mỗi lớp tương ứng với một màu sắc) mẫu mới dựa trên tập dữ liệu đã có nhãn.



Ví dụ về cây quyết định

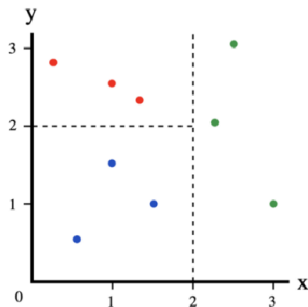
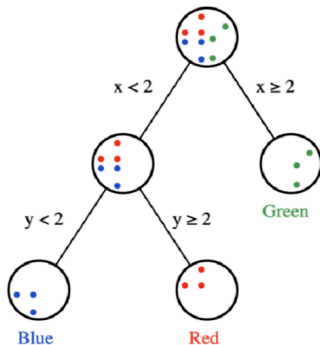
- Bây giờ tập dữ liệu có 3 lớp:



- Cây quyết định cũ không hiệu quả, với mẫu dữ liệu mới (x, y)
 - Nếu $x \geq 2$, chúng ta có thể vẫn tự tin phân loại vào lớp xanh lá cây
 - Nếu $x < 2$, chúng ta không thể phân loại ngay vào lớp xanh ra trời, nó cũng có thể đỏ.

Ví dụ về cây quyết định

- Chúng ta cần thêm node quyết định vào cây quyết định



- Đó là ý tưởng chính của cây ra quyết định.

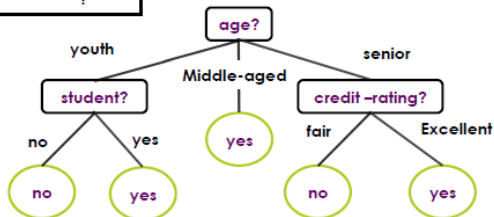
Biểu diễn cây quyết định

- Mỗi nút trong (internal node) biểu diễn một thuộc tính cần kiểm tra giá trị đối với các ví dụ (mẫu).
- Mỗi nhánh (branch) từ một nút sẽ tương ứng với một giá trị có thể của thuộc tính gắn với nút đó.
- Mỗi nút lá (leaf node) biểu diễn một lớp.
- Một cây quyết định học được sẽ phân lớp đối với một ví dụ, bằng cách duyệt cây từ nút gốc đến một nút lá
- → Nhãn lớp gắn với nút lá đó sẽ được gán cho ví dụ cần phân lớp

Ví dụ về cây quyết định

RID	age	income	student	credit-rating	Class
1	youth	high	no	fair	?

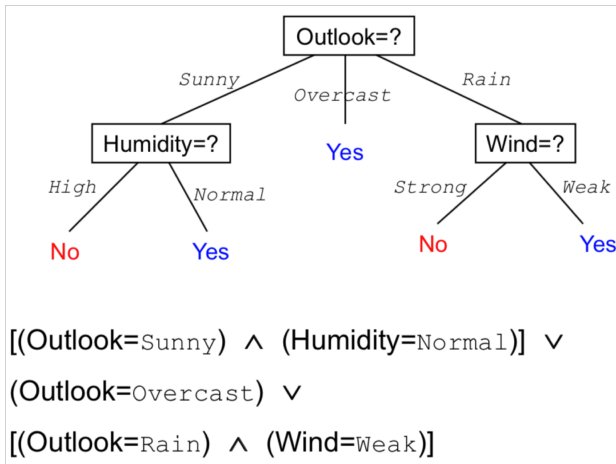
- Test on age: youth
- Test of student: no
- Reach leaf node
- Class NO:** the customer is unlikely to buy a computer



A decision tree indicating whether a customer is likely to purchase a computer

- Mỗi đường đi (path) từ nút gốc đến một nút lá sẽ tương ứng với một kết hợp (conjunction) của các kiểm tra giá trị thuộc tính (attribute tests).
- Cây quyết định (bản thân nó) chính là một phép tuyển (disjunction) của các kết hợp (conjunctions) này.

Một người có chơi tennis không?



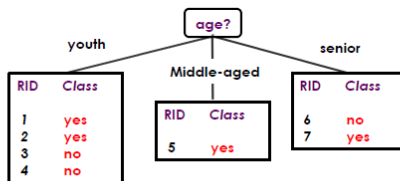
Học cây quyết định bằng ID3

- ID3 (Iterative Dichotomiser 3) thực hiện tìm kiếm tham lam trên không gian các cây quyết định (do Ross Quinlan đề xuất năm 1986).
- Giả thuyết mỗi quan sát x được biểu diễn bởi d thuộc tính
 - $\mathbf{x} = (x_1, x_2, \dots, x_d)$
 - Mỗi x_i là một thuộc tính rời rạc (discrete) hoặc định danh (categorical)
- Mỗi quan sát trong tập học có một nhãn lớp tương ứng.
- Xây dựng (học) một cây quyết định một cách quy nạp, theo chiến lược top-down, bắt đầu từ nút gốc ứng với tất cả tập học.

Học cây quyết định bằng ID3

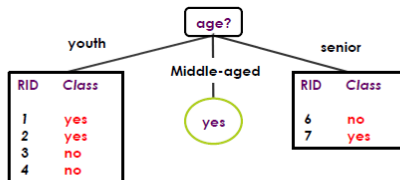
- Ở mỗi nút tiếp theo, chọn thuộc tính kiểm tra (là thuộc tính có khả năng phân loại tốt nhất đối với các ví dụ học gắn với nút đó).
- Tạo mới một cây con của nút hiện tại cho mỗi giá trị của thuộc tính kiểm tra, và tập học sẽ được tách ra (thành các tập con) tương ứng với cây con vừa tạo.
- Quá trình phát triển cây quyết định sẽ tiếp tục cho đến khi:
 - Tất cả các mẫu hoặc thuộc tính đã được sử dụng và biểu quyết theo số đông được sử dụng để gán node lá
 - Tất cả các mẫu thuộc cùng một lớp
- Chú ý: Mỗi thuộc tính chỉ được phép xuất hiện tối đa 1 lần đối với bất kỳ một đường đi nào trong cây.

Ví dụ



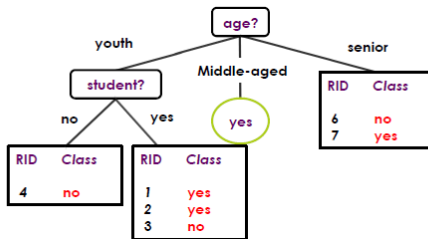
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

Ví dụ



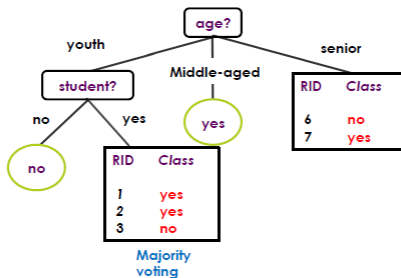
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

Ví dụ



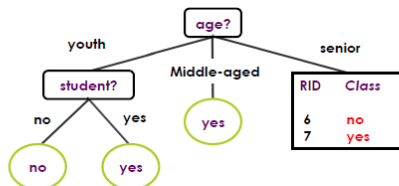
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

Ví dụ



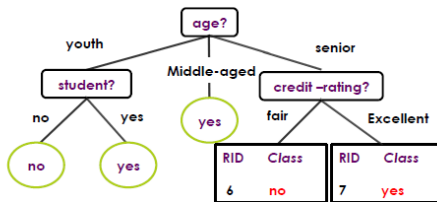
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

Ví dụ



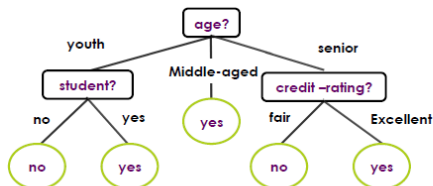
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

Ví dụ



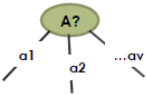
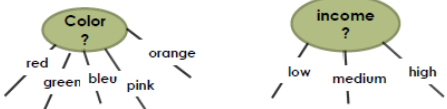
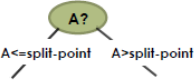
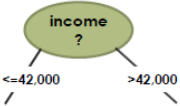

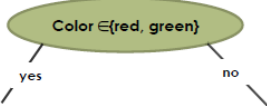
RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes

Ví dụ



RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	yes
7	senior	yes	excellent	no

Ba kịch bản phân chia giá trị thuộc tính

Partitioning scenarios	Examples
Discrete-valued 	
Continuous-valued 	
Discrete-valued + binary tree 	

Các độ đo lựa chọn thuộc tính

- Một độ đo lựa chọn thuộc tính là một phương pháp tiên nghiệm (heuristic) để lựa chọn tiêu chí phân chia để phân tách tốt nhất phần dữ liệu **D** đã cho
- Một cách lý tưởng
 - Mỗi phần được chia ra nên thuần nhất
 - Mỗi phần thuần nhất là phần chứa các mẫu cùng thuộc một lớp
- Các độ đo phân chia thuộc tính (các luật phân chia)
 - Xác định các mẫu ở một node được phân chia thế nào
 - Đưa ra cách xếp hạng các thuộc tính
 - Thuộc tính với điểm cao nhất được lựa chọn
 - Xác định một điểm phân chia hoặc một tập con phân chia
- Các phương pháp
 - Information gain
 - Gain ratio
 - Gini Index

Entropy

- Trong cả hai bức ảnh A và B, đứa trẻ đều ăn súp

Low Entropy



A

High Entropy



B

- Trong tình huống nào (A hay B) có entropy liên quan đến vị trí của súp cao/thấp hơn (theo

Hướng tiếp cận Information Gain

- **D**: Phần dữ liệu hiện tại
- **N**: Biểu diễn số mẫu của **D**
- Lựa chọn thuộc tính với Information Gain cao nhất (dựa trên bài báo của Shannon về lý thuyết thông tin)
- Thuộc tính này
 - Tối thiểu thông tin cần thiết để phân loại các mẫu trong các phần được chia
 - Phản ánh mức độ thuần nhất hoặc độ ngẫu nhiên ít nhất trong những phân chia này
- Hướng tiếp cận Information Gain tối thiểu số lần kiểm tra cần thiết trung bình để phân loại một mẫu đã cho và đảm bảo cây phân loại đơn giản nhất

Bước 1

- Tính thông tin trung bình (Entropy) cần thiết để phân loại một mẫu trong phần dữ liệu **D**

$$Info(\mathbf{D}) = - \sum_{i=1}^m p_i \log_2(p_i)$$

- m : Số lớp
- p_i : Xác suất một mẫu bất kỳ trong **D** thuộc vào lớp C_i được ước lượng bởi $|C_i, \mathbf{D}|/|\mathbf{D}|$ (tỉ lệ số mẫu C_i trong **D**)
- Hàm log lấy theo cơ số 2 bởi vì thông tin được mã hóa thành các bit
- $Info\mathbf{D}$
 - Lượng thông tin trung bình cần để xác định nhãn của mỗi mẫu trong **D**
 - Đó là **entropy**

Ví dụ

RID	age	income	student	credit-rating	class:buy_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle-aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle-aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle-aged	medium	no	excellent	yes
13	middle-aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

In partition D

m=2 (the number of classes)

9 tuples in class yes

N= 14 (number of tuples)

5 tuples in class no

$$Info(D) = -\frac{9}{14} \log_2\left(\frac{9}{14}\right) - \frac{5}{14} \log_2\left(\frac{5}{14}\right) = 0.940 \text{ bits}$$

Bước 2

- Với mỗi thuộc tính, tính lượng thông tin cần để đưa ra phân loại chính xác sau khi phân chia sử dụng thuộc tính đó
- Giả sử phân chia các mẫu trong \mathbf{D} dựa vào thuộc tính \mathbf{A} có các giá trị tương ứng $\{a_1, \dots, a_v\}$
 - Chia \mathbf{D} thành v phần $\mathbf{D}_1, \mathbf{D}_2, \dots, \mathbf{D}_v$
 - Một cách lý tưởng phần \mathbf{D}_i thuần nhất nhưng ít khi xảy ra điều đó
- Lượng thông tin cần để phân loại chính xác được đo bởi:

$$Info_{\mathbf{A}}(\mathbf{D}) = \sum_{j=1}^v \frac{|\mathbf{D}_j|}{|\mathbf{D}|} \times Info(\mathbf{D}_j)$$

- $\frac{|\mathbf{D}_j|}{|\mathbf{D}|}$: tỉ lệ mẫu của phần \mathbf{D}_j trong \mathbf{D}
- $Info(\mathbf{D}_j)$: Entropy của phần \mathbf{D}_j
- Thông tin trung bình đạt được càng nhỏ, độ thuần nhất của

Ví dụ

RID	age	income	student	credit-rating	class:buy_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle-aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle-aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle-aged	medium	no	excellent	yes
13	middle-aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Using attribute age

Part1(youth) **D1** has **2 yes** and **3 no**

Part2(middle-aged) **D2** has **4 yes** and **0 no**

Part3(senior) **D3** has **3 yes** and **2 no**

$$Info_{age}(D) = \frac{5}{14} Info(D_1) + \frac{4}{14} Info(D_2) + \frac{5}{14} Info(D_3) = 0.694$$

Bước 3

- Tính Information Gain
- Information Gain phân chia bởi thuộc tính **A**

$$Gain(\mathbf{A}) = Info(\mathbf{D}) - Info_{\mathbf{A}}(\mathbf{D})$$

- Information Gain là độ giảm trung bình trong thông tin cần thiết gây ra bởi biết giá trị của **A**
- Thuộc tính **A** với Information Gain cao nhất ($Gain(\mathbf{A})$), được lựa chọn như là thuộc tính phân chia ở node N

Ví dụ

RID	age	income	student	credit-rating	class:buy_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle-aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle-aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle-aged	medium	no	excellent	yes
13	middle-aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

$$\text{Gain}(\text{age}) = \text{Info}(D) - \text{Info}_{\text{age}}(D) = 0.246$$

Gain(income)=0.029,

Gain(student)=0.151

Gain(credit_rating)=0.048

“Age” has the highest gain \Rightarrow It is chosen as the splitting attribute

Lưu ý về các thuộc tính giá trị liên tục

- Cho **A** là một thuộc tính giá trị liên tục
- Phải xác định điểm phân chia tốt nhất cho **A**
 - Sắp xếp các giá trị của **A** theo thứ tự tăng dần
 - Thông thường, các điểm giữa mỗi cặp giá trị liên tiếp được xem là các điểm chia, $(a_i + a_{i+1})/2$ là điểm giữa a_i và a_{i+1}
 - điểm với trung bình giá trị thông tin cần thiết nhỏ nhất cho **A** được lựa chọn là điểm chia
- Phân chia
 - D_1 là tập các mẫu trong **D** thỏa mãn $A \leq \text{split} - \text{point}$
 - D_2 là tập các mẫu trong **D** thỏa mãn $A > \text{split} - \text{point}$

Hướng tiếp cận Gain Ratio

- Vấn đề của Information Gain
 - Có xu hướng kiểm tra những thuộc tính có nhiều giá trị
 - Ví dụ thuộc tính xác định duy nhất cho mỗi mẫu
 - Sinh ra một số lượng lớn các phần (một mẫu một phần)
 - Mỗi phần được chia \mathbf{D}_i là thuần nhất $Info(\mathbf{D}_i) = 0$
 - Information Gain được cực đại
- Mở rộng Information Gain
 - Sử dụng **gain ratio**
 - Vượt qua sự thiên lệch của Information gain
 - Áp dụng một kiểu chuẩn hóa đối với Information gain sử dụng giá trị **split information**

Split Information

- Giá trị **split information** biểu diễn thông tin tiềm năng sinh bởi sự phân chia tập dữ liệu huấn luyện **D** thành v phần, tương ứng với v giá trị của thuộc tính **A**

$$SplitInfo_{\mathbf{A}}(\mathbf{D}) = - \sum_{j=1}^v \frac{|\mathbf{D}_j|}{|\mathbf{D}|} \times \log_2 \left(\frac{|\mathbf{D}_j|}{|\mathbf{D}|} \right)$$

- Split Info cao: Các phần kích thước đồng đều
- Split Info thấp: Một vài phần chứa phần lớn các mẫu
- Gain ratio xác định bởi:

$$GainRatio(\mathbf{A}) = \frac{Gain(\mathbf{A})}{SplitInfo(\mathbf{A})}$$

- Thuộc tính với Gain ratio lớn nhất được lựa chọn để phân chia

Ví dụ

RID	age	income	student	credit-rating	class:buy_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle-aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle-aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle-aged	medium	no	excellent	yes
13	middle-aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Using attribute income

Part1 (low) : **4 tuples** , Part2 (medium): **6 tuples**, Part3 (high): **4 tuples**

$$SplitInfo_{income}(D) = -\frac{4}{14}\log_2\left(\frac{4}{14}\right) - \frac{6}{14}\log_2\left(\frac{6}{14}\right) - \frac{4}{14}\log_2\left(\frac{4}{14}\right) = 0.926$$

$$Gain(income) = 0.029$$

$$GainRatio(income) = \frac{0.029}{0.926} = 0.031$$

Hướng tiếp cận Gini Index

- Độ đo sự không thuần nhất của phần dữ liệu \mathbf{D}

$$Gini(\mathbf{D}) = 1 - \sum_{i=1}^m p_i^2$$

- m : Số lớp
- p_i : Xác suất một mẫu trong \mathbf{D} thuộc vào lớp C_i
- Gini Index xét một phân chia nhị phân cho mỗi thuộc tính \mathbf{A} , được \mathbf{D}_1 và \mathbf{D}_2 . Gini index của \mathbf{D} cho bởi phân chia đó là:

$$Gini_{\mathbf{A}}(\mathbf{D}) = \frac{|\mathbf{D}_1|}{|\mathbf{D}|} Gini(\mathbf{D}_1) + \frac{|\mathbf{D}_2|}{|\mathbf{D}|} Gini(\mathbf{D}_2)$$

- Độ giảm sự không thuần nhất cho bởi:

$$\Delta Gini(\mathbf{A}) = Gini(\mathbf{D}) - Gini_{\mathbf{A}}(\mathbf{D})$$

- Thuộc tính tối đại độ giảm sự không thuần nhất được lựa chọn

Phân chia nhị phân

- Các thuộc tính liên tục

- Kiểm tra mỗi điểm phân chia có thể. Điểm chính giữa mỗi cặp giá trị liên tiếp (đã được sắp xếp) được lấy như một điểm phân chia có thể
- Với mỗi điểm phân chia, tính tổng có trọng số của mỗi kết quả phân chia (thành 2 phần:

$$\mathbf{D}_1 : \mathbf{A} \leq \text{split} - \text{point}, \mathbf{D}_2 : \mathbf{A} > \text{split} - \text{point}$$

- Điểm cho Gini Index lớn nhất đối với thuộc tính \mathbf{A} được lựa chọn như điểm được phân chia

- Các thuộc tính rời rạc

- Kiểm tra các phần có thể được chia từ tất cả các tập con có thể của $\{a_1, \dots, a_v\}$
- Mỗi tập con $S_{\mathbf{A}}$ là một kiểm tra nhị phân của thuộc tính \mathbf{A} theo công thức " $\mathbf{A} \in S_{\mathbf{A}}?$ "
- 2^v tập con có thể, loại trừ tập rỗng và tập lũy thừa, chúng ta có $2^v - 2$ tập con

Ví dụ

RID	age	income	student	credit-rating	class:buy_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle-aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle-aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle-aged	medium	no	excellent	yes
13	middle-aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Compute the Gini index of the training set D: 9 tuples in class yes and 5 in class no

$$Gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

Using attribute income: there are three values: **low**, **medium** and **high**

Choosing the subset {**low**, **medium**} results in two partitions:

D1 (income ∈ {low, medium}): 10 tuples

D2 (income ∈ {high}): 4 tuples

Ví dụ

$$\begin{aligned}
 Gini_{income \in \{low, median\}}(D) &= \frac{10}{14} Gini(D_1) + \frac{4}{14} Gini(D_2) \\
 &= \frac{10}{14} \left(1 - \left(\frac{6}{10} \right)^2 - \left(\frac{4}{10} \right)^2 \right) + \frac{4}{14} \left(1 - \left(\frac{1}{4} \right)^2 - \left(\frac{3}{4} \right)^2 \right) \\
 &= 0.450 \\
 &= Gini_{income \in \{high\}}(D)
 \end{aligned}$$

The Gini Index measures of the remaining partitions are:

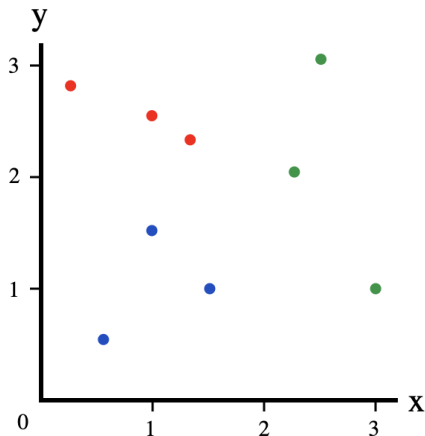
$$Gini_{\{low, high\} \text{ and } \{medium\}}(D) = 0.315$$

$$Gini_{\{medium, high\} \text{ and } \{low\}}(D) = 0.300$$

The best binary split for attribute income is on **{medium, high}** and **{low}**

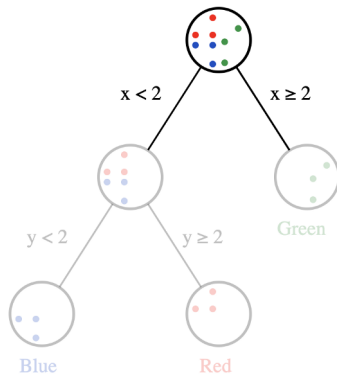
Ví dụ (tiếp)

- Trong ví dụ tiếp theo này chúng ta sử dụng lại tập dữ liệu 3 lớp



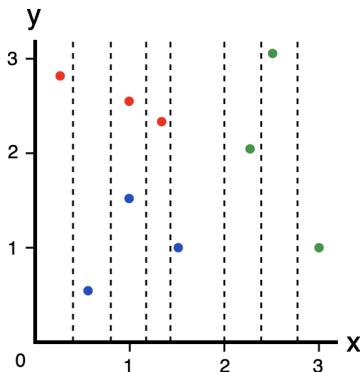
Ví dụ (tiếp)

- Nhiệm vụ đầu tiên là xác định node gốc, thuộc tính x hay y sẽ được lựa chọn, ngưỡng kiểm tra là gì sao cho phân tách các lớp khác nhau càng nhiều càng tốt





Ví dụ (tiếp)

- Thử phân chia với mọi thuộc tính x hoặc y
- Thử với mọi ngưỡng phân chia sao (đây không phải là cách tìm phân chi tốt nhất một cách nhanh nhất)



Ví dụ (tiếp)

- Thử tính Gini index với ngưỡng chia $x = 4$

Split	Left Branch	Right Branch
$x = 0.4$		

- Đầu tiên tính Gini index với toàn tập huấn luyện

$$\begin{aligned}
 G_{initial} &= \sum_{i=1}^3 p(i) * (1 - p(i)) \\
 &= 3 * \left(\frac{1}{3} * \frac{2}{3} \right) \\
 &= \boxed{\frac{2}{3}}
 \end{aligned}$$

Ví dụ (tiếp)

- Tính Gini Index cho hai nhánh

$$G_{left} = 0 * 1 + 1 * 0 + 0 * 1 = \boxed{0}$$

$$\begin{aligned} G_{right} &= \frac{3}{8} * \frac{5}{8} + \frac{2}{8} * \frac{6}{8} + \frac{3}{8} * \frac{5}{8} \\ &= \boxed{\frac{21}{32}} \end{aligned}$$

- Cuối cùng tính Gini Gain bằng cách trừ đi tổng có trọng số của Gini Index các nhánh khỏi Gini Index của toàn tập huấn luyện

$$\begin{aligned} \text{Gain} &= G_{initial} - \frac{1}{9}G_{left} - \frac{8}{9}G_{right} \\ &= \frac{2}{3} - \frac{1}{9} * 0 - \frac{8}{9} * \frac{21}{32} \\ &= \boxed{0.083} \end{aligned}$$

Ví dụ (tiếp)

- Tính Gini Gain tương tự như trên cho mọi điểm phân chia có thể

Split	Left Branch	Right Branch	Gini Gain
$x = 0.4$	•	••••••••	0.083
$x = 0.8$	••	••••••••	0.048
$x = 1.1$	••••	••••••	0.133
$x = 1.3$	•••••	•••••	0.233
$x = 2$	••••••	••••	0.333
$x = 2.4$	•••••••	••	0.191
$x = 2.8$	••••••••	•	0.083
$y = 0.8$	•	••••••••	0.083
$y = 1.2$	•••	•••••••	0.111
$y = 1.8$	••••	••••••	0.233
$y = 2.1$	•••••	•••••	0.233
$y = 2.4$	••••••	••••	0.111
$y = 2.7$	•••••••	••	0.048
$y = 2.9$	••••••••	•	0.083

Ví dụ (tiếp)

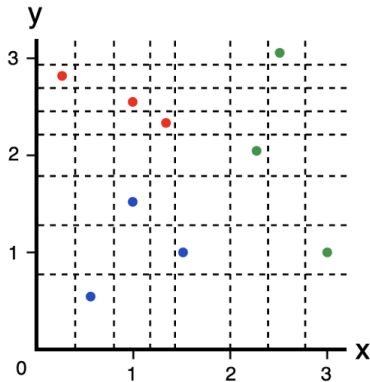
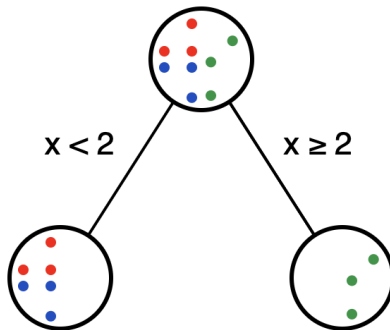


Figure 2: All thresholds

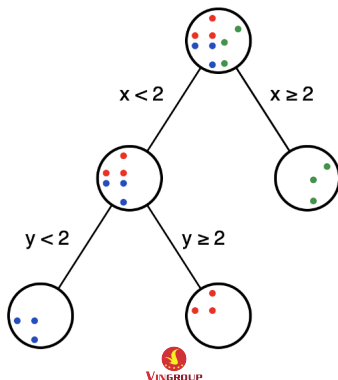
Ví dụ (tiếp)

- Sau khi thử với tất cả các ngưỡng của x và y , thấy rằng phân chia bởi $x = 2$ có Gini gain cao nhất
- Do đó lựa chọn node gốc sử dụng thuộc tính x với ngưỡng bằng 2



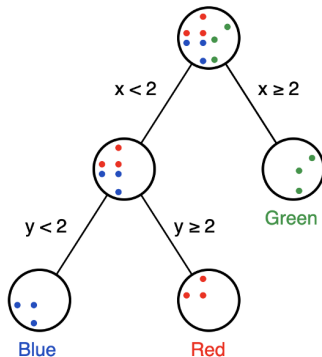
Ví dụ (tiếp)

- Chỉ sử dụng điểm dữ liệu ở nhánh trái ($x < 2$) để xây dựng node quyết định ở nhánh này
- Thử tất cả phân chia có thể và được $y = 2$ là điểm phân chia tốt nhất



Ví dụ (tiếp)

- Khi nào dừng huấn luyện cây quyết định?



So sánh các đô đo lựa chọn thuộc tính

- Informaiton Gain

- Lựa chọn thiên vị những thuộc tính nhiều giá trị

- Gain Ratio

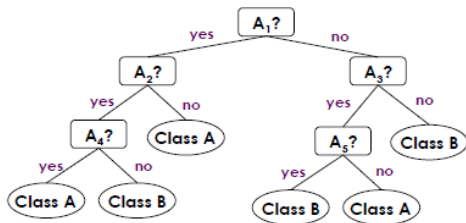
- Lựa chọn thiên vị những phân chia không cân bằng trong đó một phần dữ liệu được phân chia nhỏ hơn rất nhiều so với các phần khác

- Gini Index

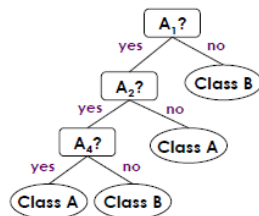
- Hướng đến những thuộc tính nhiều giá trị
 - Gặp khó khăn khi số lớp lớn
 - Hướng đến kiểm tra các phần có kích thước đều nhau và thuần nhất trong mỗi phần

Học quá

- Nhiều nhánh của cây quyết định phản ánh sự bất thường trong dữ liệu học (có thể do nhiễu hoặc giá trị ngoại lai)
- Độ chính xác thấp đối với các mẫu mới (kiểm tra)
- Giải pháp: **Cắt tỉa**
 - Loại bỏ những nhánh ít tin cậy



Before Pruning



After Pruning

Chiến thuật cắt tỉa

- Tiền cắt tỉa
 - Dừng sớm quá trình xây dựng cây, nghĩa là không phân chia một node nếu độ tốt dưới ngưỡng nào đó
 - Sự khác biệt thống kê, Information gain, Gini index được sử dụng để đánh giá độ tốt của một phân chia
 - Khi dừng thì node đó trở thành một lá và có thể nhận nhãn lớp có tần suất xuất hiện nhiều nhất trong tập con dữ liệu đó.
- Hậu cắt tỉa
 - Loại bỏ các nhánh của cây “phát triển đầy đủ”
 - Một cây con ở node cắt tỉa được thay bằng node lá
 - Node lá được gán nhãn ứng với lớp thường xuyên nhất

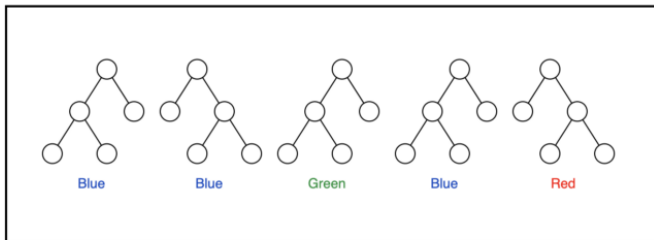
Tóm tắt cây quyết định

- Cây quyết định có tốc độ học tương đối nhanh so với các phương pháp khác
- Đơn giản và dễ hiểu các luật phân loại trong cây ra quyết định
- Information Gain, Gain Ratio, và Gini Index là những phương pháp lựa chọn thuộc tính thông dụng nhất
- Cắt tỉa cây là cần thiết để loại bỏ những nhánh không tin cậy

Bagging

- Xem xét thuật toán sau để huấn luyện một tập cây ra quyết định trên một tập n mẫu dữ liệu
 - Lấy n mẫu học có thay thế từ tập dữ liệu học đã cho
 - Huấn luyện cây ra quyết định trên n mẫu vừa lấy
 - Thực hiện lại t lần
- Để đưa ra dự đoán dựa vào mô hình t cây quyết định này, chúng ta kết hợp các dự đoán riêng rẽ của từng cây
 - Biểu quyết theo số đông nếu các cây cho nhãn (màu sắc)
 - Lấy trung bình nếu các cây cho giá trị số (khi hồi quy dự báo nhiệt độ, giá cả, ...)
- Kỹ thuật này được gọi là **bagging**, hay **bootstrap aggregating**
- Các mẫu được lấy mẫu có thay thế gọi là các mẫu **bootstrap**

Bagging



Blue

- Bagging các cây quyết định rất gần với Rừng ngẫu nhiên (Random forests), chỉ thiếu một điều

Bagging → Random Forest

- Bagging cây quyết định chỉ có tham số t , số cây quyết định
- Rừng ngẫu nhiên có tham số thứ hai điều kiện **số thuộc tính sử dụng để tìm phân chia tốt nhất**
- Nghĩa là thay vì sử dụng tất cả thuộc tính mỗi lần, chúng ta chỉ sử dụng tập con thuộc tính
- Để tăng tính ngẫu nhiên, để mỗi cây là duy nhất và **giảm sự tương quan giữa các cây**, làm tăng hiệu năng của Rừng ngẫu nhiên
- Kỹ thuật này gọi là **feature bagging**

Hồi quy Logistic

- Cho tập dữ liệu tín dụng (\mathbf{x}^i, y_i) , $i \in 1, 2, \dots, n$, trong đó $\mathbf{x}^i = (x_{i1}, x_{i2}, \dots, x_{id})^T$ ứng với các biến quan sát, y_i là nhãn lớp (1 cho nợ xấu, và 0 nợ tốt)
- $\text{Prob}(y|\mathbf{x})$ là xác suất một ứng viên thuộc vào lớp y (0 hoặc 1), trong điều kiện quan sát \mathbf{x}

$$\log \frac{\text{Prob}(y = 1|\mathbf{x})}{\text{Prob}(y = 0|\mathbf{x})} = w_0 + \mathbf{x}^T \mathbf{w} \quad (1)$$

trong đó w_0 là ngưỡng, $\mathbf{w} = (w_1, \dots, w_d)^T$ là các hệ số tuyến tính

$$\iff \text{Prob}(y = 1|\mathbf{x}) = h(w_0 + \mathbf{x}^T \mathbf{w})$$

trong đó $h(x)$ là hàm logistic

Hồi quy Logistic

- Hướng tiếp cận hợp lý cực đại (maximum likelihood) thường được sử dụng để tính các hệ số \mathbf{w} và log-likelihood được viết thành:

$$\begin{aligned}\ell(\mathbf{w}_0, \mathbf{w}) &= \sum_{i=1}^n \{y_i \log \text{Prob}(Y = 1; \mathbf{w}) \\ &\quad + (1 - y_i) \log(1 - \text{Prob}(Y = 1; \mathbf{w}))\} \\ &= \sum_{i=1}^n y_i (\mathbf{w}_0 + \mathbf{x}_i^T \mathbf{w}) - \log(1 + \exp^{\mathbf{w}_0 + \mathbf{x}_i^T \mathbf{w}}) \quad (2)\end{aligned}$$

- Mô hình hồi quy Logistic trên có thể mở rộng thành mô hình hồi quy Logistic Lasso bằng cách đặt thêm điều kiện ℓ_1 đối với tham số \mathbf{w}

Hồi quy Logistic Lasso

- Bài toán tìm hàm hồi quy Logistic Lasso trở thành bài toán tối tiểu hàm đối của hàm log-likelihood cộng với đại lượng phạt

$$\sum_{i=1}^n \{\log(1 + \exp^{w_0 + \mathbf{x}_i^T \mathbf{w}}) - y_i(w_0 + \mathbf{x}_i^T \mathbf{w})\} + \lambda \sum_{j=1}^d |w_j| \quad (3)$$

- Đến giờ, chưa có báo cáo nghiên cứu nào về sử dụng các mô hình chỉnh hoá mới nhất như Lasso để làm thuật toán học cơ sở trong học kết hợp
- Lasso hiệu quả trong xử lý với dữ liệu lớn và thuận tiện trong việc lựa chọn thuộc tính

Thuật toán Bagging Lasso

- 1 Cho D là tập n mẫu huấn luyện $(\mathbf{x}^i, y_i), i \in 1, 2, \dots, n$
- 2 t là số lần kết hợp (lặp) L là thuật toán học cơ sở
- 3 for $(i = 0; i < t; i++)\{$
- 4 Tạo ra tập huấn luyện bootstrap D_i với kích thước mẫu n bằng lấy mẫu có thay thế
- 5 Học hàm phân loại cụ thể $L_i(x, y)(= 0 \text{ hoặc } 1)$ trên tập dữ liệu T_i sử dụng thuật toán học cơ sở L
- 6 Hàm phân loại cuối cùng là sự kết hợp các hàm phân loại $C = \{L_i\}$, mẫu kiểm tra x^* được phân loại vào lớp số đông:

$$y^* = \arg \max_y \sum_{L_i \in C} L_i(x^*, y)$$

Kết quả trên tập dữ liệu Kaggle

- Sử dụng một vài biến đổi dựa vào biến gốc như biến đổi log, thêm biến mới ứng với những giá trị đặc biệt, phân khoảng, ...
- Ví dụ sinh các biến từ biến "Age":
 - 1 $\text{tuoi_tre} = 1$, nếu $\text{"Age"} \leq 17$
 - 2 $\log_tuoi_lao_dong = \log(\text{"Age"} - 17)$, nếu $18 \leq \text{"Age"} \leq 59$
 - 3 $\text{tuoi_gia} = 1$, nếu $\text{"Age"} \geq 60$
- Kết quả có 80 biến sinh ra từ biến gốc và 150, 000 mẫu
- Train and test 10 times (train samples/test samples = 4/1); using cross validation to select tuning parameter and AUC for evaluating classification performance
- Áp dụng Hồi quy logistic: Trung bình $\text{ACC} = 0.9362$, $\text{AUC} = 0.8477$
- Áp dụng Hồi quy logistic Lasso: Trung bình $\text{ACC} = 0.9366$, $\text{AUC} = 0.8607$

Kết quả trên tập dữ liệu Kaggle

- Áp dụng Bagging với thuật toán cơ sở Hồi quy logistic: Trung bình $ACC = 0.8155$, $AUC = 0.8598$
- Áp dụng Bagging với thuật toán cơ sở Hồi quy logistic Lasso: Trung bình $ACC = 0.8167$, $AUC = 0.8611$
- Áp dụng Deep Learning: Trung bình $ACC = 0.7434$, $AUC = 0.7797$
- Áp dụng Gradient Booting (GB): số cây ntrees = 10000; Số biến đầu vào cho mỗi cây là 80% ($col_sample_rate = 0.8$); Số mẫu mỗi lần huấn luyện là 80% ($sample_rate = 0.8$); Trung bình $ACC = 0.9375$, $AUC = 0.8660$

Thank You!