AI VIET NAM – COURSE 2024

Random Forest

Ngày 5 tháng 3 năm 2024

Ngày tóm tắt:	07/03/2024
Tác giả:	AIO
Nguồn dữ liệu (nếu có):	Link of Module 4 AIO 2023
Từ khóa:	Random Forest,Decision Tree, Entropy, Information Gain
Người tóm tắt:	Vũ Mai Thi

1. **Tóm lược Decision Tree (Cây quyết định):**

   - Bạn có biết rằng trong cuộc sống hàng ngày, bạn vẫn đang sử dụng phương pháp Decision Tree (Cây quyết định). Chẳng hạn, bạn đến siêu thị mua sữa cho cả gia đình. Câu đầu tiên trong đầu bạn sẽ là: Bạn cần mua bao nhiêu sữa? Bạn sẽ xác định: Nếu là ngày thường thì gia đình bạn sẽ sử dụng hết 1 lít sữa, còn cuối tuần thì sẽ là 1,5 lít. Như vậy, dựa theo ngày, bạn sẽ quyết định lượng thực phẩm cần mua cho gia đình bạn. Đó chính là một dạng của cây quyết định nhị phân.
   - Cây quyết định (Decision Tree) là một cây phân cấp có cấu trúc được dùng để phân lớp các đối tượng dựa vào dãy các luật. Các thuộc tính của đối tượngncó thể thuộc các kiểu dữ liệu khác nhau như Nhị phân (Binary) , Định danh (Nominal), Thứ tự (Ordinal), Số lượng (Quantitative) trong khi đó thuộc tính phân lớp phải có kiểu dữ liệu là Binary hoặc Ordinal.
   - Tóm lại, cho dữ liệu về các đối tượng gồm các thuộc tính cùng với lớp (classes) của nó, cây quyết định sẽ sinh ra các luật để dự đoán lớp của các dữ liệu chưa biết.

2. **Random Forest (Rừng cây ngẫu nhiên):**

   - Random forest là một phương pháp thống kê mô hình hóa bằng máy (machine learning statistic) dùng để phục vụ các mục đích phân loại, tính hồi quy và các nhiệm vụ khác bằng cách xây dựng nhiều cây quyết định (Decision tree).Random Forest cho thấy hiệu quả hơn so với thuật toán phân loại thường được sử dụng vì có khả năng tìm ra thuộc tính nào quan trọng hơn so với những thuộc tính khác
   - Hướng tiếp cận Ensemple có homogeneous approach (cùng một giải thuật đầu vào) và heterogenerous (nhiều giải thuật đầu vào)

3. **Cách xây dựng Ramdom Forest:**

   - Dữ liệu về bệnh nhân có khả năng mắc bệnh tim thông qua 4 tiêu chí: đau ngực, tuần hoàn máu tốt,tắc động mạch và cân nặng, từ đó quyết định xem người đó có khả năng bị bệnh tim hay không?

AI VIETNAM
All-in-One Course

Step to Random Rorest

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
NO	NO	NO	125	NO
YES	YES	YES	180	YES
YES	YES	NO	210	NO
YES	NO	YES	167	YES

Hình 1: Ví dụ dữ liệu về bệnh nhân có khả năng mắc bệnh tim

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
NO	NO	NO	125	NO
YES	YES	YES	180	YES
YES	YES	NO	210	NO
YES	NO	YES	167	YES

Original DATA

Chọn lựa ngẫu nhiên từ dataset ban đầu

New DATA

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

Hình 2: Tạo tập dữ liệu mới

- Bước 1: Tạo một tập dữ liệu mới, chọn lựa ngẫu nhiên từ dataset ban đầu. Đặt stt từ 1-4, mỗi lần random được stt nào thì đưa vào tập dữ liệu mới, chấp nhận cả việc dữ liệu bị trùng.

- Bước 2: Tạo cây từ Boostrapped dataset. Đầu tiên random ngẫu nhiên hai thuộc tính từ 4 thuộc tính

GENERATE DECISION TREES FROM THE BOOTSTRAPPED DATASET USING **PREDEFINED CONDITIONS**

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

A RANDOM SUBSET OF 2 ATTRIBUTES (OR 2 COLUMNS).

CHEST PAIN	GOOD BLOOD CIRCULATION	BLOCKED ARTERIES	WEIGHT	HEART DISEASE
YES	YES	YES	180	YES
NO	NO	NO	125	NO
YES	NO	YES	167	YES
YES	NO	YES	167	YES

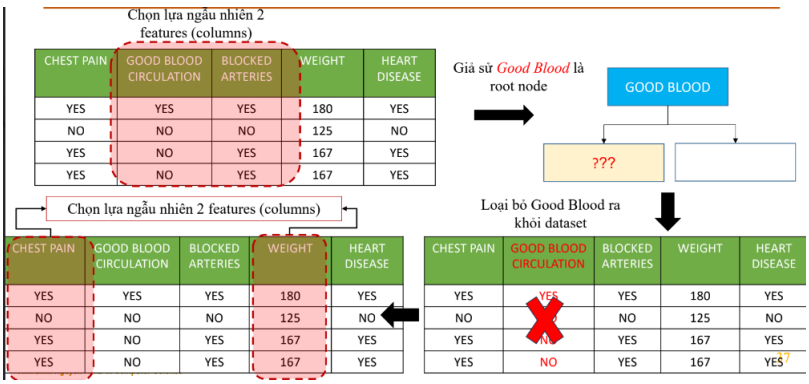
Traditional Tree

Tree with Predefined Conditions

36

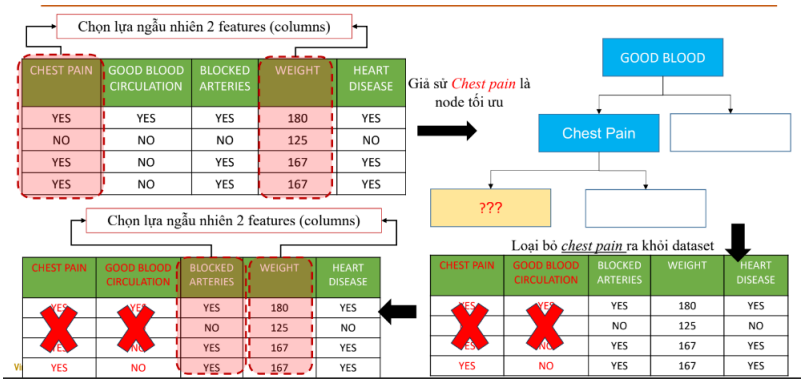
Hình 3: minh họa

- Bước 3: Tiếp theo chọn Good Blood là nút gốc thì xóa nút gốc khỏi dataset. Sau đó random ngẫu nhiên 2 thuộc tính mới. Giả sử chọn được Chest pain và Weigh

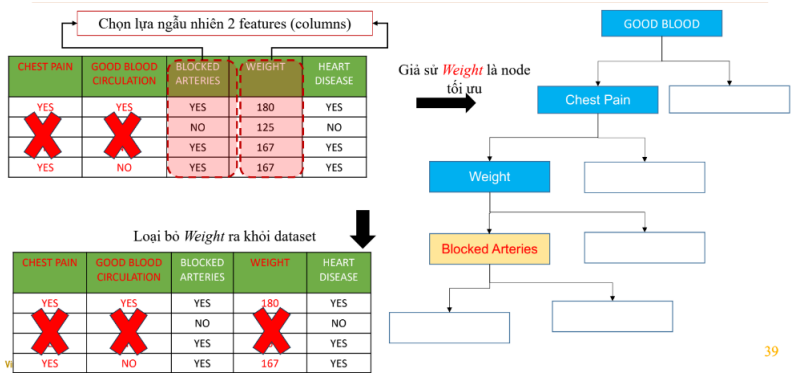


Hình 4: minh họa

- Bước 4: Chọn tiếp Chest pain là nút tối ưu, thì xóa Chest pain ra khỏi dataset. Rồi chọn nốt hai thuộc tính cuối,giả sử chọn tiếp Weigh là nút tối ưu thì sẽ xóa Weigh khỏi dataset. Từ đó xây dựng được 1 cây



Hình 5: minh họa



Hình 6: Tạo dựng được 1 cây

4. **Công thức:**

- Công thức Entropy

$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

- Ví dụ bài tập và lời giải

$s = [0, 0, 0, 0, 0, 0, 0, 1, 1, 1]$

$n_0 = 7$

$n_1 = 3$

$E(S) = -\frac{7}{10} log_2(\frac{7}{10}) - \frac{3}{10} log_2(\frac{3}{10})$

$E(S) = -0.7 log_2(0.7) - 0.3 log_2(0.3)$

$E(S) = -0.7 \times -0.51457 - 0.3 \times -1.73697$

$E(S) = 0.88129$

- code minh họa

```
[38] import numpy as np
     from collections import Counter
```

```
[40] # Tính entropy
     def entropy(s):
         counts = np.bincount(s)
         percentages = counts / len(s)

         entropy = 0
         for pct in percentages:
             if pct > 0:
                 entropy += pct * np.log2(pct)
         return -entropy
```

```
[42] # Kiểm tra
     s = [0, 0, 0, 0, 0, 0, 0, 1, 1, 1]
     print(f'Entropy: {np.round(entropy(s), 2)}')

     Entropy: 0.88
```

- Công thức Information Gain

$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

- ví dụ bài tập và lời giải

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

$$Gain(S, A) = 0.97095 - \frac{12}{20} \times 0.65002 - \frac{8}{20} \times 1$$

$$Gain(S, A) = 0.18094$$

- code minh họa

```
[44] #Tính information gain
     def information_gain(parent, left_child, right_child):
         num_left = len(left_child) / len(parent)
         num_right = len(right_child) / len(parent)

         gain = entropy(parent) - (num_left * entropy(left_child) + num_right * entropy(right_child))
         return gain
```

```
[45] # Kiểm thử
     parent = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1]
     left_child = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1]
     right_child = [0, 0, 0, 0, 1, 1, 1, 1]

     print(f'Information gain: {np.round(information_gain(parent, left_child, right_child), 5)}')

     Information gain: 0.18094
```