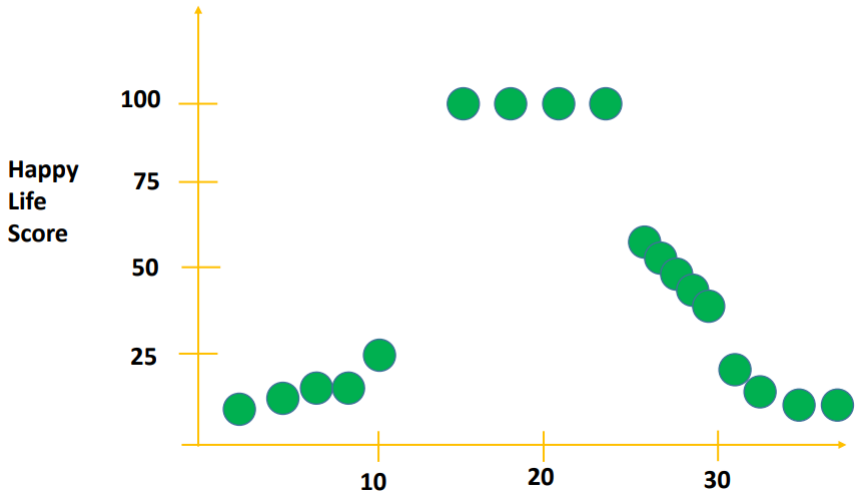
AI VIET NAM – COURSE 2024

Decision Tree for Regression

Ngày 4 tháng 3 năm 2024

Ngày công bố:	16/02/2024
Tác giả:	
Người tóm tắt:	Nguyễn Văn Nam

1. **Đặt vấn đề:**



Hình 1: Dataset phức tạp

- Nếu dataset phức tạp thì dùng giải thuật Linear regression sẽ rất khó khăn => Vì vậy giải thuật Regression tree được ra đời để giải quyết vấn đề này
- Bài toán về regression là chia nhỏ dữ liệu và tính trung bình của đoạn dữ liệu đó sao cho đường đại diện cho đoạn dữ liệu có độ lệch trung bình nhỏ tốt nhất, đại diện tối ưu nhất cho đoạn dữ liệu đó.

2. **Regression tree:**

- Xét 1 ví dụ cụ thể, chúng ta cần dự đoán kết quả đầu ra của dataset dựa trên các thuộc tính: Unit, Age, Sex như hình bên dưới
- Ý tưởng xây dựng giải thuật: dựa trên dataset có sẵn, ta đi tìm node gốc của bộ dữ liệu bằng cách thử lần lượt các thuộc tính của dataset, thuộc tính nào tốt nhất thì sẽ là root node của cây

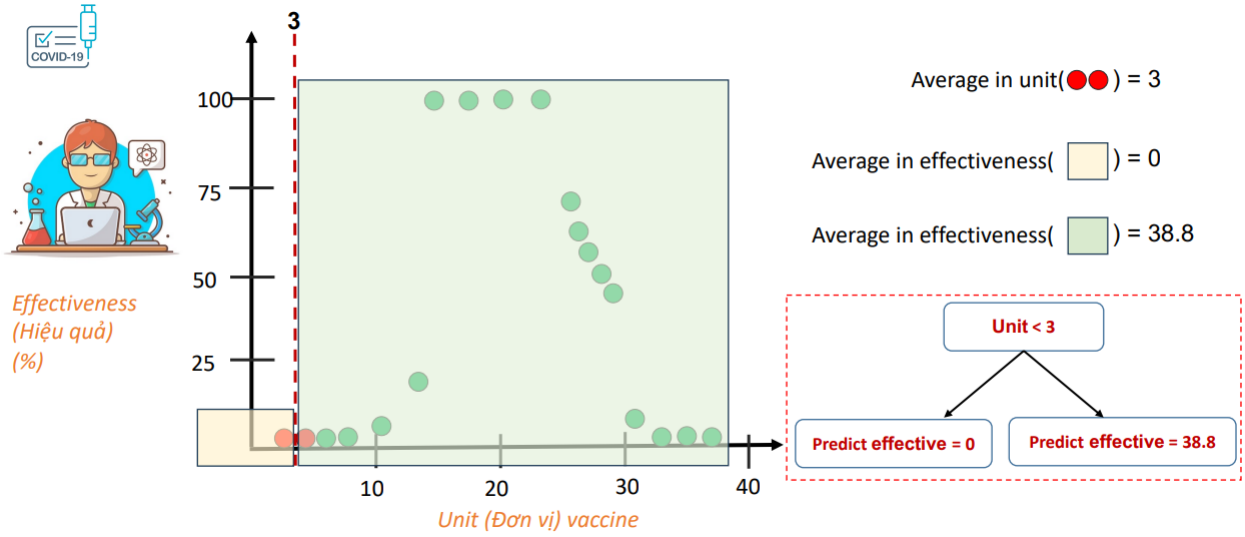
3. **Cách xác định:**

Unit	Age	Sex	Effect (%)
10	25	Female	98
20	73	Male	0
35	54	Female	100
5	12	Male	44
…	…	…	…

Khi có 1 vaccine ra đời, chúng ta muốn dự đoán xem nó hiệu quả bao nhiêu % ứng với liều lượng dùng cố định (unit), tuổi (age) và giới tính (sex) của bệnh nhân.

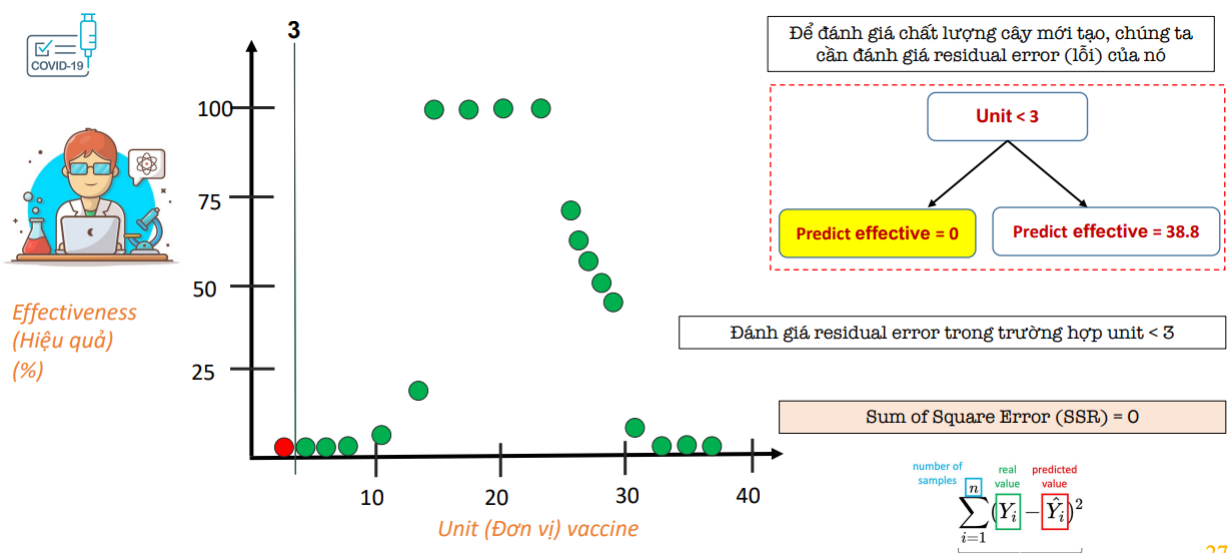
Hình 2: Ví dụ về dataset phức tạp

- Giả sử ta lấy Unit là root node
- Ta cần tìm ngưỡng của dataset:

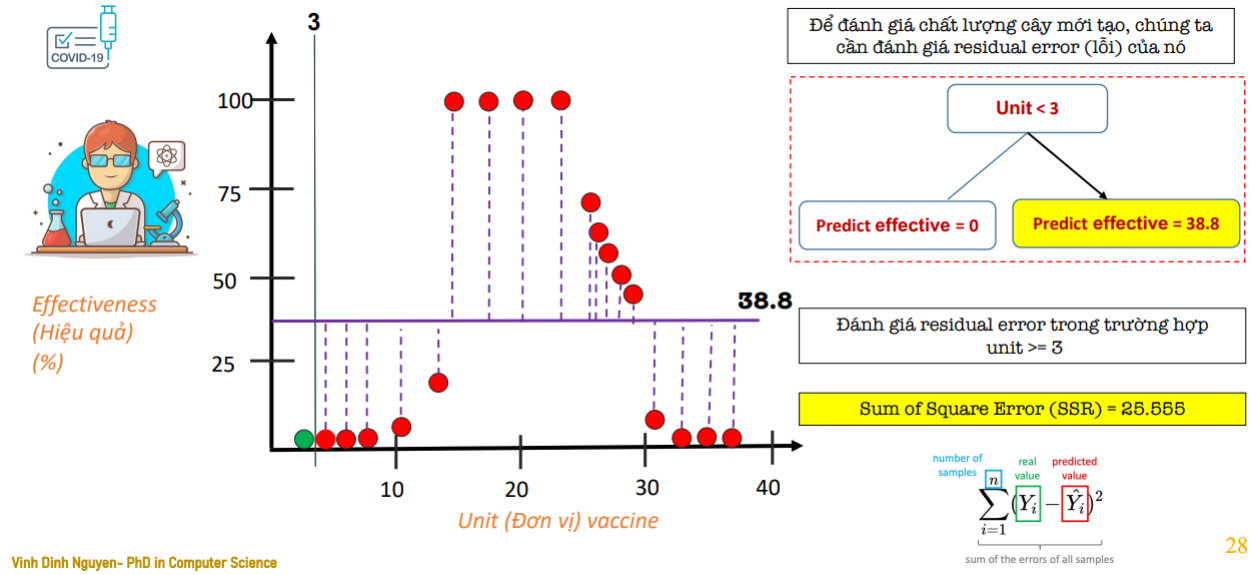


Hình 3: Xác định ngưỡng

- Từ trái qua phải, ta lấy 2 điểm liền kề đầu tiên và lấy giá trị trung bình của 2 điểm đó (ở đây trung bình sẽ là 3)
- Từ trung bình ở trên, ta đã chia dữ liệu thành 2 phần trái và phải. Tiếp theo ta tiến hành lấy giá trị trung bình của mỗi phía
- Theo dataset ta có được trung bình bên trái là 0, trung bình bên phải sẽ là 38.8
- Ta đã có được 1 thành phần cây: nếu Unit $<$ 3 thì hiệu quả sẽ $=$ 0, ngược lại hiệu quả đạt 38.8
- Tiến hành đánh giá độ chính xác của cây: SSR $=$ tổng bình phương của hiệu giá trị thực tế với giá trị dự đoán
- Ta lần lượt đánh giá độ chính xác của từng phần

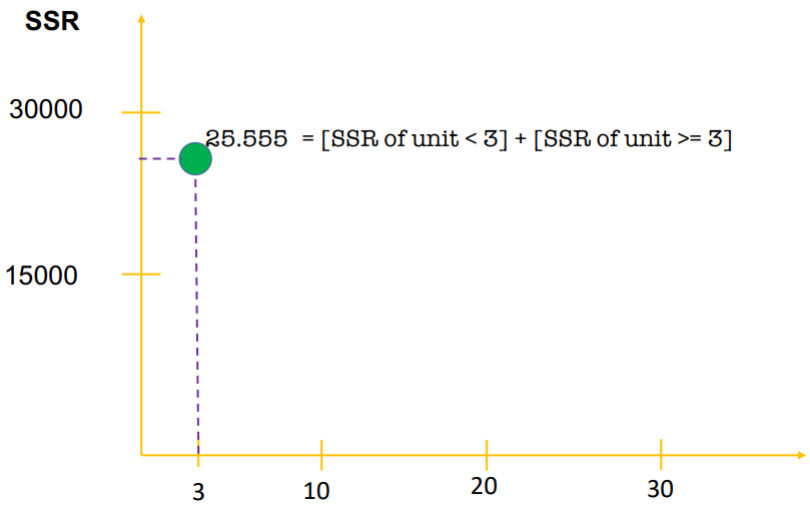


Hình 4: Tính SSR của trường hợp Unit $<$ 3

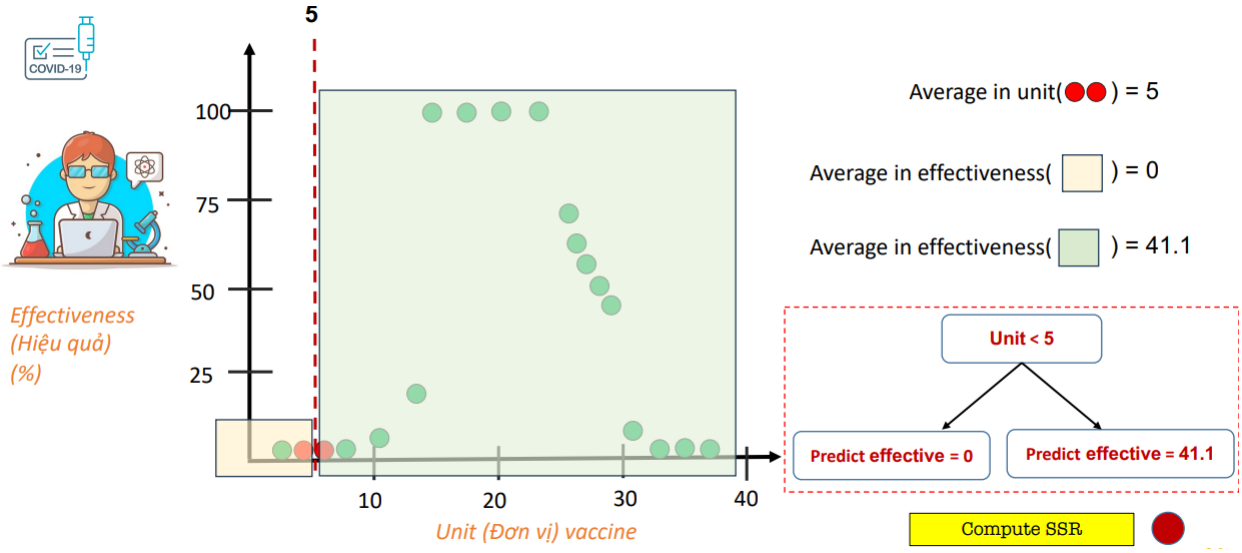


Hình 5: Tính SSR của trường hợp Unit $>$ 3

- Cuối cùng ta tính được tổng SSR 2 nhánh của cây
- Lưu ý rằng tổng SSR hiện tại chỉ là 1 cây đầu tiên, ta phải tiếp tục thực hiện tính tổng SSR của các trường hợp khác và đánh giá tiếp để lấy trường hợp tối ưu nhất.
- Ta tiếp tục đi tính trung bình 2 điểm tiếp theo trong bộ dữ liệu
- Đường trung bình có giá trị bằng 5 và đường trung bình cũng chia dữ liệu thành 2 phần, bên trái và bên phải. Ta tiến hành thực hiện tính giá trị trung bình của bên trái và bên phải, tính SSR của từng phần của tổng giá trị của SSR của cây

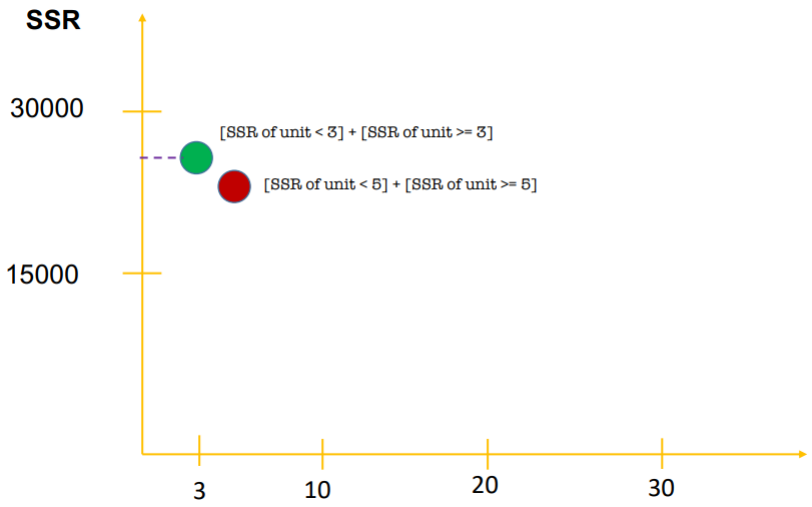


Hình 6: Tổng SSR

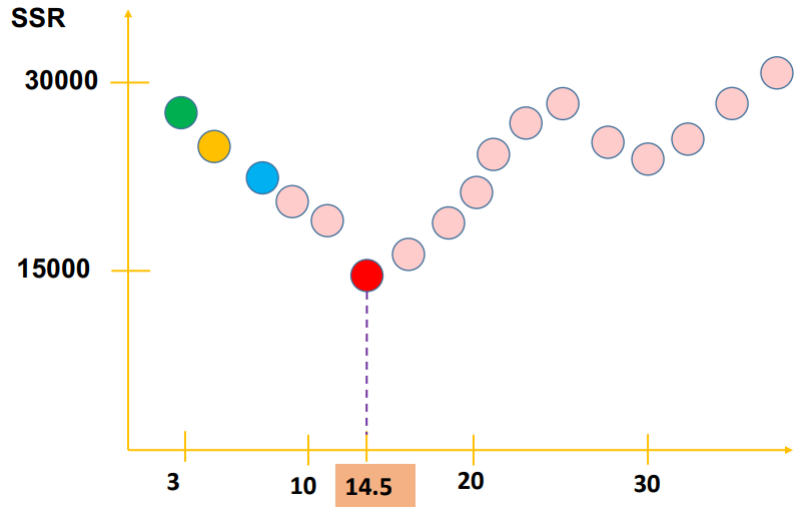


Hình 7: Tìm đường trung bình của 2 điểm tiếp theo

- Quá trình này được lặp qua tất cả các điểm của data, với mỗi 1 ngưỡng (trung bình của 2 điểm) ta sẽ có được giá trị SSR của cây. Cuối cùng ta được 1 ngưỡng có SSR nhỏ nhất chính là ngưỡng tốt nhất của cây



Hình 8: Tổng SSR của 2 điểm kế tiếp



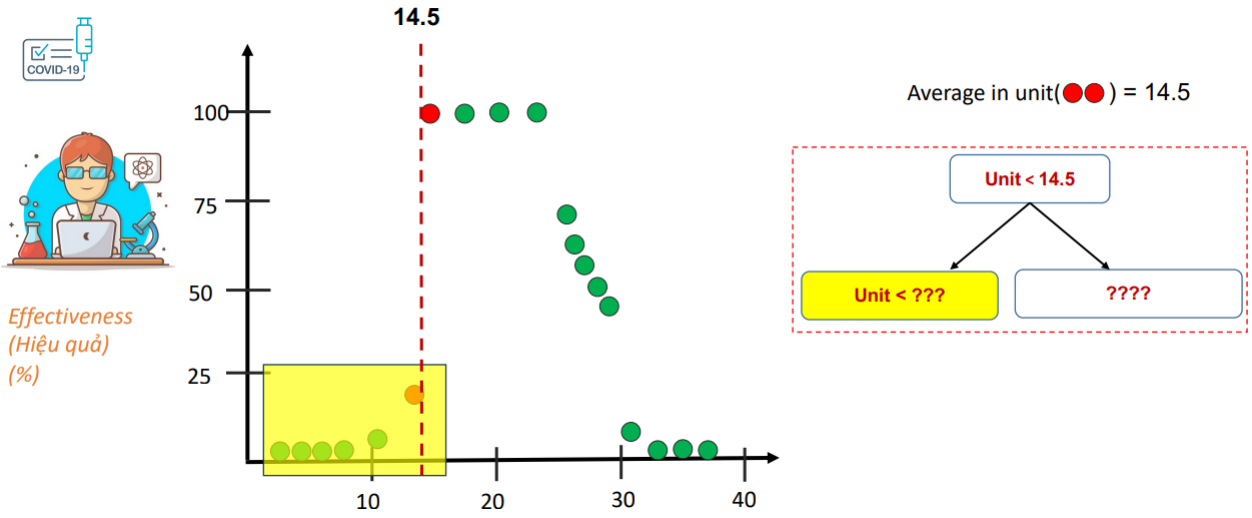
Hình 9: Tìm SSR nhỏ nhất

- Sau khi từng được ngưỡng tốt nhất (ở đây là 14.5), ta được 2 nhánh của cây
- Tiếp tục thực hiện tìm ngưỡng tối ưu nhất đối với từng nhánh của cây tương tự như data ban đầu. Việc tìm ngưỡng ở đây sẽ áp dụng đối với từng nhánh của cây sau khi đã được tách từ đường 14.5
- Quá trình lại tiếp tục lặp đi lặp lại đối với từng ngưỡng, từng nhánh của cây do ngưỡng phân tách

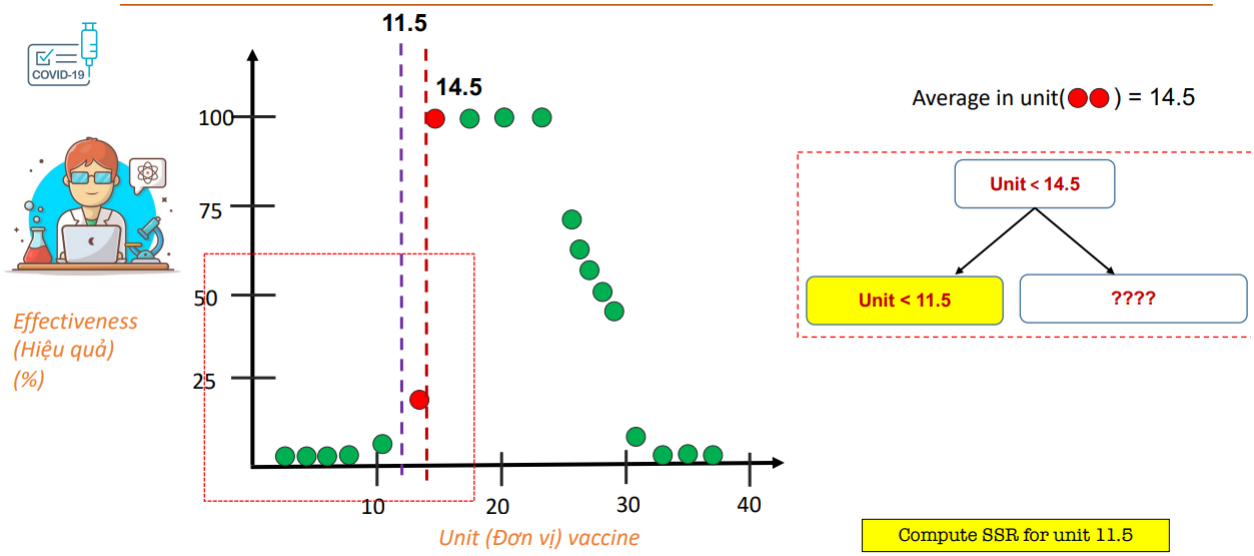
(?) Vấn đề phát sinh hiện tại là đến khi nào ra sẽ dừng việc tìm ngưỡng và đánh giá nó?

- Có rất nhiều điều kiện dừng khác nhau tuỳ thuộc vào lựa chọn hay dataset mà ta có thể dùng các điều kiện dừng khác nhau, một vài điều kiện có thể có như:

+ Độ sâu của cây không quá 3
+ Tổng số sample trong node không vượt quá 4,5 hoặc 6
+ Các sample giống nhau hơn



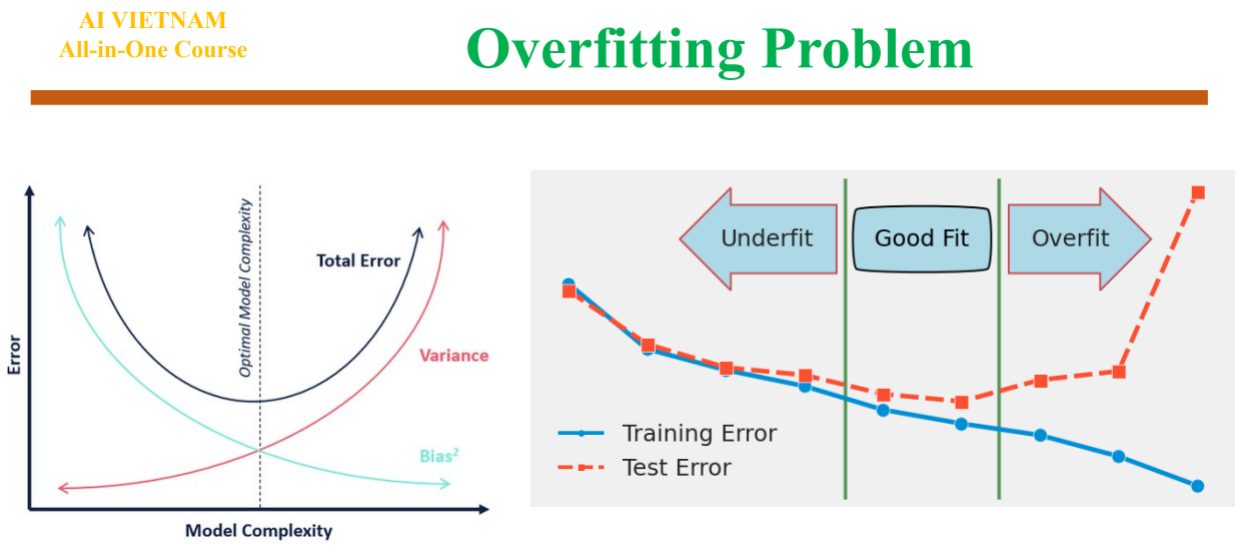
Hình 10: 2 nhánh của cây từ ngưỡng 14.5



Hình 11: Ngưỡng bên trái của cây

4. **Overfitting:**

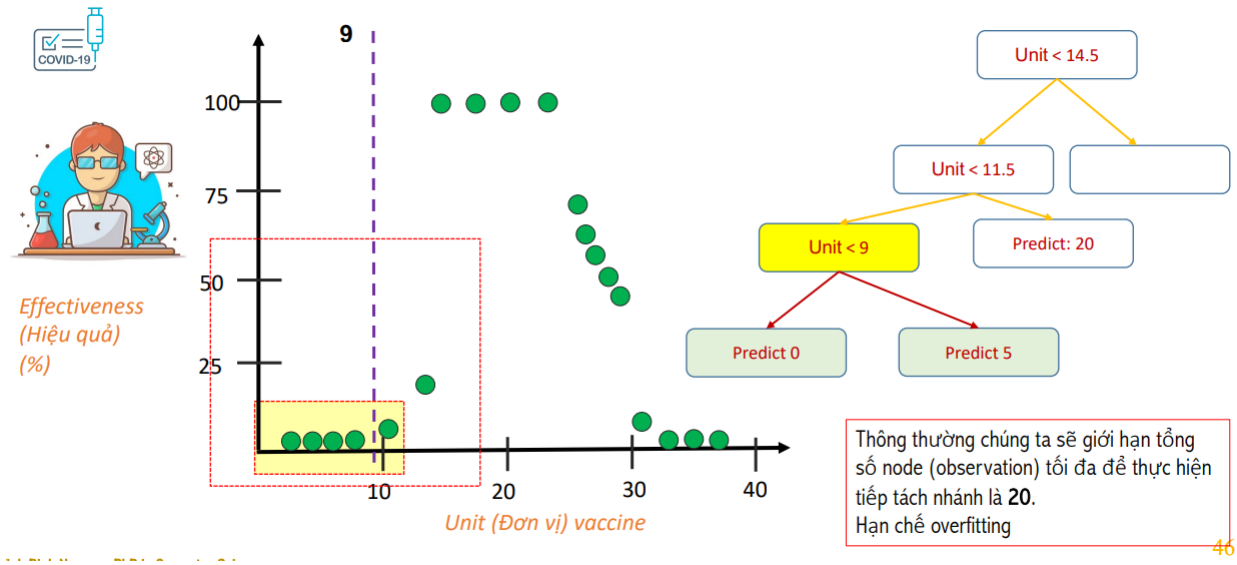
- Underfit: là hiện tượng mô hình Machine Learning hoặc Deep Learning không học được đủ kiến thức từ dữ liệu huấn luyện và không đạt được hiệu suất tốt trên cả tập huấn luyện và tập kiểm tra (high bias or low variance)
- Good Fit: là nằm giữa Underfitting và Overfitting. Mô hình cho ra kết quả hợp lý với cả tập dữ liệu huấn luyện và các tập dữ liệu mới. Đây là mô hình lý tưởng mang được tính tổng quát và khớp được với nhiều dữ liệu mẫu và cả các dữ liệu mới.
- Overfit: là mô hình rất hợp lý, rất khớp với tập huấn luyện nhưng khi đem ra dự đoán với dữ liệu mới thì lại không phù hợp. Nguyên nhân có thể do ta chưa đủ dữ liệu để đánh giá hoặc do mô hình của ta quá phức tạp. Mô hình bị quá phức tạp khi mà mô hình của ta sử dụng cả những nhiễu lớn trong tập dữ liệu để học, dẫn tới mất tính tổng quát của mô hình (high variance or low bias). Nếu kết quả training quá tốt đạt tỷ lệ 100% thì cần phải xem



Hình 12: Overfitting

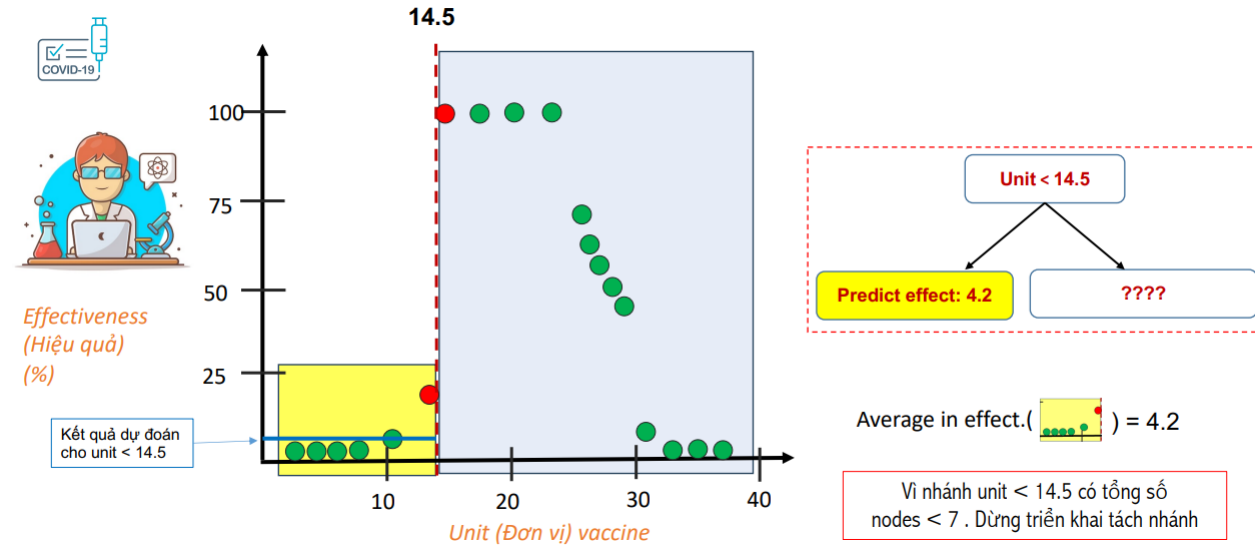
xét lại dataset vì rất có thể ta đang mắc phải trường hợp overfitting.

- Để tránh overfitting thông thường ta sẽ giới hạn tổng số node (observation) tối đa để thực hiện tách tiếp là từ 8-20



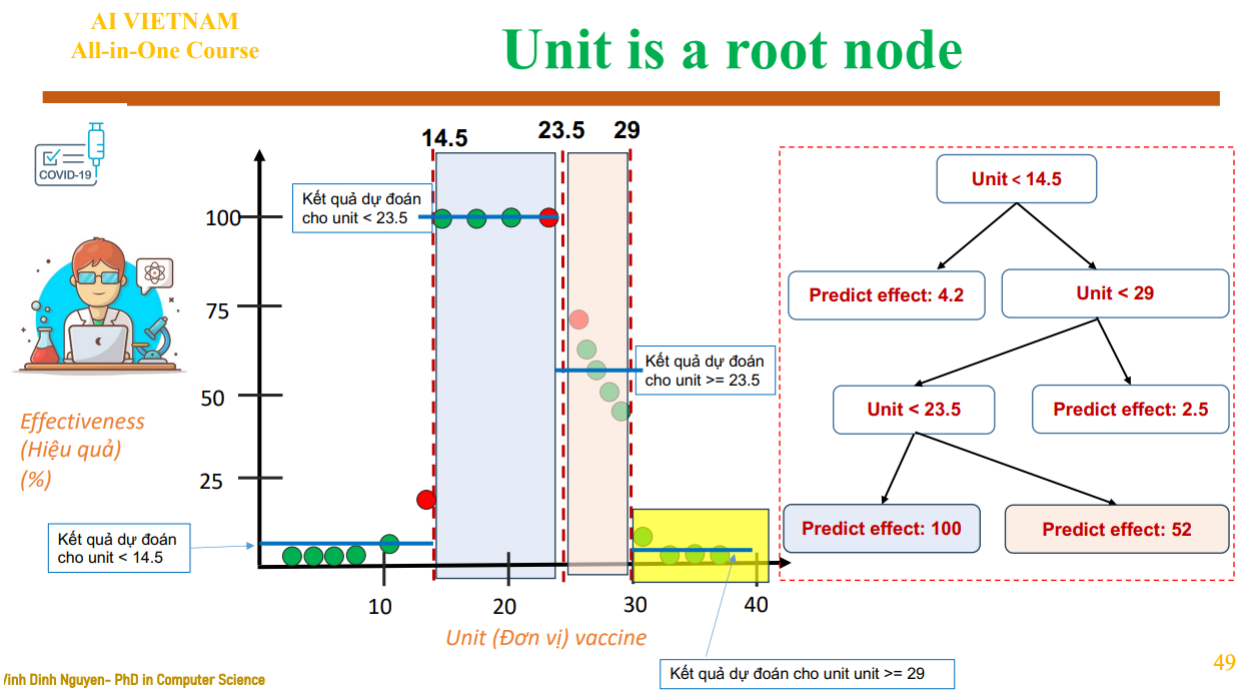
Hình 13: Hạn chế overfitting

- Áp dụng điều kiện vào bài toán, ta chọn số node là 7, ta được kết quả:



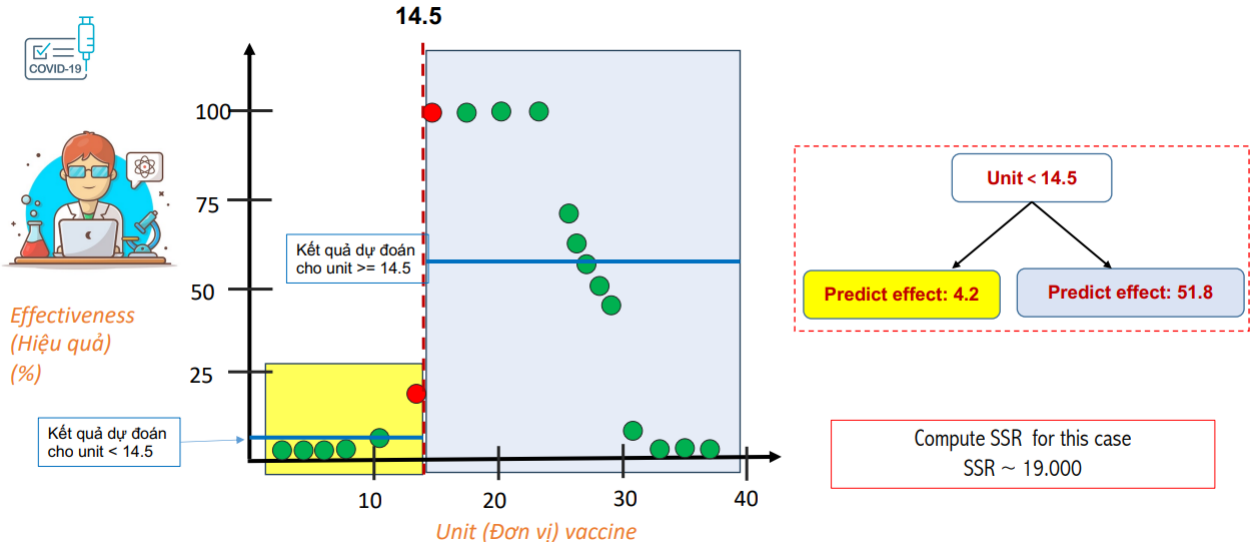
Hình 14: Đánh giá điều kiện dừng

- Vì nhánh bên trái thoả mãn điều kiện dừng nên ta sẽ tính trung bình của các node và dừng việc tách nhánh. Nhánh bên phải không thoả mãn điều kiện nên ta tiếp tục phân nhánh của nhánh bên phải
- Sau khi lặp lại các bước tìm ngưỡng tách nhánh kết hợp với điều kiện dừng thì ta được cây hoàn chỉnh như sau:



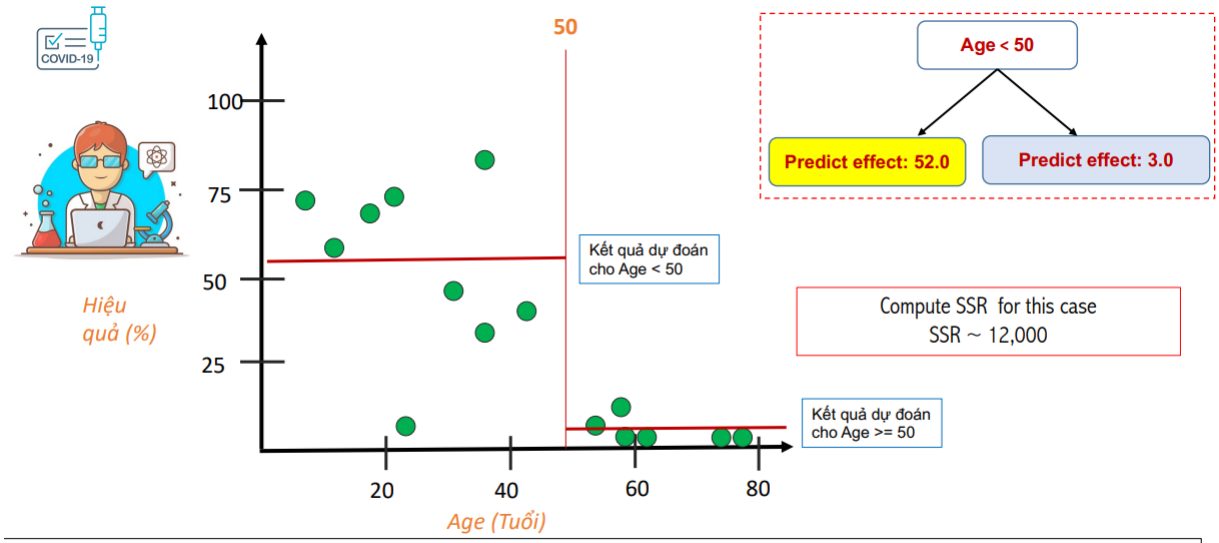
Hình 15: Cây hoàn chỉnh với node < 7

- Tiếp theo ra đi tính SSR của toàn bộ cây bằng tổng SSR của từng nhánh.
- Nếu ra thay đổi điều kiện dừng của cây là 20 thì cây sẽ có dạng:

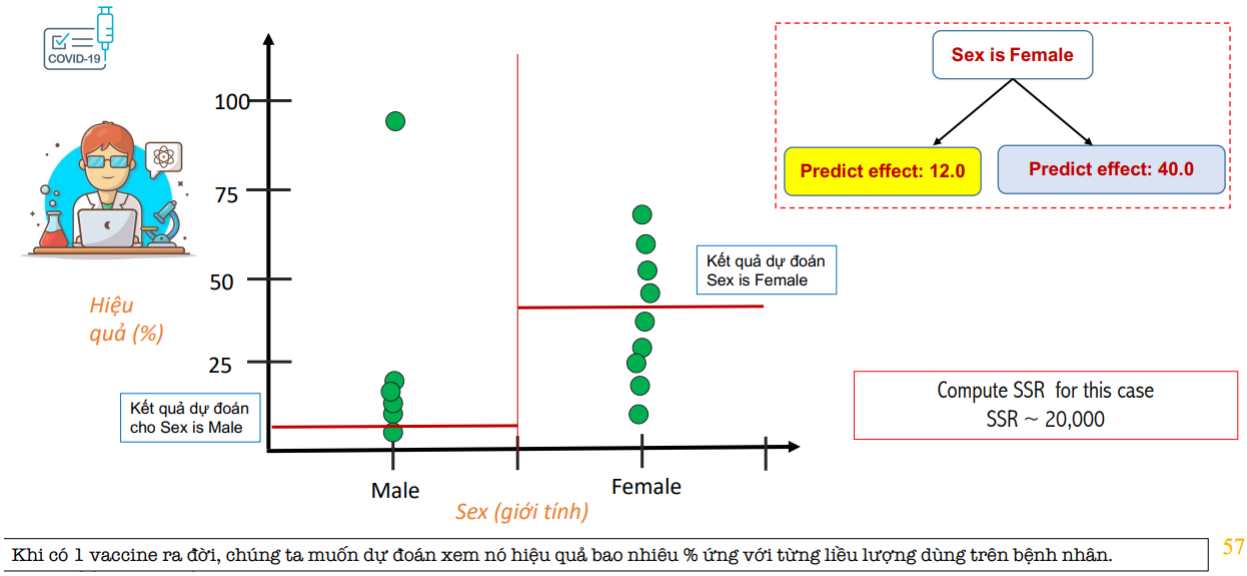


Hình 16: Cây hoàn chỉnh với node < 20

- Ta cũng đi tính tổng SSR của cây và sẽ lấy cây có SSR tốt nhất
- Vì dataset có 3 thuộc tính nên sẽ đi lần lượt các thuộc tính tiếp theo. Nếu Age, Sex là root node, ta cũng làm tương tự như đối với Unit là root node, kết quả nếu Age, Sex là root node lần lượt có dạng:

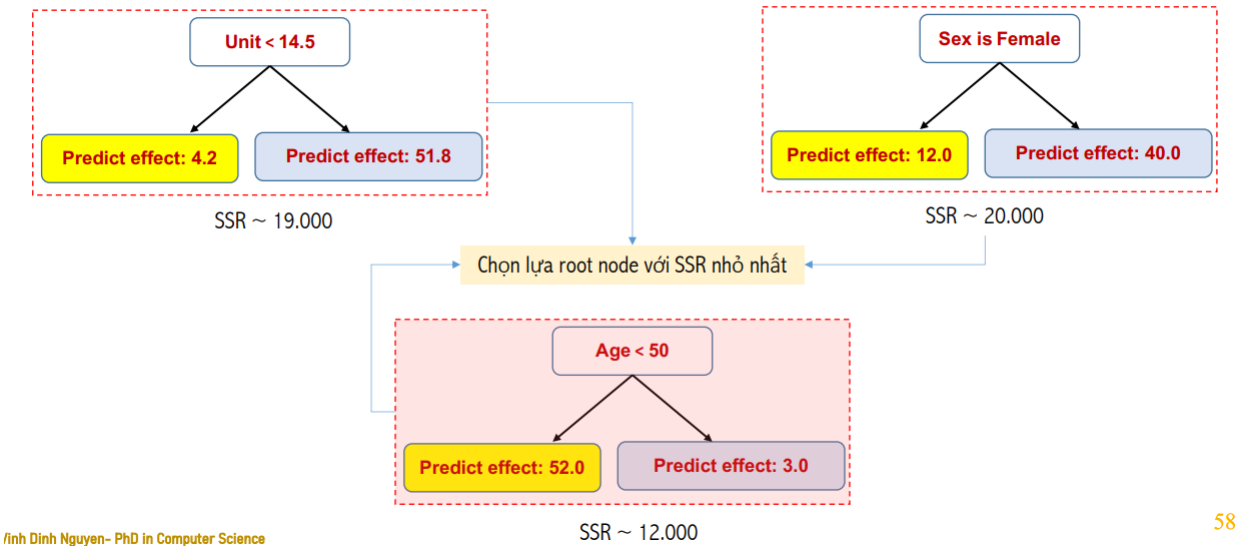


Hình 17: Cây hoàn chỉnh với Age là root node



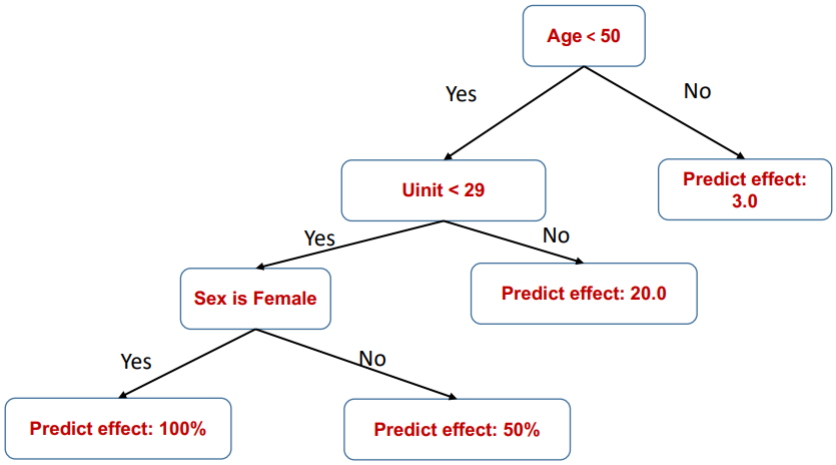
Hình 18: Cây hoàn chỉnh với Sex là root node

- Từ 3 thuộc tính của dataset, ta lại lấy node có SSR nhỏ nhất



Hình 19: Chọn node có SSR nhỏ nhất

- Sau khi chọn được node gốc đầu tiên để phân nhánh, ta tiếp tục tiến hành lấy node tiếp theo từ các thuộc tính còn lại của dataset, kết quả sẽ được:

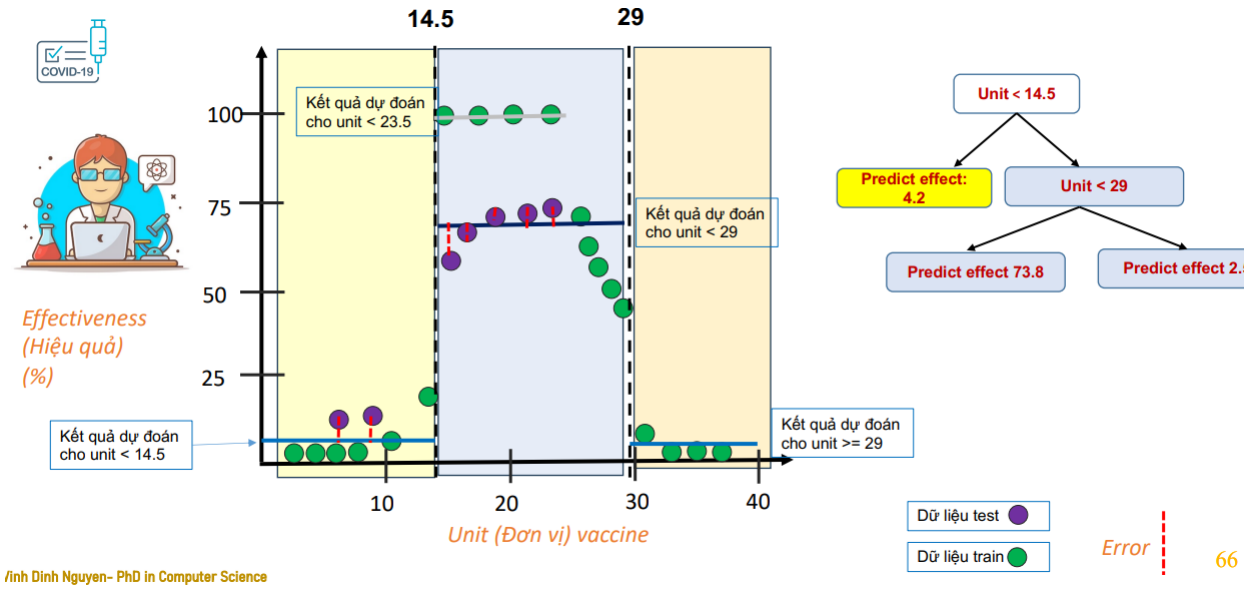


Hình 20: Cây hoàn chỉnh

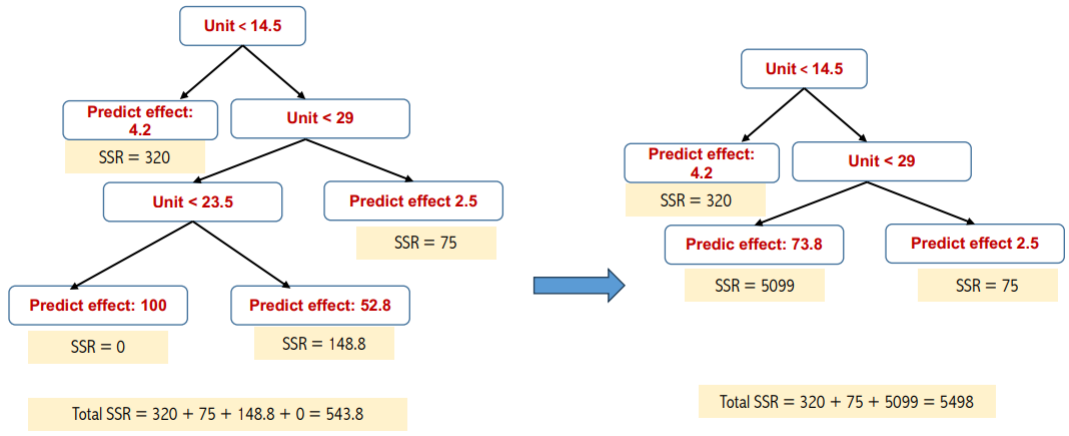
5. **Prunning:**

- Sau khi xây dựng được cây, ta thấy cây sẽ gặp hiện tượng overfitting (dữ liệu train và test chêch lệch nhau quá lớn), vì vậy ra cần bỏ bớt nhánh đã phân (prunning)

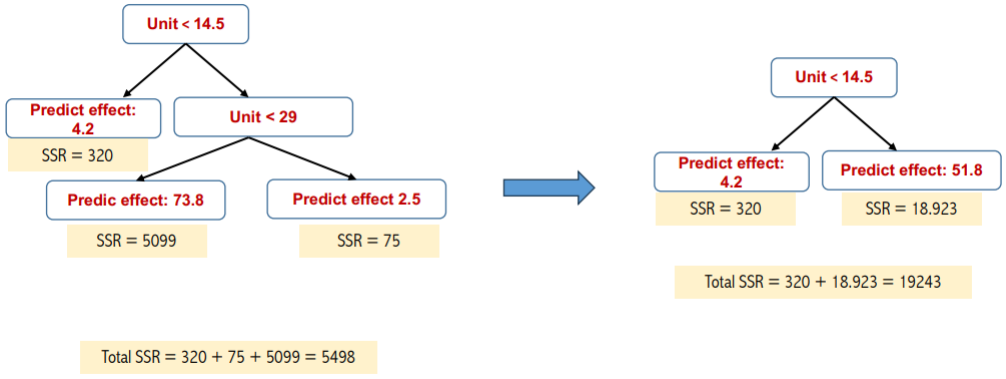
- Ta sẽ tiến hành bỏ nhánh của cây lần lượt, ta được



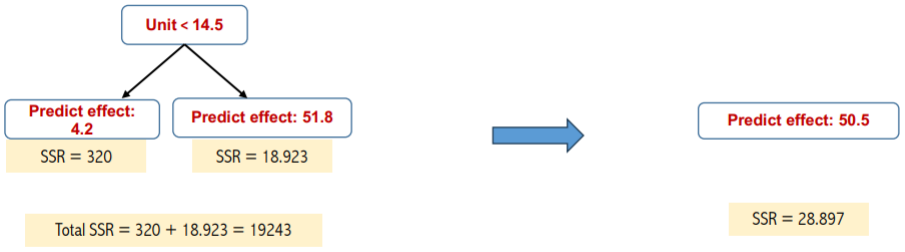
Hình 21: Bỏ bớt nhánh



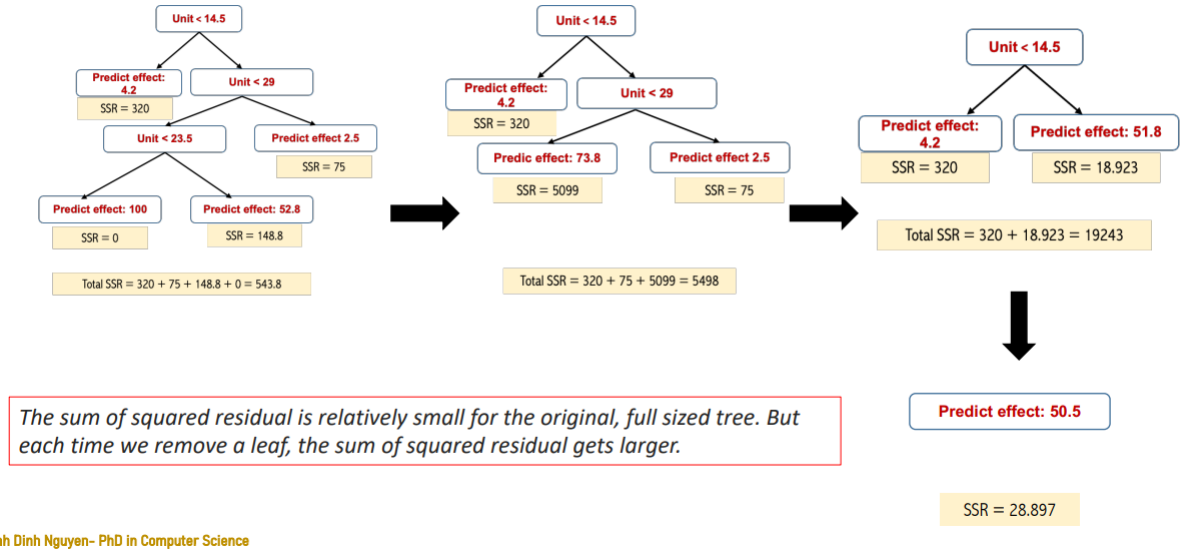
Hình 22: Tổng SSR sau khi prunning



Hình 23: Bỏ bớt nhánh lần 2



Hình 24: Tổng SSR sau khi prunning lần 2

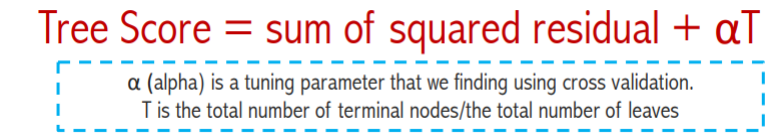


Hình 25: So sánh SSR giữa các lần prunning

? Ta thấy khi prunning thì hiện tượng overfitting sẽ được khắc phục nhưng SSR lại rất lớn, vậy làm cách nào để xác định cây tốt nhất?

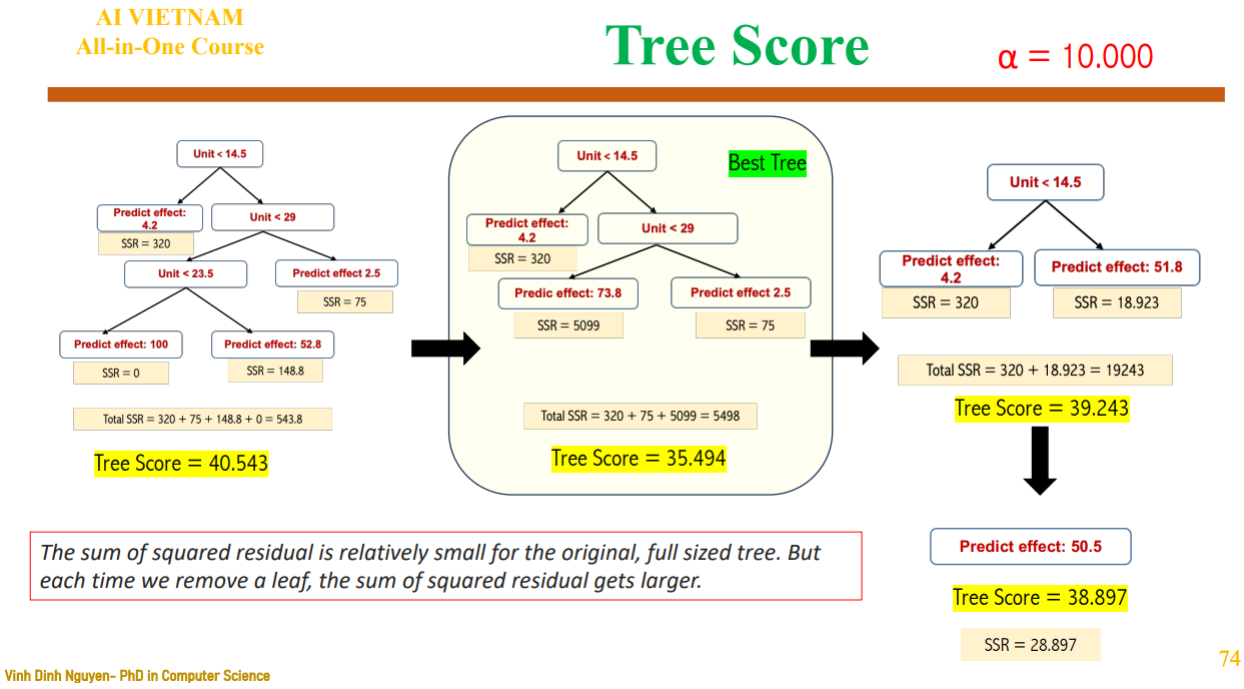
6. **Tree complexity penalty:**

- Tree score là độ phức tạp của cây



Hình 26: Công thức tree score

- Áp dụng công thức ta tính tree score của 3 cây đã tỉa bớt nhánh, ở đây ta chọn $\alpha = 10.000$

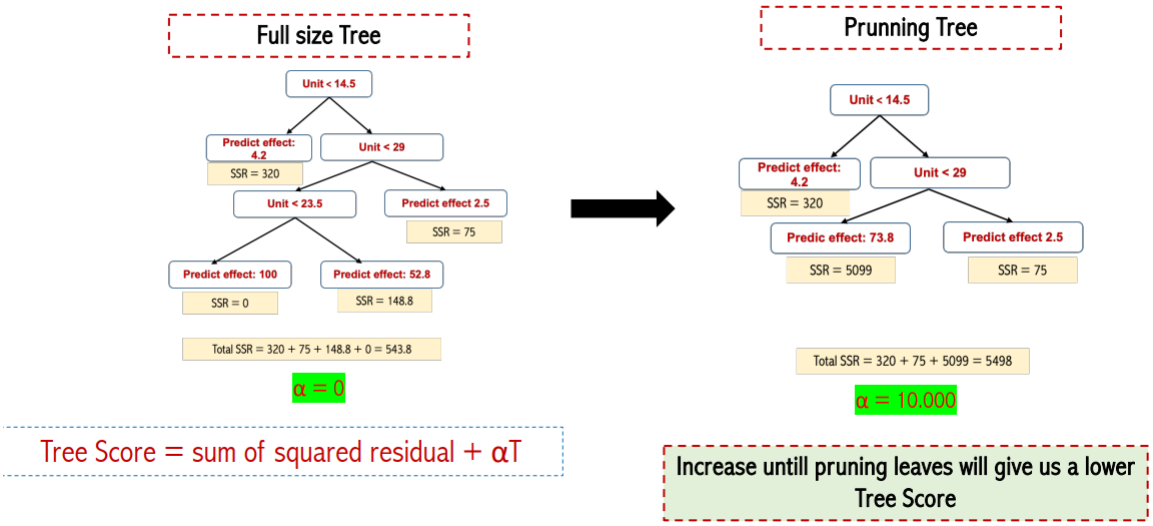


Hình 27: Tính tree score

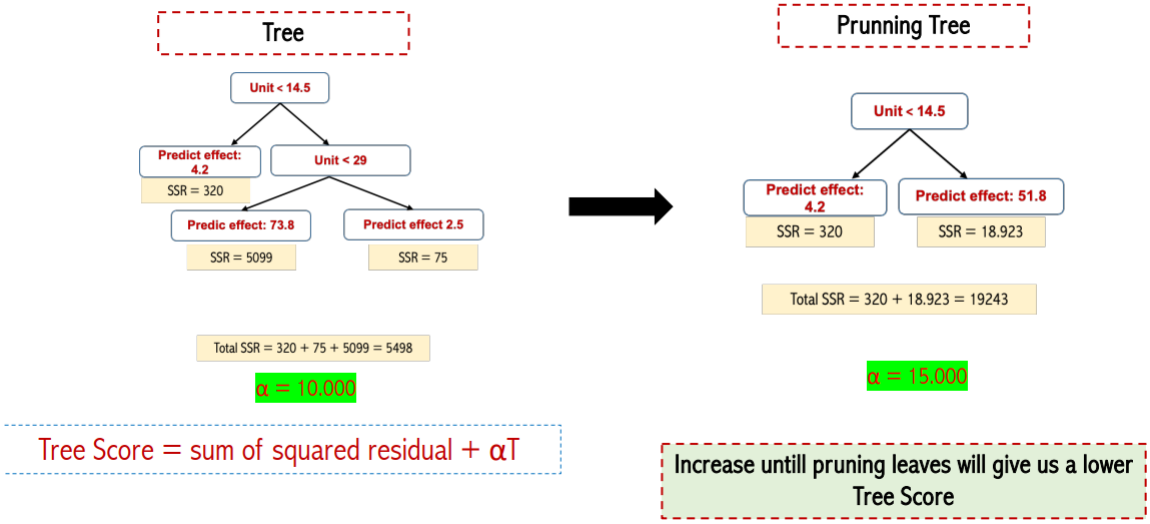
- Ta sẽ chọn cây có tree score tốt nhất

? Chọn α như thế nào?

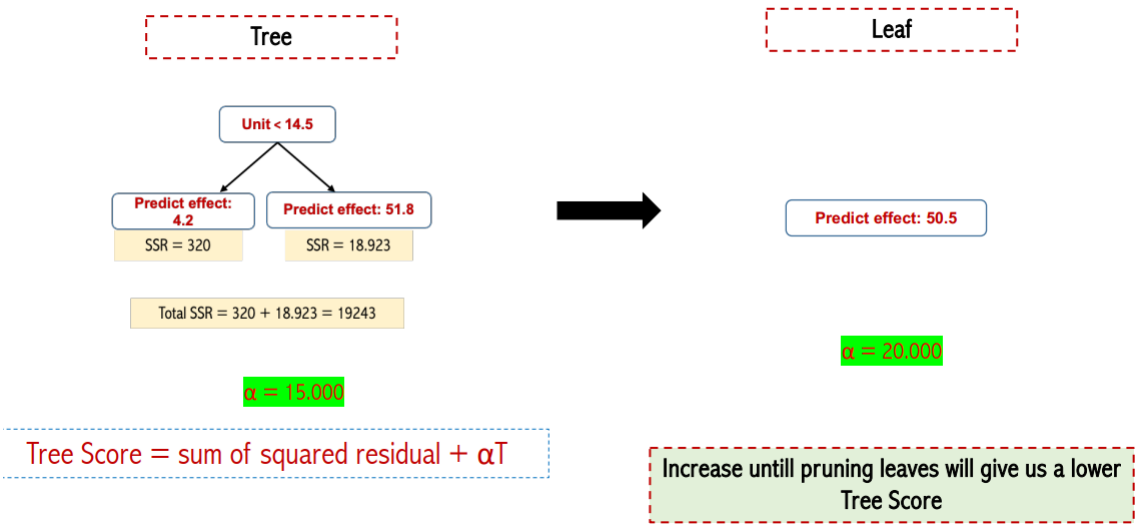
- Cho $\alpha = 0$ thì ta được giá trị tree score nhỏ nhất
- Khi ta bỏ bớt nhánh của cây, ta sẽ tính tree score với α tăng dần sao cho giá trị tree score mới tốt hơn tree score ban đầu
- Tiếp tục bỏ bớt nhánh và tăng giá trị α
- Có bao nhiêu cây được bỏ nhánh thì có bấy nhiêu giá trị α được xác định



Hình 28: Tính tree score

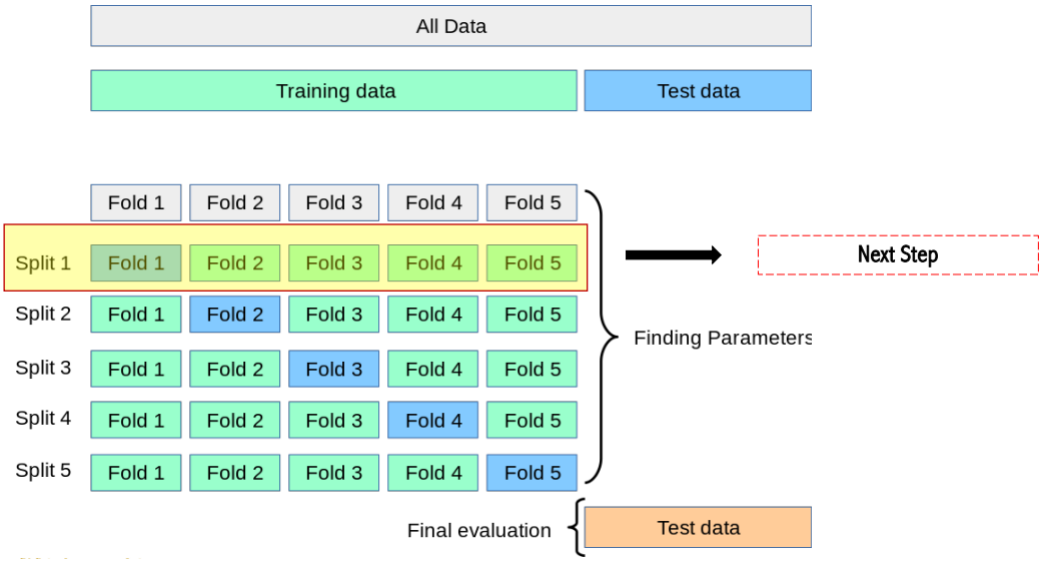


Hình 29: Tính tree score

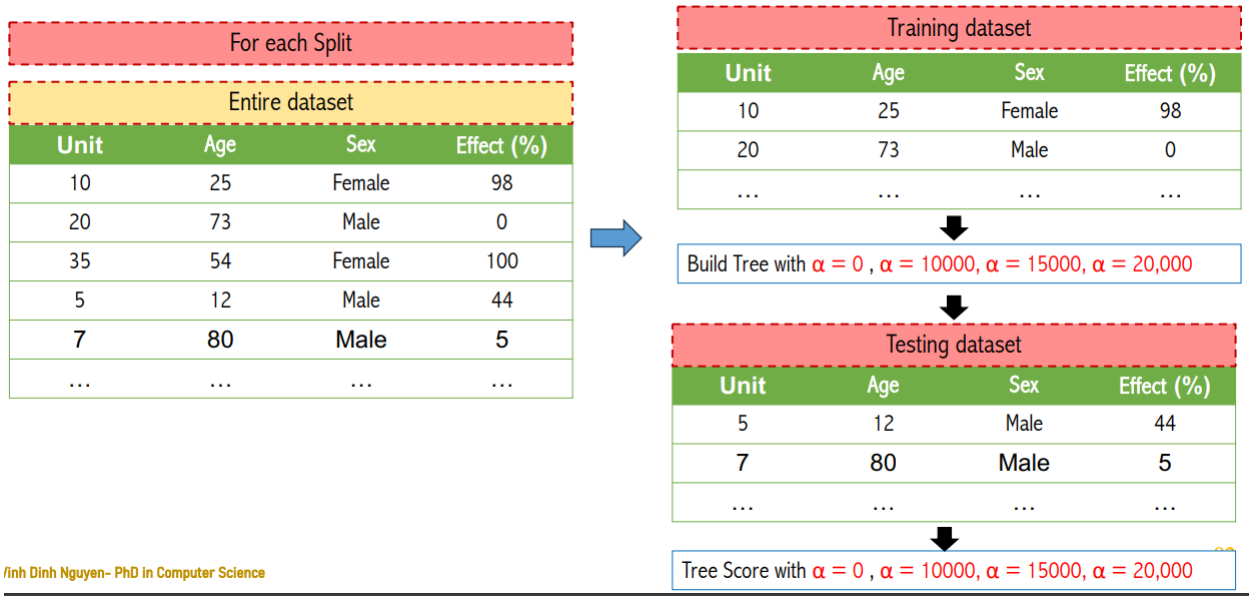


Hình 30: Tính tree score

- Sử dụng K fold chia dữ liệu thành dữ liệu train và dữ liệu test, dữ liệu được phân tách thành các fold, ta phải đi xây dựng cây theo giá trị α đã được xác định ở trên. Mục đích chia dữ liệu để tăng sự đa dạng cho dữ liệu để quá trình huấn luyện được diễn ra tốt hơn

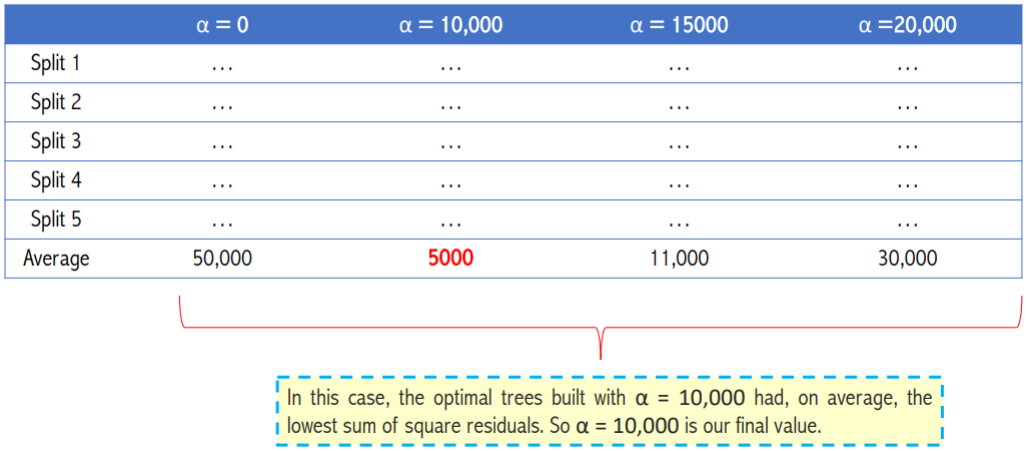


Hình 31: Phân chia data thành các fold



Hình 32: Quá trình phân chia data

- Lập bảng giá trị, tính giá trị trung bình theo từng giá trị α và chọn giá tri α tốt nhất



Hình 33: Chọn *alpha* nhỏ nhất là giá trị tốt nhất