AI VIET NAM – COURSE 2024

Nguyễn Hữu Hoàng Long

Ngày 16 tháng 3 năm 2024

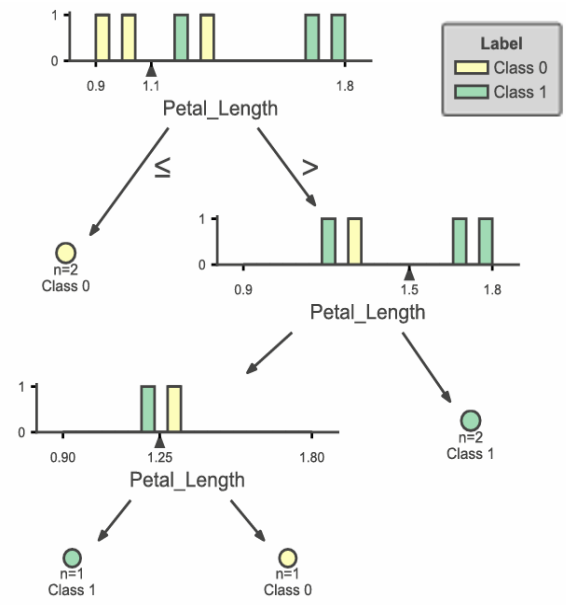
1 Giới thiệu

Random Forest và AdaBoost là hai thuật toán học máy được phát triển dựa trên mục tiêu chung là khắc phục những hạn chế của thuật toán Decision Tree. Cả hai đều là các phương pháp ensemble learning, kết hợp nhiều mô hình học máy để tạo ra một mô hình tổng thể mạnh mẽ hơn.

1.1 Decision Tree

Decision Tree là một thuật toán học máy cơ bản được sử dụng rộng rãi cho cả hai nhiệm vụ phân loại (classification) và hồi quy (regression). Thuật toán này hoạt động bằng cách tạo ra một mô hình mô phỏng cấu trúc cây, với các nhánh và nút tương ứng với các quyết định và kết quả dự đoán. Việc xây dựng cây dựa trên các tiêu chí đánh giá thông tin như entropy hoặc Gini, giúp lựa chọn các đặc điểm quan trọng nhất để phân loại dữ liệu.

- Bộ dữ liệu Iris nổi tiếng là một ví dụ điển hình để minh họa cho việc áp dụng Decision Tree. Bộ dữ liệu này bao gồm thông tin về ba loài hoa Iris khác nhau (setosa, versicolor và virginica) với bốn đặc điểm: chiều dài cánh hoa, chiều rộng cánh hoa, chiều dài đài hoa và chiều rộng đài hoa.
- Sử dụng Decision Tree, ta có thể xây dựng mô hình phân loại các loài hoa Iris. Mô hình này sẽ phân tích các đặc điểm của hoa và đưa ra dự đoán về loài hoa tương ứng.



Hình 1. Ví dụ cấu trúc Decision Tree thực hiện Training trên Dataset Iris

Hạn chế của Decision Tree:

- Mô hình có thể tập trung vào một đặc điểm nhất định, dẫn đến việc bỏ qua các đặc điểm quan trọng khác. Hình ảnh minh họa cho thấy mô hình Decision Tree được xây dựng trên bộ dữ liệu Iris đạt hiệu quả phân loại cao trên tập dữ liệu. Tuy nhiên, mô hình lại quá tập trung vào đặc điểm "chiều dài cánh hoa" (Petal_Length) mà bỏ qua các đặc điểm khác.
- Mô hình cũng có thể trở nên cứng nhắc và khó thích nghi với dữ liệu mới.

Hệ quả

- Hạn chế khả năng khai thác hiệu quả các đặc điểm khác: Mô hình không thể tận dụng đầy đủ thông tin từ các đặc điểm còn lại để nâng cao hiệu quả phân loại.
- Giảm độ tin cậy của mô hình: Khi mô hình phụ thuộc quá nhiều vào một đặc điểm, nó sẽ trở nên nhạy cảm với sự thay đổi của đặc điểm đó, dẫn đến việc giảm độ tin cậy của dự đoán.

Kết luận

Decision Tree là một thuật toán học máy hữu ích và dễ sử dụng. Tuy nhiên, để nâng cao hiệu quả và độ tin cậy của mô hình, cần lưu ý đến những hạn chế của thuật toán này và áp dụng các kỹ thuật phù hợp để khắc phục, ví dụ như sử dụng kỹ thuật "cắt tỉa" để loại bỏ các nhánh không quan trọng hoặc kết hợp Decision Tree với các thuật toán khác.

1.2 Random Forest

Để khắc phục những hạn chế của Decision Tree, các nhà khoa học đã phát triển thuật toán Random Forest. Random Forest là một thuật toán học máy mạnh mẽ được sử dụng cho cả hai nhiệm vụ phân loại (classification) và hồi quy (regression). Thuật toán này hoạt động dựa trên việc tạo ra một tập hợp gồm nhiều Decision Tree (cây quyết định) và kết hợp dự đoán từ các cây này để đưa ra kết quả cuối cùng.

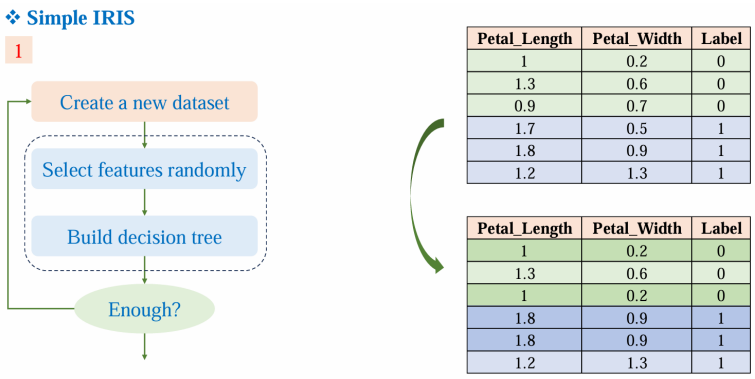
- **Tính ngẫu nhiên:** Random Forest sử dụng tính ngẫu nhiên để chọn lựa các tập hợp con dữ liệu và các đặc điểm để xây dựng nhiều Cây Quyết Định.
- **Tính đa dạng:** Nhờ tính ngẫu nhiên, mỗi Cây Quyết Định trong Random Forest sẽ có cấu trúc và tập dữ liệu riêng biệt, tạo nên sự đa dạng cho khu rừng.
- **Kết hợp dự đoán:** Dự đoán từ các Cây Quyết Định được kết hợp bằng phương pháp bỏ phiếu hoặc trung bình để đưa ra kết quả cuối cùng.

Cách hoạt động của RF-Classification

1. **Tạo dataset mới:** Thuật toán tạo ra một dataset mới từ dataset ban đầu bằng phương pháp bootstrap.
2. **Lựa chọn feature:** Các feature được lựa chọn một cách ngẫu nhiên từ tập feature của dataset mới.
3. **Xây dựng Decision Tree:** Một Decision Tree được xây dựng dựa trên các feature được lựa chọn và dataset mới.
4. **Lặp lại:** Các bước trên được lặp lại nhiều lần cho đến khi đạt được số lượng Decision Tree mong muốn.

Ví dụ

- Ví dụ troang bộ dữ liệu sau, thuật toán lựa chọn ngẫu nhiên các sample tạo thành một dataset mới như bên dưới

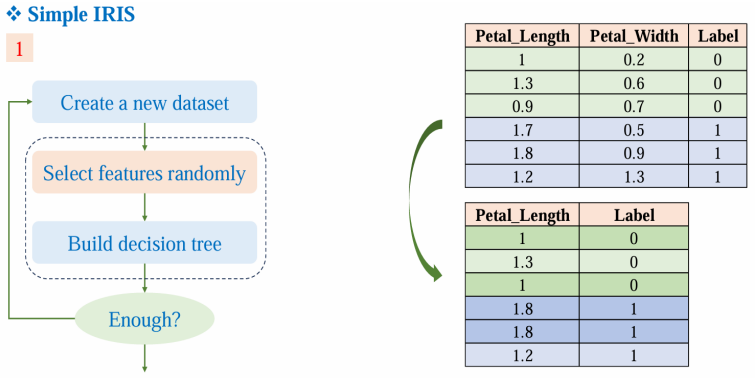


Petal_Length	Petal_Width	Label
1	0.2	0
1.3	0.6	0
0.9	0.7	0
1.7	0.5	1
1.8	0.9	1
1.2	1.3	1

Petal_Length	Petal_Width	Label
1	0.2	0
1.3	0.6	0
1	0.2	0
1.8	0.9	1
1.8	0.9	1
1.2	1.3	1

Hình 2. Thực hiện tạo ra dataset mới bằng phương pháp Bootstrap

- Sau đó, thuật toán thực hiện chọn ngẫu nhiên các feature với số lượng cố định, ở trường hợp trong bài là 1 feature duy nhất. Ta có tập dataset mới như sau:

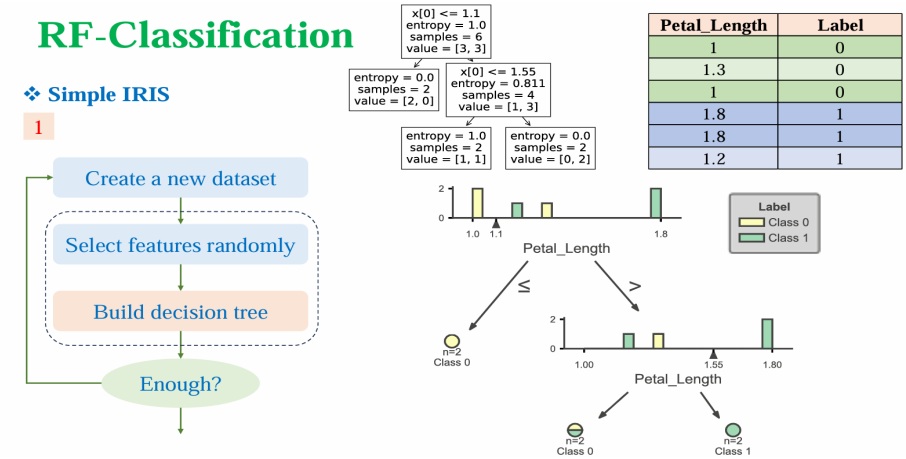


Petal_Length	Petal_Width	Label
1	0.2	0
1.3	0.6	0
0.9	0.7	0
1.7	0.5	1
1.8	0.9	1
1.2	1.3	1

Petal_Length	Label
1	0
1.3	0
1	0
1.8	1
1.8	1
1.2	1

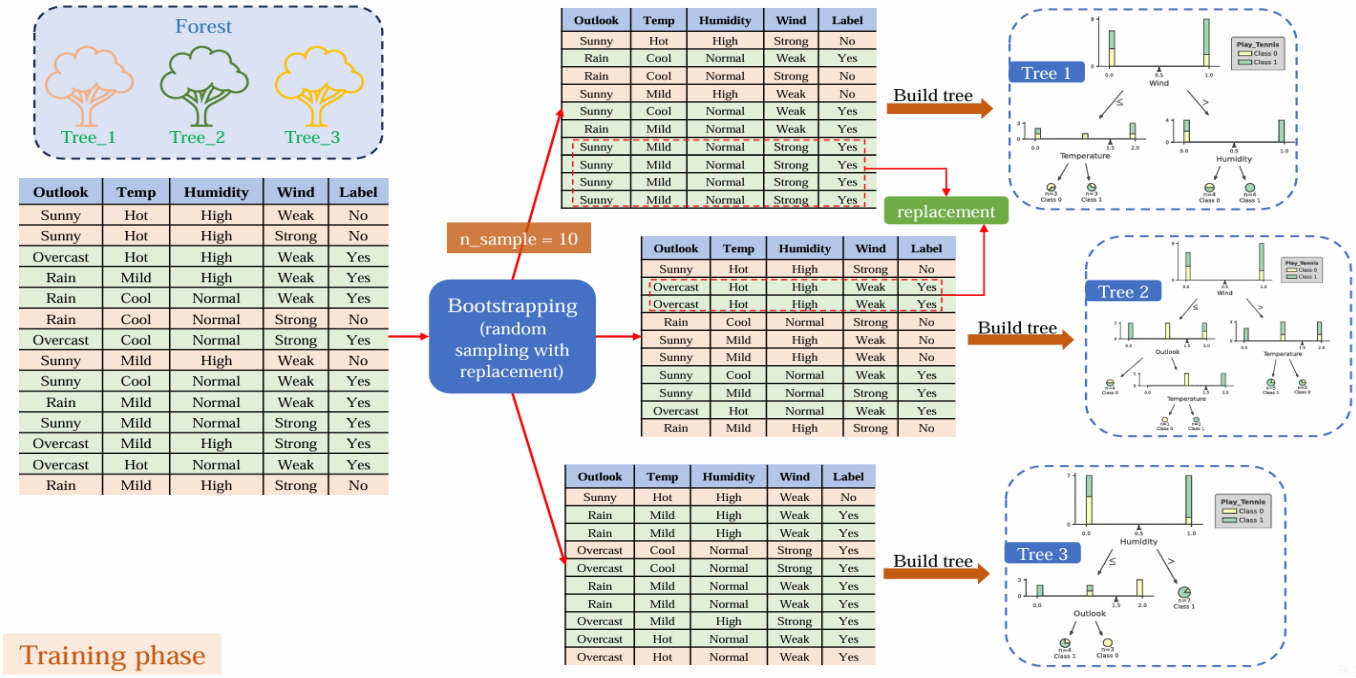
Hình 3. Dataset mới được tạo thành gồm 1 feature Petal_Length

- Thực hiện xây dựng Decision Tree dựa trên tập dữ liệu vừa tạo ta sẽ có cây như bên dưới. Tiếp tục ta kiểm tra điều kiện và lặp lại quá trình như trên cho đến khi đủ số lượng cây cho trước.

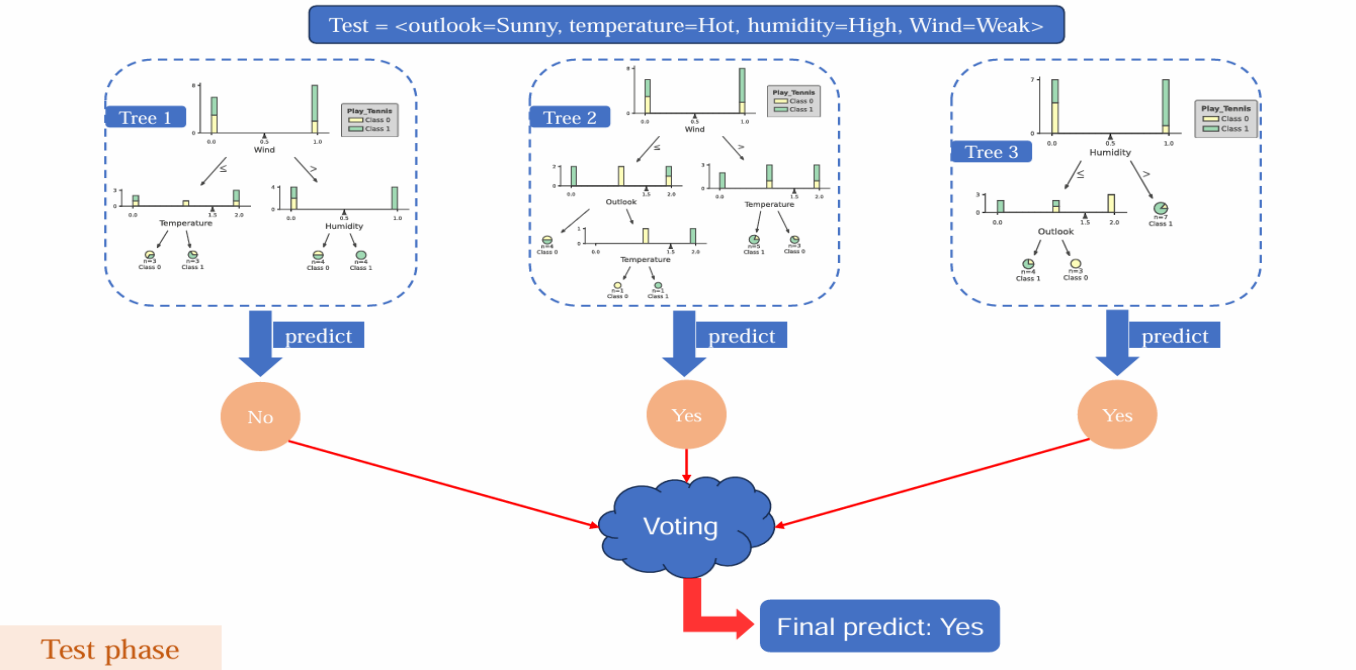


Petal_Length	Label
1	0
1.3	0
1	0
1.8	1
1.8	1
1.2	1

Hình 4. Thực hiện tạo Decision Tree trên Dataset mới



Hình 5. Tương tự với các bước trên ta tạo được 3 Decision Tree tương ứng với 3 Dataset mới



Hình 6. Thực hiện dự đoán trên mỗi cây với dữ liệu mới và thực hiện cơ chế voting để cho kết quả cuối cùng

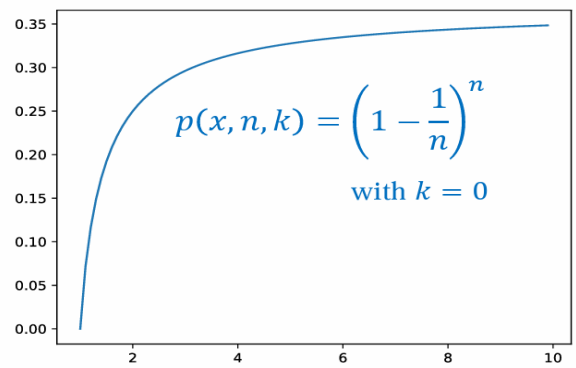
Cách hoạt động của RF-Regression

1. **Tạo tập hợp Decision Tree:** Random Forest tạo ra một tập hợp gồm nhiều Decision Tree được xây dựng độc lập với nhau.

2. **Xây dựng Decision Tree:** Mỗi Decision Tree được xây dựng dựa trên một tập con dữ liệu được chọn ngẫu nhiên và một tập hợp các feature được chọn ngẫu nhiên.
3. **Dự đoán:** Mỗi Decision Tree đưa ra dự đoán cho giá trị mục tiêu của dữ liệu.
4. **Kết hợp dự đoán:** Các dự đoán từ các Decision Tree được kết hợp bằng phương pháp trung bình để đưa ra dự đoán cuối cùng cho giá trị mục tiêu.

Bernouli Random Variables

1. **Giới thiệu**
   - **Định lí Bernouli:** Mô tả xác suất thành công trong một thí nghiệm Bernoulli, nơi chỉ có hai kết quả có thể xảy ra (thành công hoặc thất bại) với xác suất cố định.
   - **Bootstrap:** Kỹ thuật lấy mẫu lại với thay thế được sử dụng trong học máy để tạo ra các tập dữ liệu con từ một tập dữ liệu ban đầu.
2. **Xác suất mất dữ liệu**
   - **Mô hình:** Giả sử tung đồng xu với xác suất sấp và ngửa là như nhau.
   - **Xác suất k lần ngửa:** Theo định lý Bernoulli, xác suất để khi tung n lần xảy ra k lần mặt ngửa là:
   $$p(x, n, k) = C_n^k \frac{1}{n^k}(1 - \frac{1}{n})^k$$
   - **Với trường hợp bằng 0:** Xác suất để không có lần nào ngửa là:
   $$p(x, n, k = 0) = (1 - \frac{1}{n})^n$$
3. **Áp dụng cho Bootstrap:**
   - **Chọn feature:** Xác suất để chọn ngẫu nhiên một feature cụ thể là $\frac{1}{n}$
   - **Xác suất feature không được chọn:**Theo định lý Bernoulli, xác suất để feature đó không được chọn xuyên suốt quá trình bootstrap là:
   $$p(x, n, k = 0) = (1 - \frac{1}{n})^n$$
4. **Phân tích:**



Hình 7. Đồ thị xác suất Bernouli với $k = 0$

- **Đồ thị hàm số:** Biểu thị hàm số trên đồ thị hàm số, ta thấy xác suất dần về 0.3.
- **Ước tính:** Trong các bài toán sử dụng Random Forest, người ta thường ước tính rằng tầm 30% dữ liệu sẽ bị mất đi trong quá trình bootstrap.

$$p(x, n, k = 0) = (1 - \frac{1}{n})^n$$

5. **Kết luận:**

- Bootstrap có thể dẫn đến việc mất dữ liệu, với tỷ lệ ước tính khoảng 30% trong trường hợp các feature có xác suất được chọn là như nhau.
- Việc mất dữ liệu có thể ảnh hưởng đến hiệu quả của mô hình học máy, do đó cần cân nhắc kỹ lưỡng khi sử dụng kỹ thuật bootstrap.

2 AdaBoost

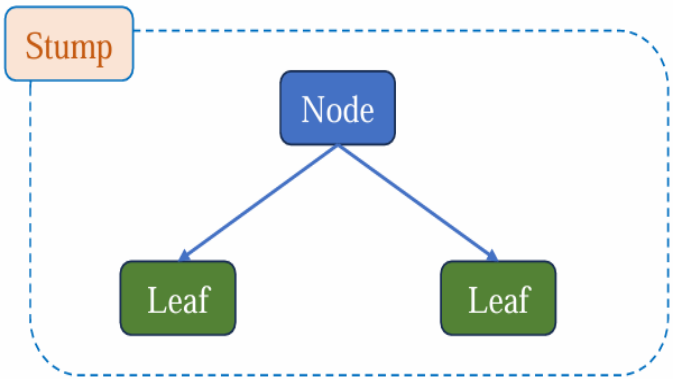
2.1 Giới thiệu

AdaBoost, hay "Adaptive Boosting", là một thuật toán học máy thuộc nhóm Ensemble Learning. Nó kết hợp nhiều mô hình học yếu (weak learners) để tạo ra một mô hình học mạnh (strong learner). AdaBoost là thuật toán học máy mạnh mẽ được sử dụng cho cả phân loại và hồi quy. Hiệu quả của AdaBoost phụ thuộc vào chất lượng dữ liệu được sử dụng để huấn luyện.

2.2 Stump là gì?

Ta sẽ làm quen với khái niệm Stump:

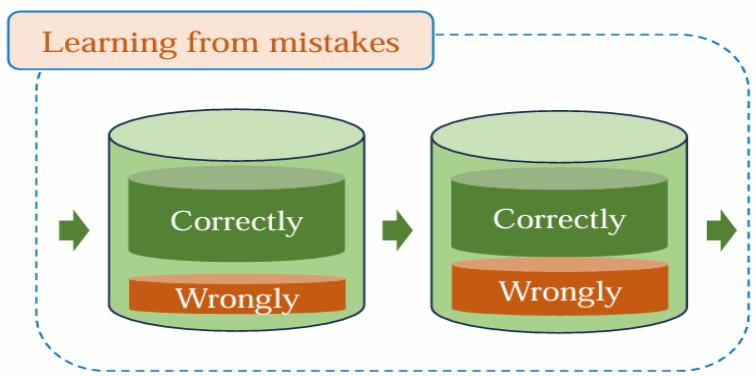
- **Stump** thường được gọi là một "Decision Stump", đó là một mô hình cây quyết định với độ sâu tối đa là 1. Nói cách khác, nó chỉ gồm một nút gốc (root node) và hai nút con (child nodes).
- Mỗi **Decision Stump** đưa ra quyết định dựa trên giá trị của một đặc trưng duy nhất. Do đơn giản và dễ hiểu, **Decision Stump** thường được sử dụng như là các **weak learners** trong các thuật toán ensemble như AdaBoost.



Hình 8. Cấu trúc của Stump

2.3 Motivation

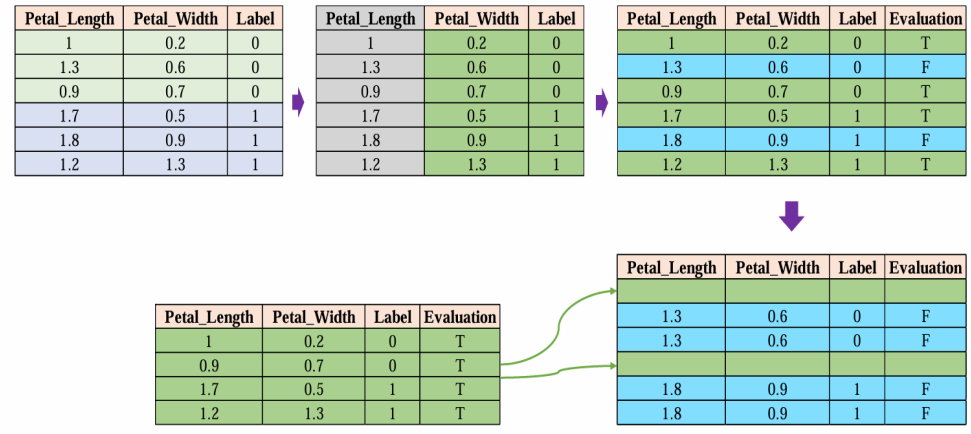
Bắt nguồn từ ý tưởng khắc phục điểm yếu từ Decision Tree, AdaBoost được xây dựng dựa trên việc xây dựng các Stump phía sau khắc phục dần các dự đoán sai từ các Stump phía trước.



Hình 10. Khắc phục dần các sample dự đoán sai.

2.4 Xây dựng dữ liệu

Ví dụ sau đây sẽ minh họa cách xây dựng dữ liệu cho thuật toán AdaBoost.



Petal_Length	Petal_Width	Label
1	0.2	0
1.3	0.6	0
0.9	0.7	0
1.7	0.5	1
1.8	0.9	1
1.2	1.3	1

Petal_Length	Petal_Width	Label
1	0.2	0
1.3	0.6	0
0.9	0.7	0
1.7	0.5	1
1.8	0.9	1
1.2	1.3	1

Petal_Length	Petal_Width	Label	Evaluation
1	0.2	0	T
1.3	0.6	0	F
0.9	0.7	0	T
1.7	0.5	1	T
1.8	0.9	1	F
1.2	1.3	1	T

Petal_Length	Petal_Width	Label	Evaluation
1	0.2	0	T
0.9	0.7	0	T
1.7	0.5	1	T
1.2	1.3	1	T

Petal_Length	Petal_Width	Label	Evaluation
1.3	0.6	0	F
1.3	0.6	0	F
1.8	0.9	1	F
1.8	0.9	1	F

Hình 11. Mô tả cách xây dựng dữ liệu

1. Bảng dữ liệu ban đầu bao gồm các thuộc tính và nhãn phân loại. Và có xác suất được chọn là như nhau.

2. Dữ liệu được sử dụng để huấn luyện và dự đoán bằng một mô hình học máy.

3. Cột mới "Evaluation" được thêm vào, với "T" (True) cho các mẫu được dự đoán đúng và "F" (False) cho các mẫu được dự đoán sai.

4. Dữ liệu sau khi dự đoán kết hợp với thông tin "Evaluation" được sử dụng để tạo tập dữ liệu mới cho mô hình tiếp theo (Stump). Thay đổi xác suất được chọn của các sample dựa vào thông tin đó.

5. Tần suất xuất hiện của các mẫu được phân loại sai trong tập dữ liệu mới cao hơn so với các mẫu được phân loại đúng.

Mục đích

- AdaBoost sử dụng chiến lược "học tập theo hướng tăng cường" để xây dựng mô hình tổng thể. Tập trung vào các mẫu khó phân loại để nâng cao hiệu quả của mô hình tổng thể.

- Mỗi mô hình con (Stump) được huấn luyện trên tập dữ liệu được điều chỉnh để tập trung vào các mẫu khó phân loại.
- Trọng số được gán cho các mô hình con dựa trên hiệu suất của chúng.
- Mô hình tổng thể kết hợp các dự đoán từ các mô hình con với trọng số tương ứng.

Phân tích mức độ nghiêm tọng của lỗi phân loại trong quá trình tạo tập dữ liệu mới cho Stump

Việc xây dựng tập dữ liệu mới cho các Stump đóng vai trò quan trọng trong thuật toán AdaBoost. Trong quá trình tạo dataset mới cho các stump, ta có thể bắt gặp các trường hợp đối với các sample bị phân loại sao như sau:

- **Trường hợp 1:** Các sample được phân loại sai ở Stump sau khác biệt so với các sample được phân loại bởi Stump đằng trước.

Petal_Length	Petal_Width	Label
1	0.2	0
1.3	0.6	0
0.9	0.7	0
1.7	0.5	1
1.8	0.9	1
1.2	1.3	1

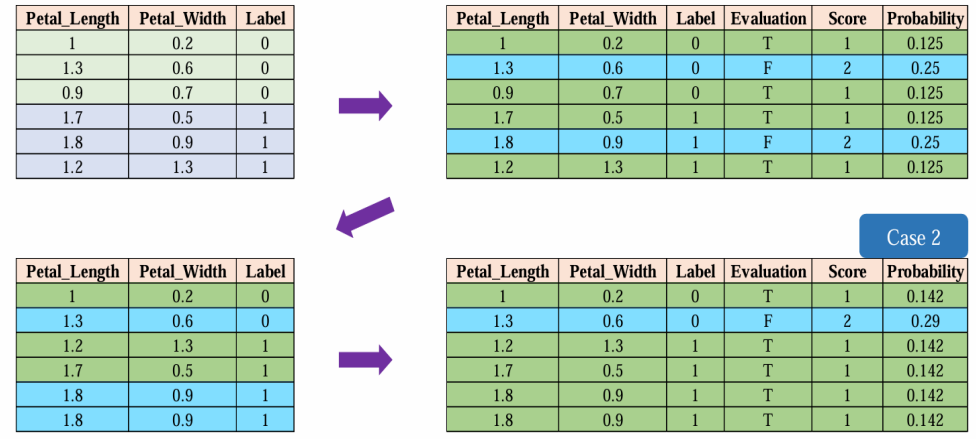
Petal_Length	Petal_Width	Label	Evaluation	Score	Probability
1	0.2	0	T	1	0.125
1.3	0.6	0	F	2	0.25
0.9	0.7	0	T	1	0.125
1.7	0.5	1	T	1	0.125
1.8	0.9	1	F	2	0.25
1.2	1.3	1	T	1	0.125

Petal_Length	Petal_Width	Label
1	0.2	0
1.3	0.6	0
1.2	1.3	1
1.7	0.5	1
1.8	0.9	1
1.8	0.9	1

Petal_Length	Petal_Width	Label	Evaluation	Score	Probability
1	0.2	0	T	1	0.142
1.3	0.6	0	T	1	0.142
1.2	1.3	1	F	2	0.29
1.7	0.5	1	T	1	0.142
1.8	0.9	1	T	1	0.142
1.8	0.9	1	T	1	0.142

Hình 12. Dữ liệu ở trường hợp 1

- **Trường hợp 2:** Các sample được phân loại sai ở Stump sau lặp lái đối với các sample được phân loại bởi Stump đằng trước.



Petal_Length	Petal_Width	Label
1	0.2	0
1.3	0.6	0
0.9	0.7	0
1.7	0.5	1
1.8	0.9	1
1.2	1.3	1

Petal_Length	Petal_Width	Label	Evaluation	Score	Probability
1	0.2	0	T	1	0.125
1.3	0.6	0	F	2	0.25
0.9	0.7	0	T	1	0.125
1.7	0.5	1	T	1	0.125
1.8	0.9	1	F	2	0.25
1.2	1.3	1	T	1	0.125

Petal_Length	Petal_Width	Label
1	0.2	0
1.3	0.6	0
1.2	1.3	1
1.7	0.5	1
1.8	0.9	1
1.8	0.9	1

Petal_Length	Petal_Width	Label	Evaluation	Score	Probability
1	0.2	0	T	1	0.142
1.3	0.6	0	F	2	0.29
1.2	1.3	1	T	1	0.142
1.7	0.5	1	T	1	0.142
1.8	0.9	1	T	1	0.142
1.8	0.9	1	T	1	0.142

Hình 13. Dữ liệu ở trường hợp 2

Nhận xét:

- Để nâng cao hiệu quả phân loại của AdaBoost bằng cách tập trung vào dữ liệu được phân loại lỗi. Ta phải thực hiện khuất đại xác suất của các mẫu được phân loại lỗi để thu hút sự chú ý của Stump tiếp theo. Do đó có tính kế thừa giữa các dataset của các Stump liên tiếp.
- Ta nhận xét rằng phân loại sai các sample ở trường hợp 2 nghiêm trọng hơn só với trường hợp 1. Nên theo trực quan ta có thể ước lượng rằng score của sample phân loại sai ở trường hợp 2 phải cao hơn so với trường hợp 1. Score chỉ mức độ nghiêm trọng của sample. Ta sẽ thực hiện tính toán xác suất dựa vào score phía trước

Scaled weight đối với các sample được phân loại sai

Đối với độ lỗi (error) khác nhau của mô hình thì scaled weight sẽ có giá trị cao hoặc thấp:

- **Với error $<$ 0.5:** Đối với các mô hình tốt thì scaled weight sẽ tăng đáng kể.
- **Với error $>$ 0.5:** Đối với các mô hình không tốt thì phải thực hiện đổi lại việc ưu tiên tập trung vào các sample được phân loại tốt. Dẫn đến xác suất sẽ giảm (scaled weight $<$ 1) đối với các sample phân loại sai và tăng đối với các sample phân loại đúng. Nhằm đảm bảo qua các bước selection thì độ error của các stump sau luôn không tăng.

2.5 Nên tăng bao nhiêu đối với scaled weight

Trước khi thực hiện lựa chọn hàm phù hợp cho scaled weight, ta có hai nhận xét sau

- **Scaled weight nhỏ:** Dẫn đến bảng dữ liệu sau sẽ gần giống với bảng dữ liệu của Stump trước.
- **Scaled weight lớn:** Dẫn đến bảng dữ liệu sau sẽ tập trung hơn vào các phần lỗi.

Đối với các mô hình tốt có độ lỗi (bad error) lớn 0.5 và các mô hình xấu có độ lỗi (good error) bé hơn 0.5, ta sẽ lựa chọn hàm số cho scaled weight thỏa mãn hai điều kiện sau:

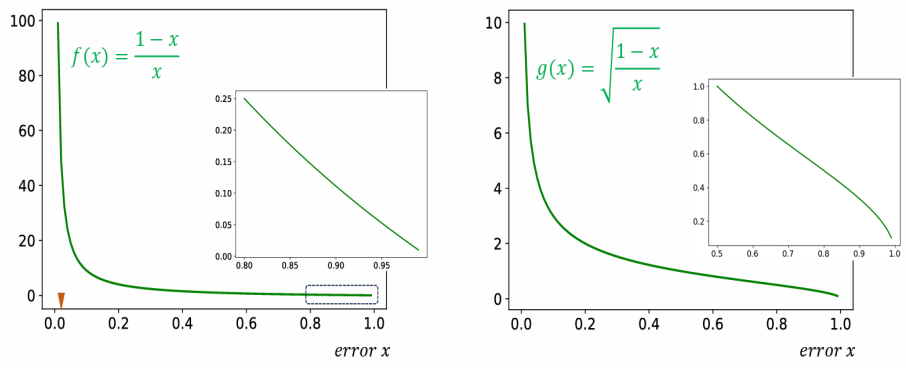
- **Đối với good error:** scaled weight $>$ 1.
- **Đối với bad error:** scaled weight $<$ 1.

Sau quá trình nghiên cứu, người ta đã chọn được các hàm số phù hợp cho hai trường hợp khác nhau thỏa mãn 2 điều kiện trên:

- **Đối với incorrect samples:** Scaled weight là hàm

$$g(x) = \sqrt{\frac{1-x}{x}}$$

:

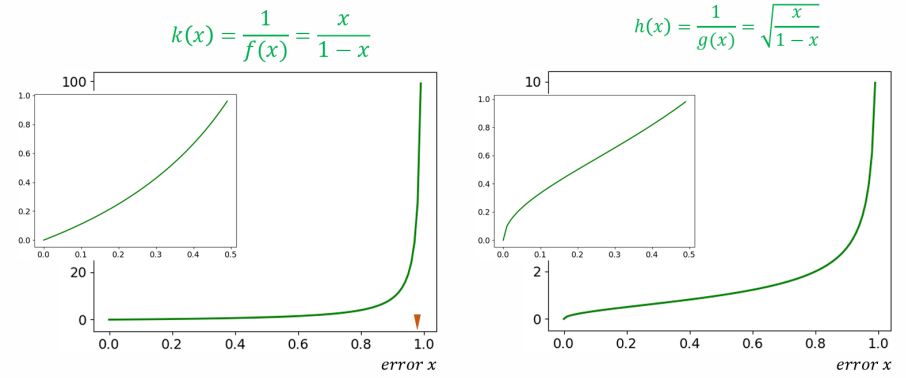


Hình 14. Đồ thị hàm $g(x)$

- **Đối với correct samples:** Scaled weight là hàm

$$h(x) = \sqrt{\frac{x}{1-x}}$$

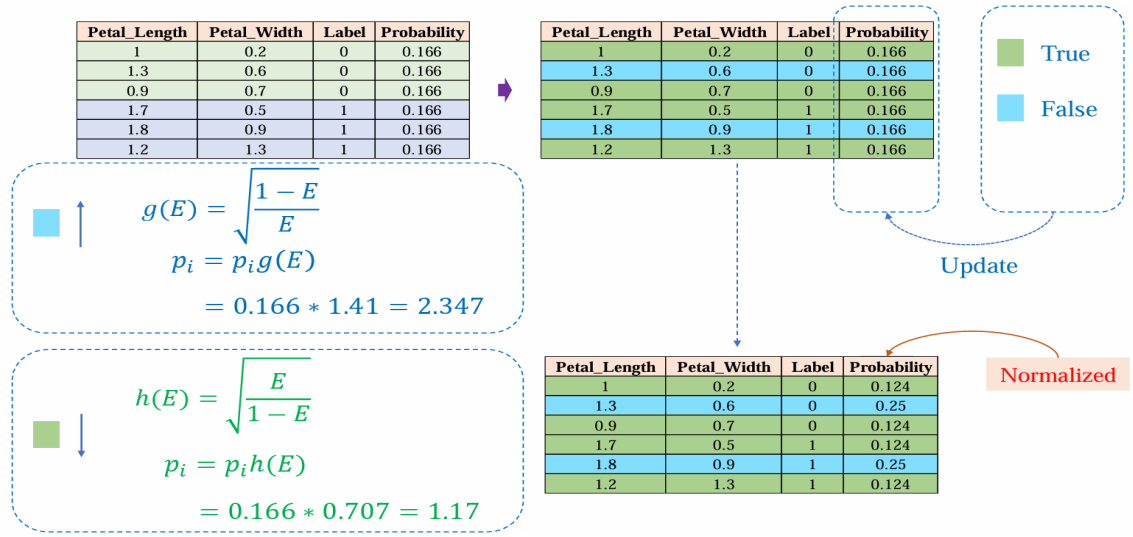
:



Hình 15. Đồ thị hàm $h(x)$

Căn bậc hai đóng vai trò như scale dữ liệu. Scale dữ liệu nhằm mục đích bảo toàn mục tiêu ban đầu là chỉ lựa chọn nhiều chứ không phải là tất cả. Ví dụ như nếu scale weight là 100 thì ta chắc chắn chọn những sample đó.

Một số ví dụ cụ thể



Hình 16.Ví dụ cụ thể

- Đầu tiên thực hiện phân loại trên Dataset ban đầu.
- Dựa vào kết quả phân loại. Tính error sau khi thực hiện dự đoán. Ở đây ta có thể thấy error $<$ 0.5 nên thực hiện tính scaled weight đối với các incorrect sample và correct sample. Scaled weight màu xanh dương tương ứng với incorrect sample và scaled weight màu xanh lá tương ứng với correct sample.
- Thực hiện tính probability mới dựa trên scaled weight và probability cũ.
- Thực hiện normalize để đưa về tổng 1.