Đại học Quốc Gia Thành phố Hồ Chí Minh
Đại học Công nghệ Thông tin

NHÓM 2 – LỚP CS115.M11

BÁO CÁO ĐỀ TÀI

ENSEMBLE LEARNING WITH RANDOM FORESTS

| Giáo viên hướng dẫn |
GV.Lương Ngọc Hoàng
Môn học: Toán cho Khoa học máy tính

Thành phố Hồ Chí Minh - 2021

DANH SÁCH THÀNH VIÊN

STT	Họ tên	MSSV	Gmail
1	Mai Văn Thiên Phước	20521776	20521776@gm.uit.edu.vn
2	Lâm Minh Tuấn	20520843	20520843@gm.uit.edu.vn
3	Đậu Văn Nam	20521626	20521626@gm.uit.edu.vn
4	Phạm Xuân Hoàng	20520519	20520519@gm.uit.edu.vn
5	Trần Văn Lực	20521587	20521587@gm.uit.edu.vn

TỔNG QUAN ĐỀ TÀI

Nhắc đến Machine Learning thì ta không thể không nhắc đến Decision Tree hay còn được biết đến với tên gọi Cây quyết định. Decision Tree 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. Tuy nhiên Cây quyết định có nhiều nhược điểm dẫn đến kết quả cuối cùng không chính xác. Để khắc phục các nhược điểm của cây quyết định, người ta sử dụng phương pháp Ensemble Learning (học kết hợp nhiều mô hình với nhau) từ đó ra đời thuật toán Random Forest hay còn gọi là Rừng ngẫu nhiên để đưa ra kết quả một cách chính xác hơn dựa trên ý tưởng rất đơn giản: "Một cây làm chẳng lên non – Ba cây chụm lại nên hòn núi cao". Random Forest sử dụng nhiều kết quả của các cây ngẫu nhiên khác nhau để tổng hợp kết quả cuối cùng dẫn đến kết quả cuối cùng sẽ chính xác hơn rất nhiều so với Decision Tree. Tuy Random Forest dựa trên một ý tưởng rất đơn giản nhưng lại đang là một trong những thuật toán Machine Learning xuất sắc nhất mà chúng ta đang có hiện tại.

MỤC LỤC

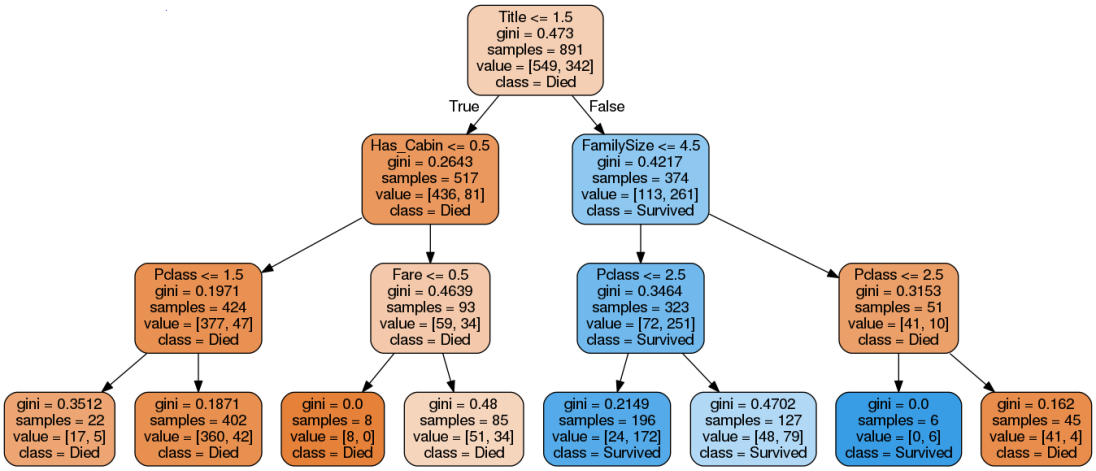
TÀI LIỆU THAM KHẢO

1. Random Forests From Scratch (carbonati.github.io)_)
2. https://machinelearningcoban.com/2018/01/14/id3/)
3. Random Forest algorithm — Machine Learning cho dữ liệu dạng bảng (machinelearningcoban.com)
4. https://stats.stackexchange.com/questions/88980/why-on-average-does-each-bootstrap-sample-contain-roughly-two-thirds-of-observat
5. https://en.wikipedia.org/wiki/Information_gain_in_decision_trees

CHƯƠNG I: DECISION TREE

1. Khái niệm

Decision Tree (Cây quyết định) 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 tượng có 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.



Hình 1: Minh họa mô hình Decision Tree

2. Entropy

Công cụ đo lường được sử dụng nhiều nhất để xây dựng cây quyết định là Entropy và hệ số Gini. Ở đây chúng ta sẽ tập trung vào Entropy, là một đại lượng đo lường sự không chắc chắn (đôi khi còn gọi là sự hỗn tạp), để đo lường Entropy chúng ta dùng công thức như sau:

Với p_j là xác suất đối tượng thuộc lớp j.

$$H(X) = -\sum_j p_j \log p_j$$

Hãy đến với một ví dụ để tìm hiểu rõ hơn ý nghĩa của Entropy:

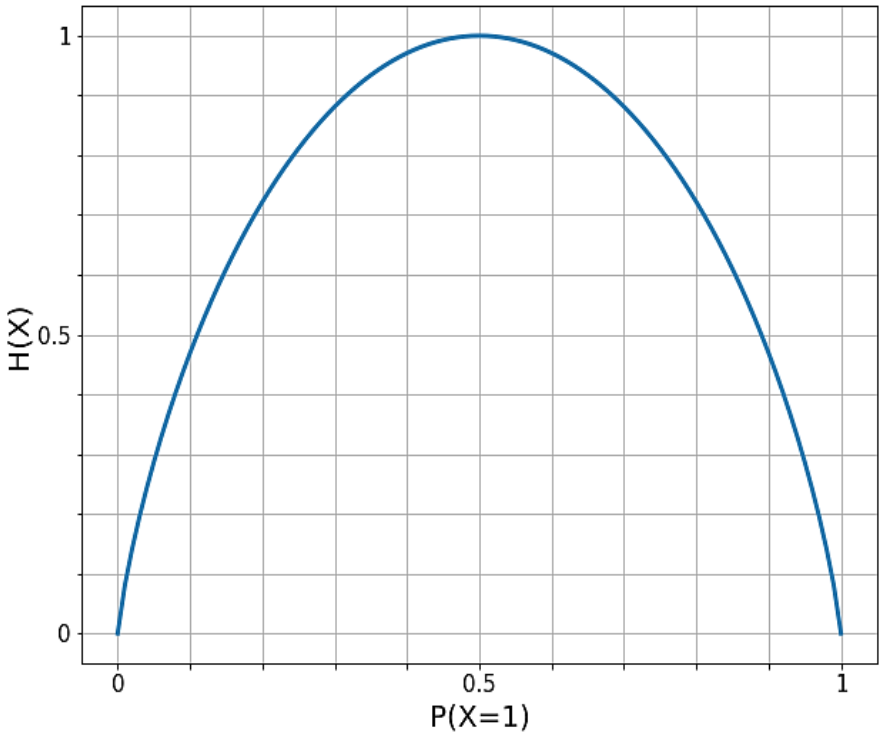
Ta tung một đồng xu 2 lần. Xét 2 trường hợp:

- Trường hợp 1: Cả 2 lần tung đều là mặt ngửa hoặc mặt úp. Ta có Entropy như sau:

$$-1 * \log_2(1) - 0 * \log_2(0) = 0$$

- Trường hợp 2: 1 lần tung ra mặt ngửa và lần còn lại ra mặt úp. Ta có Entropy như sau:

$$- 0.5 * \log_2(0.5) – 0.5 * \log_2(0.5) = 1$$



Hình 2: Đồ thị biểu diễn sự thay đổi của hàm Entropy ứng với các xác suất mà đối tượng thuộc các lớp khác nhau

Đồ thị trên biểu diễn sự thay đổi của hàm Entropy. Ta có thể thấy rằng, Entropy đạt tối đa khi xác suất xảy ra trên cả 2 lớp bằng nhau. Qua ví dụ và đồ thị trên ta có thể thấy được Entropy càng nhỏ, dữ liệu càng được phân chia rõ ràng hơn vào các lớp và khi Entropy càng lớn, dữ liệu càng được phân chia đồng đều hơn vào các lớp. Hay nói cách khác, Entropy càng lớn thì độ ngẫu nhiên của các biến rời rạc càng cao.

3. Information Gain

Information Gain dựa trên sự giảm của hàm Entropy khi tập dữ liệu được phân chia trên một thuộc tính. Information Gain được tính như sau

$$\mathrm{H}(T|a) = \sum_{v \in vals(a)} \frac{|S_a(v)|}{|T|} \cdot \mathrm{H}\left(S_a(v)\right).$$

$$IG(T,a) = \mathrm{H}(T) - \mathrm{H}(T|a)$$

Với $S_a(v) = \{x \in T | x_a = v\}$. Chúng ta luôn muốn chia dữ liệu sao cho Information Gain đạt giá trị lớn nhất, bởi vì khi đó Entropy của tập con đạt giá trị nhỏ nhất đồng nghĩa với việc dữ liệu sẽ được phân bố một cách rõ ràng hơn vào các lớp. Nếu đây là giá trị Information Gain cao nhất trong số tất cả các phân chia có thể, chúng ta sẽ chia dữ liệu vào nút con tương ứng với các giá trị của thuộc tính đó rồi tiếp tục áp dụng phương pháp này cho mỗi nút con cho đến khi đạt độ sâu mong muốn. Việc chọn ra thuộc tính tốt nhất ở mỗi bước như thế này được gọi là cách chọn *greedy* (tham lam). Cách chọn này có thể không phải là tối ưu, nhưng trực giác cho chúng ta thấy rằng cách làm này sẽ gần với cách làm tối ưu. Ngoài ra, cách làm này khiến cho bài toán cần giải quyết trở nên đơn giản hơn. Đây chính là thuật toán ID3 (Iterative Dichotomiser 3). Chúng ta đã biết cách xây dựng một cây quyết định duy nhất, từ đó sẽ đến với thuật toán Random Forest là tập hợp của nhiều cây quyết định ngẫu nhiên.

CHƯƠNG II: ENSEMBLE LEARNING

1. Khái niệm

Giả sử, ta đặt một câu hỏi phức tạp cho hàng nghìn người được chọn lọc ngẫu nhiên, và tổng hợp các câu hỏi trả lời của họ. Trong nhiều trường hợp, ta sẽ thấy câu trả lời có được thường tốt hơn câu trả lời của một chuyên gia. Hiện tượng này được gọi là *Trí tuệ đám đông*. Tương tự, nếu kết hợp một nhóm bộ dự đoán (chẳng hạn như các bộ phận phân loại và hồi quy), dự đoán tổng hợp nhận được thường tốt hơn dự đoán riêng lẻ của bộ dự đoán tốt nhất. Một nhóm các bộ phân loại được gọi là Ensemble, và vì thế kỹ thuật còn được gọi là Ensemble Learning. Một thuật toán Ensemble Learning được gọi là *phương pháp Ensemble*.

2. Phân loại

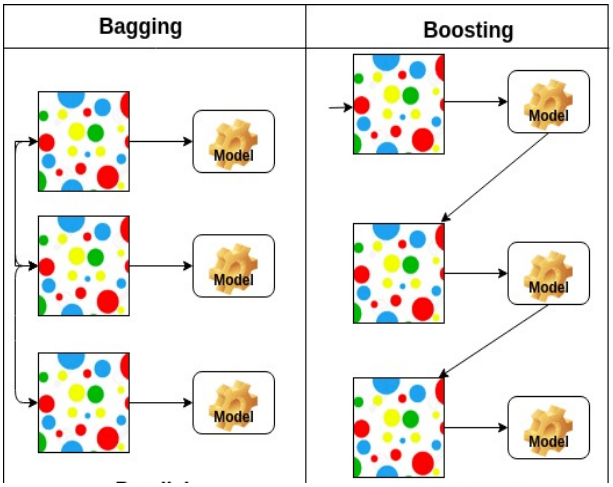
Các phương pháp Ensemble Learning được chia thành 3 loại sau đây:

- Bagging (đóng bao)

  Bagging là viết tắt của tổng hợp bootstrap (bootstrap aggregation). Một cách để giảm phương sai của ước tính là tính trung bình nhiều ước tính cùng nhau. Để tổng hợp kết quả đầu ra của những người học cơ sở, tính năng đóng gói sử dụng "bỏ phiếu để phân loại" (voting for classification) và "lấy trung bình để hồi quy" (averaging for regression).

- Boosting (tăng cường)

  Boosting đề cập đến một nhóm các thuật toán có thể chuyển đổi người học yếu thành người học mạnh. Các dự đoán sau đó được kết hợp thông qua biểu quyết đa số có trọng số (classification) hoặc tổng có trọng số (regression) để đưa ra dự đoán cuối cùng.



Hình 3: Mô hình Bagging và Boosting

- Stacking (Xếp chồng)

  Stacking là một kỹ thuật học tập tổng hợp kết hợp nhiều phân loại hoặc mô hình hồi quy thông qua bộ phân loại siêu phân loại hoặc hồi quy siêu hồi quy. Các mô hình cấp cơ sở được đào tạo dựa trên một tập hợp đào tạo hoàn chỉnh, sau đó mô hình meta được đào tạo dựa trên các kết quả đầu ra của mô hình cấp cơ sở (base level model) dưới dạng các tính năng (features).

CHƯƠNG III: RANDOM FOREST

1. Khái niệm

Random Forest (Rừng ngẫu nhiên) là một Ensemble Learning để phân loại, hồi quy được phát triển bởi Leo Breiman (2001). Breiman cũng là đồng tác giả của phương pháp Classification Tree và Regression Tree (CART). Random Forest là phương pháp cải tiến của tổng hợp bootstrap (bagging).

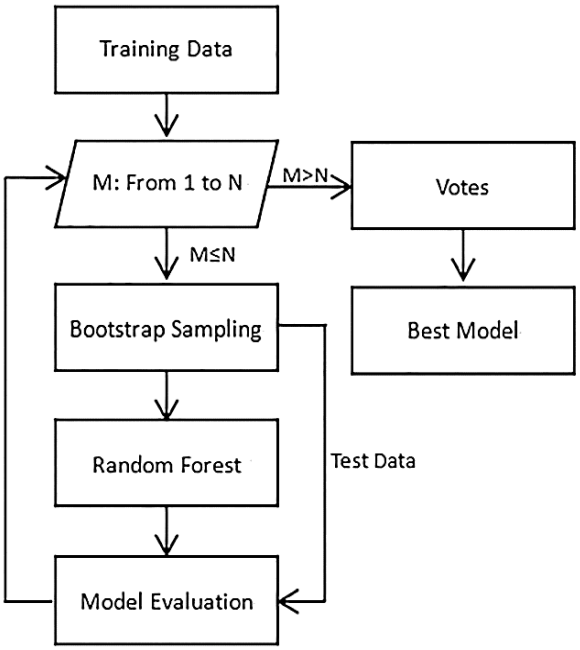
Toàn bộ quá trình của Random Forest đều được ngẫu nhiên theo 2 bước:

- Bước 1: Ngẫu nhiên theo mẫu (sample).
  Sử dụng phương pháp bootstrap có hoàn lại (with replacement).
- Bước 2: Lấy ngẫu nhiên các thuộc tính từ các tập ban đầu.
  Các tập dữ liệu con được tạo ra từ các bước ngẫu nhiên trên có tính đa dạng cao, khả năng trùng lặp thấp, và đặc biệt là giảm phương sai (variance). Từ các tập dữ liệu con này sẽ tạo thành một rừng cây ngẫu nhiên. Khi tổng hợp kết quả, Random Forest sử

dụng bỏ phiếu (voting method) đối với bài toán phân loại, và lấy giá trị trung bình (average) đối với bài toán hồi quy. Kết hợp các thuật toán này với nhau để ra một kết quả cuối cùng, nên Random Forest được gọi là Ensemble Learning.

2. Thuật toán

Tóm tắt thuật toán Random Forest cho phân loại dữ liệu:



Hình 4: Cơ chế hoạt động của thuật toán Random Forest

Bước 1: Ngẫu nhiên hóa và không cắt tỉa (Randomization and no pruning):

1.1. Với mỗi cây và tại mỗi nút, chúng ta chọn ngẫu nhiên một tập nhỏ các thuộc tính.

1.2. Phát triển các nhánh con từ các nút mà ta đã chọn.

1.3. Tất cả các cây sẽ phát triển tối đa tới kích thước lớn nhất của chúng mà không bị cắt tỉa (phát triển độc lập).

Bước 2: Kết hợp (Combination): Mỗi lần phán đoán, đưa quan sát cho tất cả các cây, yêu cầu mỗi cây phán đoán, lấy trung bình các phán đoán rồi đem ra phán đoán.

Bước 3: Bagging: Tập huấn luyện cho mỗi tập cây được tạo ra bằng cách lấy mẫu (có hoàn lại) từ dữ liệu gốc.

CHƯƠNG IV: ENSEMBLE LEARNING WITH RANDOM FOREST

1. Bootstrapping

Một trong những lý do chính khiến thuật toán Random Forest rất hiệu quả chính là do sự ngẫu nhiên của mỗi cây. Mỗi cây quyết định riêng lẻ sẽ được xây dựng dựa trên tập huấn luyện được tạo ra bằng cách lấy mẫu (có hoàn lại) từ dữ liệu gốc.

Chính xác hơn, mỗi mẫu bootstrap sẽ chứa khoảng $1 - \frac{1}{e} \approx 0.632$ mẫu dữ liệu gốc. Hãy tìm hiểu kỹ thuật Bootstraping, chúng ta có dữ liệu gốc $x_1, x_2, \dots x_n$ với n dữ liệu trong đó. Chúng ta lấy mẫu (có hoàn lại) từ dữ liệu gốc cho đến khi chúng ta có một tập dữ liệu khác có kích thước n. Từ đó, xác suất lựa chọn bất cứ dữ liệu nào ở lần lấy đầu tiên là $\frac{1}{n}$, vì thế xác suất không lựa chọn dữ liệu đó là $1 - \frac{1}{n}$. Đó chỉ là lần lấy đầu tiên, có tổng cộng n lần lấy, tất cả lần lấy đều độc lập, vì vậy xác suất không bao giờ chọn một dữ liệu bất kì là $\left(1 - \frac{1}{n}\right)^n$.Cho rằng n ngày càng tăng cho đến vô cùng, chúng ta có: $\lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = \frac{1}{e} \approx 0.368 \approx \frac{1}{3}$.

Như vậy có xấp xỉ 1/3 dữ liệu trong dữ liệu gốc không không nằm trong mẫu (out-of-bag) vì thế chúng không tham gia vào quá trình huấn luyện. Điều này có nghĩa là chỉ có 2/3 các phần tử trong tập tập huấn luyện (in-of-bag) được tham gia vào quá trình tính toán để phân loại. 1/3 các phần tử không thuộc trên dùng để kiểm tra sai số được gọi là dữ liệu OOB. Dữ liệu OOB có tác dụng ước lượng sai số từ việc kết hợp các mô hình riêng lẻ sau đó được tổng hợp trong mô hình Random Forest.

2. Độ lỗi out-of-bag (Độ lỗi OOB)

Bởi vì tập out-of-bag (OOB) không được sử dụng để huấn luyện cây trong Random Forest, đó chính là một tập test hoàn hảo dành cho các cây. Độ lỗi OOB là độ lỗi trung bình cho mỗi lần tính toán sử dụng dự đoán từ tập OOB của các cây. Để tính độ lỗi OOB, ta làm như sau:

Bước 1: *Sử dụng các dữ liệu trong tập OOB của từng cây như một tập test để lấy kết quả của cây đưa ra.*

Bước 2: *Tổng hợp kết quả sử dụng bỏ phiếu đối với bài toán phân loại và lấy giá trị trung bình đối với bài toán hồi quy.*

Bước 3: *Tổng hợp sai số của kết quả so với giá trị thực.*

3. Lựa chọn ngẫu nhiên tập thuộc tính tại mỗi nút của cây

Một đặc điểm quan trọng mà Random Forest có chính là tính ngẫu nhiên khi lựa chọn tập thuộc tính tại mỗi nút của cây. Đối với thuật toán ID3, để xây dựng cây quyết định, ta lựa chọn thuộc tính mà có Information Gain

của thuộc tính đó thấp nhất để quyết định thuộc tính nào tại nút đó, điều này làm cho thuật toán ID3 có xu hướng tạo ra cây lùn. Ví dụ nếu xét lớp học có 10 học sinh với các đặc trưng khác nhau, giả sử 10 học sinh này có 10 ngày sinh khác nhau và các đặc trưng khác có thể giống nhau giữa các học sinh thì thuật toán ID3 sẽ lấy đặc trưng ngay tại nút gốc là ngày sinh vì thuộc tính ngày sinh có Information Gain cao nhất, và từ đó ta có 10 nhánh tương ứng với 10 ngày sinh khác nhau và không quan tâm những đặc trưng khác. Điều này làm cho cây fit hoàn toàn dữ liệu huấn luyện nhưng sẽ vô nghĩa nếu đem ra dự đoán trên dữ liệu tương lai. Đây cũng là nhược điểm chính của cây quyết định – rất dễ xảy ra tình trạng Overfitting. Đối với thuật toán Random Forest, ta lựa chọn tập thuộc tính không theo Information Gain mà hoàn toàn ngẫu nhiên. Bằng cách này, ta có thể tạo ra nhiều cây khác nhau với xác suất trùng lặp thấp. Tập thuộc tính con chứa m thuộc tính tại mỗi nút thông thường được tính như sau:

- Đối với mô hình phân loại:

$$m = \sqrt{p}$$

- Đối với mô hình hồi quy:

$$m = \frac{p}{3}$$

Với p là số lượng thuộc tính trong bộ dữ liệu.

4. Bagging

Khi tổng hợp kết quả, Random Forest sử dụng bỏ phiếu (voting method) đối với bài toán phân loại, và lấy giá trị trung bình (average) đối với bài toán hồi quy. Kết hợp các thuật toán này với nhau để ra một kết quả cuối cùng, nên Random Forest được gọi là Ensemble Learning.

5. Thuật toán rừng ngẫu nhiên

Gọi B là số lượng cây mà chúng ta muốn xây dựng trong khu rừng. Ta có thuật toán sau:

- Xây dựng các cây

for $b = 1$ to B:

Bước 1: *Lấy ngẫu nhiên tập dữ liệu huấn luyện cho cây từ bộ dữ liệu với kỹ thuật Bootstrapping.*

Bước 2: *Xây dựng cây quyết định từ tập bootstrap và tại mỗi nút, chúng ta chọn ngẫu nhiên một tập nhỏ các thuộc tính.*

Bước 3: *Phát triển các nhánh con từ các nút mà ta đã chọn.*

Bước 4: *Tất cả các cây sẽ phát triển tối đa tới kích thước lớn nhất của chúng mà không bị cắt tỉa (phát triển độc lập).*

- Tổng hợp kết quả của các cây trong rừng để đưa ra kết quả dự đoán

Bỏ phiếu (voting method) đối với bài toán phân loại, và lấy giá trị trung bình (average) đối với bài toán hồi quy để đưa ra kết quả cuối cùng.

6. Đánh giá tầm quan trọng của thuộc tính

Để đánh giá tầm quan trọng của thuộc tính trong mô hình, ta làm theo các bước sau:

Bước 1: *Tính độ lỗi OOB với bộ dữ liệu ban đầu trong mô hình (e_i).*

Bước 2: *Tính độ lỗi OOB với bộ dữ liệu mà các giá trị ứng với thuộc tính đó bị hoán đổi ngẫu nhiên (p_i).*

Bước 3: *Tính độ quan trọng của thuộc tính dựa trên trung bình và độ lệch chuẩn của ($p_i - e_i$) qua tất cả các cây trong quần thể.*

Nếu độ quan trọng của thuộc tính trong Random Forest càng cao thì độ lỗi OOB trước và sau khi hoán đổi các giá trị của thuộc tính đó càng chênh lệch nhiều. Ta có mức độ chênh lệch của độ lỗi OOB trước và sau khi hoán đổi các giá trị của thuộc tính đó như sau:

$$d_i^m = p_i^m - e_i^m$$

Trung bình và phương sai của độ lỗi OOB cho tất cả cây:

$$\overline{d}_i = \frac{1}{m}\sum_{i=1}^{m} d_i^m, \quad s_i^2 = \frac{1}{m-1}\sum_{i=1}^{m}(d_i^m - \overline{d}_i)^2$$

Độ quan trọng của thuộc tính được tính như sau: $v_i = \frac{\overline{d_i}}{s_i}$