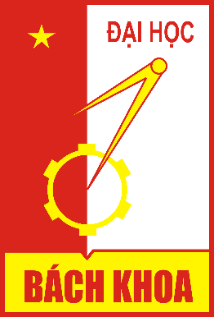
**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC**

🙥🙤✯✯✯🙦🙧



**ĐỒ ÁN I**

**Thuật toán cây quyết định và kỹ thuật Bagging**

**trong bài toán phân lớp**

**Chuyên ngành: Toán Tin**

|  |  |
| --- | --- |
| Giảng viên hướng dẫn: | **TS. Nguyễn Tuấn Dũng** |
| Sinh viên thực hiện: | **Nguyễn Đình Nam** |
| MSSV:  Lớp: | **20216859**  **Toán Tin 01 – K66** |
|  |  |

*Hà Nội, tháng 4 năm 2024*

**LỜI MỞ ĐẦU**

Mục lục

[**CHƯƠNG I. Giới thiệu bài toán phân lớp** 1](#_Toc164151077)

[I.1. Tổng quan về bài toán phân lớp 1](#_Toc164151078)

[I.2. Phân loại các bài toán phân lớp 1](#_Toc164151079)

[I.3. Cách đánh giá một bài toán phân lớp 1](#_Toc164151080)

[I.4. Độ lệch và phương sai 1](#_Toc164151081)

[**CHƯƠNG II. Thuật toán cây quyết định** 1](#_Toc164151082)

[**Chương III. Kỹ thuật Bagging** 1](#_Toc164151083)

[III.1. Giới thiệu chung về Ensemble Learning 1](#_Toc164151084)

[III.2. Kỹ thuật Bagging 1](#_Toc164151085)

[III.3. Thuật toán Random Forest 1](#_Toc164151086)

[Chương IV. Chương trình minh họa 1](#_Toc164151087)

# **CHƯƠNG I. Giới thiệu bài toán phân lớp**

## I.1. Tổng quan về bài toán phân lớp

Bài toán phân loại là bài toán thuộc lớp bài toán học có giám sát trong Machine Learning. Bằng việc sử dụng các thuật toán trong Machine Learning để gán nhãn một cách chính xác nhất cho các mẫu từ bộ dữ liệu đầu vào. Một ví dụ đơn giản là dự đoán một người có bị bệnh tiểu đường không thông qua các chỉ số HbA1c, đường huyết,….

## I.2. Phân loại các bài toán phân lớp

Các bài toán phân lớp được chia thành các loại sau:

* Bài toán phân lớp nhị phân (Binary Classification): Mục đích của bài toán là phân loại dữ liệu thành 2 nhãn lớp. Một số ví dụ ta thường gặp như : phân loại email là thư rác hoặc không?, khách hàng sử dụng dịch vụ sẽ rời đi hay tiếp tục sử dụng?,… Thông thường bài toán này dựa trên mô hình dự đoán phân phối xác suất Bernoulli. Một số thuật toán phổ biến được sử dụng gồm: Hồi quy Logistic, Cây quyết định, SVM, ….
* Phân lớp nhiều lớp (Multi-Class Classification): Nhiệm vụ của bài toán này là phân loại có nhiều hơn 2 nhãn lớp. Từ bộ dữ liệu đầu vào, các mẫu sẽ được phân loại vào một trong các lớp đã biết. Ví dụ: Phân loại khuôn mặt, thực vật,… Người ta thường lập mô hình nhiệm vụ phân lớp nhiều lớp với một mô hình dự đoán phân phối xác suất Multinoulli cho mỗi mẫu. Một số thuật toán phổ biến: k-Nearest Neighbors, Cây quyết định, Rừng ngẫu nhiên, Gradient Boosting,… Các thuật toán phân lớp nhị phân có thể dùng cho các bài toán phân loại nhiều lớp.
* Phân lớp nhiều nhãn (Multi-label Classification) : đề cập đến các nhiệm vụ phân lớp có hai hoặc nhiều nhãn lớp, trong đó một hoặc nhiều nhãn lớp có thể được dự đoán cho mỗi mẫu. Bài toán phân loại ảnh là bài toán khá phổ biến trong loại bài toán này.Nhiệm vụ phân loại nhiều nhãn này được dự đoán với một mô hình có nhiều đầu ra và 1 đầu ra tương ứng với một phân phối xác suất Bernoulli. Vậy thì bản chất của nó chính là tạo ta nhiều dự đoán phân loại nhị phân cho mỗi mẫu. Một số thuật toán phổ biến có thể kể đến là: Multi-label Decision Trees, Multi-label Random Forest,…
* Phân lớp không cân bằng (Imbalanced Classification): Đề cập đến các nhiệm vụ phân lớp trong đó số lượng ví dụ trong mỗi lớp được phân bố không đồng đều. Một số ví dụ: Xét nghiệm y tế, phát hiện gian lận trong thi cử,…

## I.3. Cách đánh giá một bài toán phân lớp

I.3.1. Ma trận nhầm lẫn  
 Ma trận nhầm lẫn (Confusion matrix) của bài toán nhị phân được xác định như sau:

|  |  |  |
| --- | --- | --- |
| Prediction  Reality | 1 | 0 |
| 1 | TP | FN |
| 0 | FP | TN |

Trong đó:

TP (True positive): số lượng mẫu được dự đoán là positive và thực tế là positive.

FP (False positive): số lượng mẫu được dự đoán là positive và thực tế là negative.

TN (True negative): số lượng mẫu được dự đoán là negative và thực tế là negative.

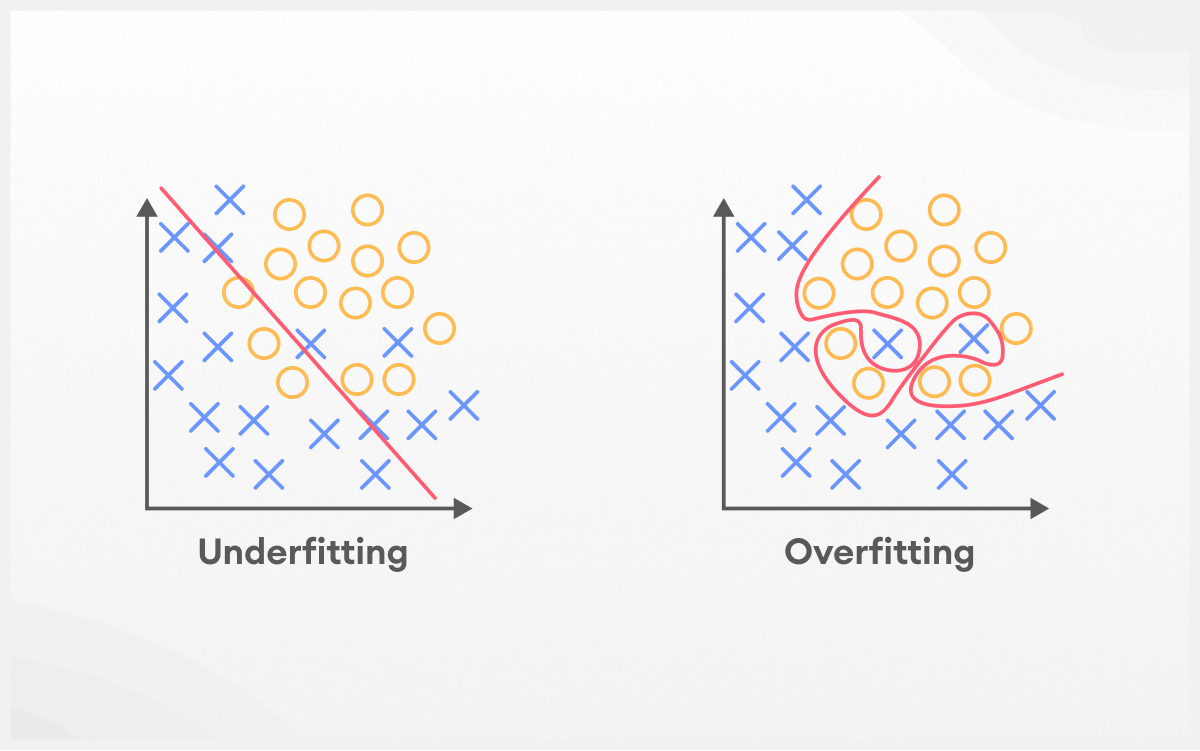
FN( False negative): số lượng mẫu được dự đoán là negative và thực tế là positove.

* Độ chính xác (Accuracy): là đại lượng thể hiện tỷ lệ giữa số lượng mẫu được phân loại đúng so với tổng số mẫu. Tuy nhiên nó không thể phản ánh được đầy đủ hiệu suất của mô hình trong 1 số trường hợp.
* Độ nhạy (Recall/Sensitivity): là đại lượng đo lường tỷ lệ mẫu dương thực sự đã được phân loại đúng so với tổng số mẫu dương.
* Độ chính xác (Precision): là đại lượng đo tỷ lệ mẫu dương được phân loại đúng so với tổng số mẫu được phân loại là dương.
* Điểm F1 (F1 score): đo lường tỷ lệ mẫu dương thực sự đã được phân loại đúng so với tổng số mẫu dương.
* Chỉ số Specificity: là tỷ lệ giữa số lượng mẫu thực sự thuộc lớp âm và được phân loại đúng so với tổng số mẫu thực sự thuộc lớp âm.
* ROC – AUC:
  + ROC (Receiver Operating Char-acteristic) được sử dụng để đánh giá hiệu quả của mô hình dự đoán. ĐườngROC biểu diễn sự tương quan giữa tỷ lệ dương tính và tỷ lệ giả âm tính của mô hình trên các ngưỡng khác nhau.
  + AUC (Area under the curve): là diện tích ở bên dưới của đường ROC.

## I.4. Độ lệch và phương sai

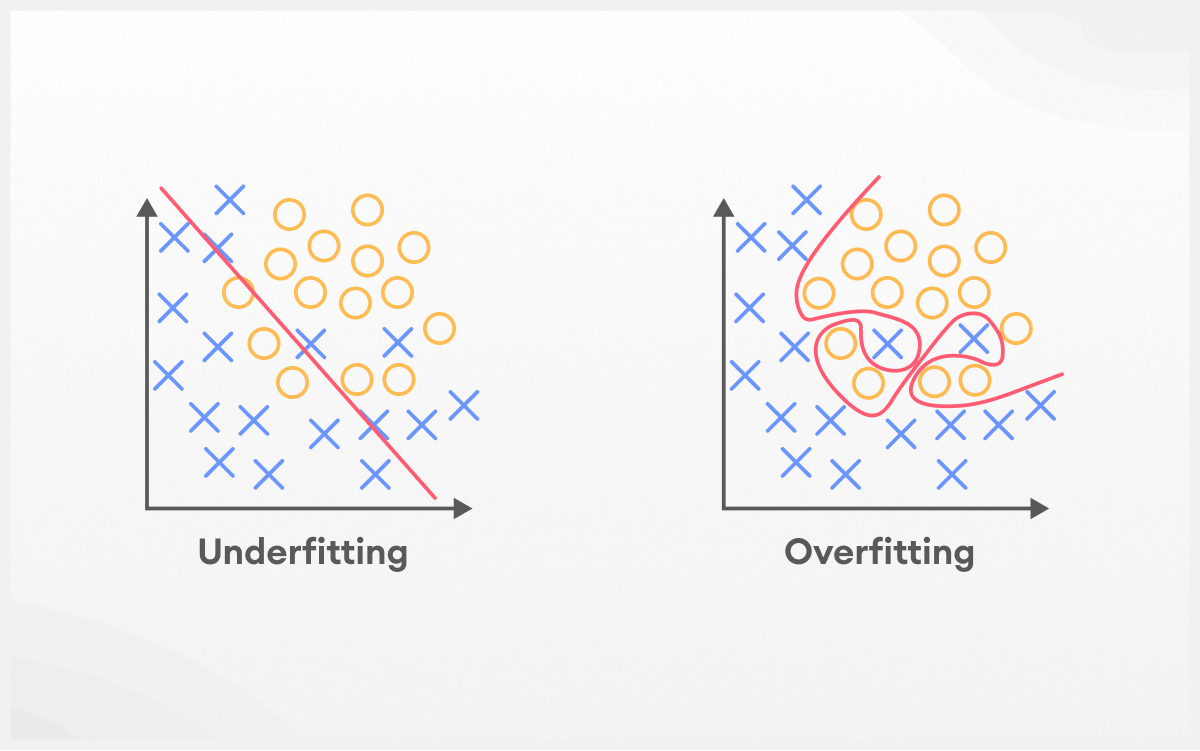
I.4.1. Độ lệch (Bias)

Độ lệch là chỉ số đo lường mức độ sai lệch của mô hình so với thực tế. Ta luôn mong muốn chỉ số này sẽ thấp khi xây dựng mô hình vì giá trị dự đoán sẽ gần với giá trị thực tế. Tuy nhiên trong những bộ dữ liệu lớn, việc mô hình đơn giản quá có thể dẫn tới vấn đề Underfitting. Underfitting là tình trạng khi mô hình học máy không đủ mạnh để hợp lý diễn giải dữ liệu huấn luyện, dẫn đến kết quả là mô hình có độ chính xác thấp trên dữ liệu huấn luyện và có thể không tốt trên dữ liệu kiểm tra. Nguyên nhân là do mô hình của ta quá đơn giản trong khi các dữ liệu có mối quan hệ phức tạp với nhau. Để giảm độ lệch ta có thể sử dụng một số phương án sau : tăng độ phức tạp của mô hình, tìm siêu tham số, đánh giá lại bộ dữ liệu, thu thập thêm dữ liệu và ensemble learning.



I.4.2. Phương sai (Variance)

Phương sai là chỉ số đo lường mức độ biến động của dự đoán của một mô hình khi được huấn luyện trên các tập dữ liệu khác nhau. Nó đặc trưng cho độ nhạu của mô hình tức là khi ta thay đổi dữ liệu huấn luyện. Một mô hình có phương sai cao có thể dẫn tới việc Overfitting, nơi mô hình “nhớ” dữ liệu huấn luyện 1 cách quá mức và không thể tổng quát hóa cho dữ liệu mới. Vì thế khi gặp những dữ liệu mới, nó sẽ đưa ra một dự đoán thiếu độ chính xác. Để khắc phục điều này ta có thể sử dụng một số cách sau: giảm độ phức tạp của mô hình, sử dụng phương pháp đánh giá chéo, ensemble learning.



# **CHƯƠNG II. Thuật toán cây quyết định**

Cây quyết định (Decision Tree) là một trong những phương pháp vô cùng phổ biến trong machine learning, được ứng dụng trong cả bài toán phân loại và hồi quy. Đây là một mô hình mạnh mẽ và sẽ được trình bày ngay dưới đây.

II.1. Định nghĩa

Cây quyết định sử dụng cấu trúc dạng cây để thể hiện một số đường dẫn quyết định có thể có và kết quả của mỗi đường dẫn. Với cách thể hiện này sẽ vô cùng dễ hiểu trong việc sử dụng để mô trả trực quan dữ liệu. Các nút ở dưới cùng gọi là nút lá, mỗi nút lá thể hiện cho các lớp (bài toán phân loại) hoặc giá trị dự đoán (bài toán hồi quy). Các nút trong cây đại diện cho một thuộc tính của bộ dữ liệu và mỗi cạnh từ nút đại diện cho một giá trị trị thuộc tính.

Ví dụ:

A screenshot of a computer

Description automatically generated

Minh họa cây quyết định

II.2. Thuật toán cây quyết định

II.2.1. Thuật toán ID3

Ý tưởng của thuật toán ID3: Ý tưởng của thuật toán là chúng ta sẽ xác định của thuộc tính được xem xét tại mỗi bước. Đối với các bài toán có nhiều thuộc tính và mỗi thuộc tính có nhiều giá trị khác nhau, việc tìm nghiệm tối ưu là khó khăn và ít khả thi. Thay vào đó, có cách khác là tại mỗi bước, thuộc tính tốt nhất được chọn dựa trên 1 tiêu chuẩn nào đó. Ta sẽ chia dữ liệu vào các child node tương ứng với các giá trị của thuộc tính đó rồi áp dụng phương pháp này cho mỗi child node. Cách chọn greedy (tham lam) sẽ giúp ra chọn thuộc tính tốt nhất tại mỗi bước. Tuy nhiên đây có thể không phải cách tối ưu nhưng giúp ta thấy gần với cách làm tối ưu. Sau mỗi câu hỏi, dữ liệu được phân chia vào từng child node tương ứng với các câu trả lời cho câu hỏi đó. Câu hỏi ở đây là một thuộc tính, câu trả lời là giá trị của thuộc tính đó. Nhưng câu hỏi đặt ra ở đây là phân chia như thế nào là tốt, là tối ưu? Thực tế thì có nhiều hệ số khác nhau mà phương pháp Cây quyết định sử dụng để phân chia nhưng phổ biến nhất là hệ số Entropy

Thực tế thì có nhiều hệ số khác nhau mà phương pháp Cây quyết định sử dụng để phân chia nhưng phổ biến nhất là hệ số Entropy

Entropy

Entropy là một thuật ngữ thuộc lĩnh vực Nhiệt động lực học, là thước đó của sự biến đổi, hỗn loạn hoặc ngẫu nhiên. Sau này, nó được mở rộng sang lĩnh vực nghiên cứu, thống kê.

Với n giá trị của biến rời rạc x nhận giá trị lần lượt là Giả sử rằng xác suất để x nhận các giá trị này là . Ký hiệu phân phối này là p = .

Entropy của phân phối này được định nghĩa như sau:

Giả sử tung một đồng xu, entropy được tính như sau:

* P tinh khiết: . Tinh khiết tức là việc lựa chọn biến sao cho sau khi phân chia thì kết quả trả về tại nút con chỉ thuộc về một lớp.
* P vẩn đục: . Vẩn đục là phân phối của các nhãn tại nút con còn khá mập mờ, không có xu hướng thiên về một nhãn nào cụ thể.

Entropy tại một nút lá được tính bằng tổng có trọng số của entropy của từng lớp dữ liệu tại nút đó. Nếu tập dữ liệu của một nút lá chứa 𝑁 quan sát và có 𝐶 lớp khác nhau, với số lượng quan sát của lớp thứ 𝑖 là ​, entropy tại nút lá đó được tính theo công thức:

Tính hàm số [Entropy](https://trituenhantao.io/tu-dien-thuat-ngu/entropy/) tại mỗi thuộc tính: với thuộc tính x, các điểm dữ liệu trong S được chia ra nút con với số điểm trong mỗi child node lần lượt là. Ta có:

Thuộc tính x được lựa chọn sẽ thỏa mãn tổng trọng số Entropy đạt giá trị nhỏ nhất.

II.2.2. Ví dụ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| outlook | **temperature** | **humidity** | **wind** | **play** |
| sunny | hot | high | weak | no |
| sunny | hot | high | strong | no |
| overcast | hot | high | weak | yes |
| rainy | mild | high | weak | yes |
| rainy | cool | normal | weak | yes |
| rainy | cool | normal | strong | no |
| overcast | cool | normal | strong | yes |
| sunny | mild | high | weak | no |
| sunny | cool | normal | weak | yes |
| rainy | mild | normal | weak | yes |
| sunny | mild | normal | strong | yes |
| overcast | mild | high | strong | yes |
| overcast | hot | normal | weak | yes |
| rainy | mild | high | strong | no |

Ví dụ được trích từ cuốn sách Data Mining: Practical Machine Learning Tools and Techniques, trang 11. Đây có thể coi là một ví dụ khá phổ biến trong các bài giảng về cây quyết định.

Có bốn thuộc tính thời tiết:

Outlook: có 3 giá trị sunny, overcast, rainy.

Tempereture: có 3 giá trị hot, cool, mild.

Humidity: có 2 giá trị high, normal.

Wind: có 2 giá trị weak, strong.

Entropy tại root node của bài toán:

Kế tiếp, ta sẽ tính tổng các trọng số của các child node nếu chọn một trong bốn thuộc tính: outlook, temperature, humidity, wind để phân chia dữ liệu.

Giả sử ta chọn outlook. Gọi tập hợp các điểm trong mỗi child node này ứng với 3 giá trị sunny, overcast, rainy lần lượt là với phần tử.

Ta có:

÷

II.2.3. Điều kiện dừng

* Nếu tại một nút có Entropy bằng 0, nghĩa là mọi điểm trong nút đều thuộc một lớp.
* Nếu tại một nút có số phần tử nhỏ hơn một ngưỡng nào đó. Lúc này, ta chấp nhận có một số điểm phân lớp sai để tránh vấn đề overfitting.
* Nếu khoảng cách từ nút đó đến nút gốc đạt một giá trị nào đó. Việc hạn chế chiều sâu của cây làm giảm độ phức tạp của cây và đồng thời tránh được overfitting.
* Nếu việc phân chia nút đó không làm giảm Entropy quá nhiều.

II.2.3. Ưu nhược điểm của Cây quyết định

Ưu điểm:

* Mô hình dễ hiểu, dễ sử dụng.
* Dữ liệu đầu vào có thể là là dữ liệu missing, không cần chuẩn hóa hoặc tạo biến giả.
* Có thể làm việc với cả dữ liệu số và dữ liệu phân loại.
* Làm việc với cả bài toán phân loại và hồi quy.
* Cho biết ảnh hưởng của những thuộc tính đến kết quả

Nhược điểm:

* Thường xuyên gặp vấn đề overfitting.
* Ít ổn định
* Phụ thuộc lớn vào dữ liệu đầu vào. Chỉ với một thay đổi nhỏ của dữ liệu, cấu trúc mô hình của cây sẽ thay đổi hoàn toàn
* Xử lý thiên vị. Thông thường cây quyết định ưu thích những kiểu dữ liệu danh mục.

# **CHƯƠNG III. Kỹ thuật Bagging**

## III.1. Giới thiệu chung về Ensemble Learning

## III.2. Kỹ thuật Bagging

## III.3. Thuật toán Random Forest

# **CHƯƠNG IV. Chương trình minh họa**