TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

**KHOA CÔNG NGHỆ THÔNG TIN**

---


# ĐỒ ÁN TỐT NGHIỆP

ĐỀ TÀI

## XÂY DỰNG MÔ HÌNH DỰ ĐOÁN HÀNH VI NGHỈ HỌC CỦA SINH VIÊN


| | | |
|---|---|---|
| Giảng viên hướng dẫn | : | TS. Vũ Huấn |
| Sinh viên thực hiện | : | Nguyễn Đức Nguyên |
| Lớp | : | Công nghệ thông tin 3 K61 |
| Mã sinh viên | : | 201210257 |


Hà Nội, năm 2024

# LỜI CẢM ƠN

Em xin chân thành cảm ơn Thầy TS. Vũ Huấn trong thời gian qua đã giúp đỡ và hỗ trợ tận tình giúp em hoàn thành đề tài này một cách tốt nhất.

Qua đây, em cũng xin được gửi lời cảm ơn đến các thầy cô công tác tại khoa Công nghệ thông tin – Trường Đại học Giao thông vận tải đã dìu dắt và chỉ bảo em trong suốt quá trình học tập tại trường.

Em cũng xin gửi lời cảm ơn tới anh Hà Ngọc Linh – Project Manager phòng Công nghệ thông tin thuộc Trường Cao đẳng FPT Polytechnic Hà Nội đã giúp đỡ và tạo điều kiện cho em tiếp cận và phát triển đồ án trong thời gian thực tập tại đơn vị.

Cuối cùng, em xin gửi lời cảm ơn đến gia đình và tất cả người thân, bạn bè, những người đã luôn động viên, ủng hộ và giúp đỡ em trong thời gian hoàn thành đồ án.

Mặc dù em đã cố gắng rất nhiều nhưng cũng không tránh khỏi thiếu sót và hạn chế trong quá trình thực hiện. Em rất mong nhận được những đóng góp chân thành của các thầy cô và mọi người để có thể hoàn thiện đồ án được tốt hơn.

Em xin chân thành cảm ơn!

Hà Nội, ngày … tháng … năm 2024

Sinh viên thực hiện

Nguyễn Đức Nguyên

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

Hà Nội, ngày …  tháng … năm 2024

Giảng viên hướng dẫn

Vũ Huấn

# MỤC LỤC

# DANH MỤC BẢNG BIỂU

# DANH MỤC HÌNH ẢNH

# DANH MỤC CÁC TỪ VIẾT TẮT

| STT | Từ viết tắt | Tên đầy đủ | Giải thích |
|---|---|---|---|
| 1 | ML | Machine Learning | Lĩnh vực học máy |
| 2 | HDI | | Trạng thái sinh viên tiếp tục lên kỳ |
| 3 | THO | | Trạng thái sinh viên nghỉ học |

# LỜI MỞ ĐẦU

Tại Trường Cao đẳng FPT Polytechnic, ban quản lý nhà trường đang tìm kiếm giải pháp để giảm thiểu số lượng sinh viên bỏ học. Bằng việc dự đoán sinh viên nào có khả năng sẽ bỏ học, ban quản lý nhà trường sẽ có thể đưa ra thay đổi giáo dục và các chương trình khuyến học phù hợp để khuyến khích sinh viên không từ bỏ việc học. Tuy nhiên, những giải pháp hiện tại của ban quản lý vẫn chưa thực sự hiệu quả. Các kết quả dự đoán của ban quản lý có kết quả chính xác khá thấp. Ban quản lý quyết định sử dụng Machine Learning (Học máy) để tăng tỉ lệ dự đoán chính xác.

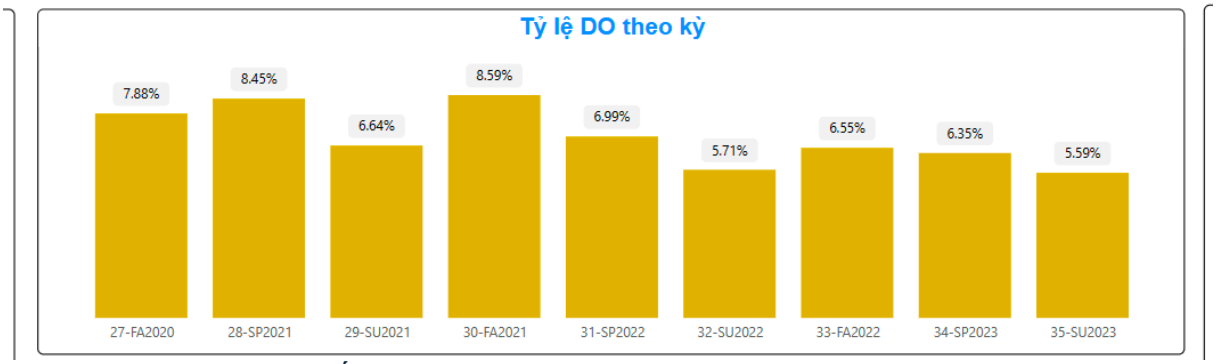
Đề tài sẽ trình bày về quá trình nghiên cứu dựa trên cơ sở dữ liệu của nhà trường cung cấp và thử nghiệm kết quả của Machine Learning để tìm ra dự đoán hiệu quả cung cấp danh sách sinh viên có dự định nghỉ học.

# CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

## 1.1 Tổng quan đề tài

### 1.1.1. Đặt vấn đề

Mục tiêu của mọi tổ chức giáo dục là xây dựng, hình thành thói quen học tập và truyền cảm hứng cho người học, giúp người học có thể phát triển bản thân theo hướng phù hợp nhất với mình. Để xác định được kết quả của quá trình này, từ trước tới nay, các bài kiểm tra, các bài đánh giá học thuật và các hình thức tương tự được coi là hình thức đánh giá giáo dục hiệu quả nhất. Bởi vậy, những kết quả đạt được của sinh viên là một trong những yếu tố quan trọng đối với mọi tổ chức giáo dục. Tuy nhiên hiện tại đã có nhiều yếu tố khác đóng vai trò tác động tới sự thành công hay thất bại của một tổ chức giáo dục, cụ thể là tỉ lệ sinh viên bỏ học. Sinh viên có thể bỏ học bởi nhiều lý do khác nhau: vấn đề về kinh tế, vấn đề về tâm lý sinh viên, các vấn đề liên quan tới môi trường giáo dục, kết quả học hay kinh nghiệm học tập của sinh viên. Hầu hết nguyên nhân dẫn tới quyết định nghỉ học của sinh viên tới từ việc sinh viên đạt kết quả không tốt trong quá trình học. Việc nghỉ học của sinh viên có ảnh hưởng lớn tới nguồn tài chính của nhà trường. Nhận ra được vấn đề này, trường Cao đẳng FPT Polytechnic quyết định sẽ tiến hành nghiên cứu để tìm ra giải pháp giảm thiểu tỉ lệ bỏ học của sinh viên. Bằng việc theo dõi tiến trình học tập của sinh viên (dựa theo kết quả học tập, kỳ học, số buổi xuất hiện, bài đánh giá, …), ban quản lý nhà trường tìm ra rằng những sinh viên sẽ bỏ học sẽ có chung một vài đặc điểm nhất định. Để tìm ra chính xác những đặc điểm này, ban quản lý đã sử dụng cách thống kê bằng Microsoft Excel để tổng hợp số liệu. Sau khi tìm ra một vài tiêu chí nhất định, nhà trường quyết định tiến hành dự đoán sinh viên. Với việc sử dụng Microsoft PowerBI để tạo ra các biểu đồ thống kê khi kết hợp với Excel, ban quản lý đưa ra được một danh sách dự đoán bỏ học theo từng kỳ. Sử dụng kết quả này, nhà trường đã đưa ra một danh sách những sinh viên có thể sẽ bỏ học vào kỳ tiếp theo.




**Hình 1.1.** Thống kê tỷ lệ sinh viên bỏ học theo kỳ của nhà trường

Áp dụng kết quả này vào một vài kỳ học, kết quả chỉ cho ra tỉ lệ dự đoán thành công vào khoảng 30% đến 60%. Tỉ lệ này trải dài qua một số lượng lớn các kỳ học và năm học. Điểm yếu của phương pháp này là sau mỗi kỳ, công thức để dự đoán phải được điều chỉnh tương ứng dẫn tới mức độ biến thiên của kết quả trở nên không thể kiểm soát.

Nhận ra vấn đề rằng kết quả của phương pháp hiện tại không hiệu quả, nhà trường quyết định tìm giải pháp mới để đưa ra kết quả chính xác và ổn định hơn. Machine Learning được lựa chọn để xây dựng mô hình mới này. Đề tài sẽ trình bày quá trình thu thập, xử lý và nghiên cứu dữ liệu của trường Cao đẳng FPT Polytechnic và thử nghiệm một vài mô hình Machine Learning cụ thể.

### 1.1.2. Mục tiêu đề tài

Trong đề tài này, chúng ta sẽ tập trung vào một số mục tiêu như sau:

- Tìm hiểu và phân tích cơ sở dữ liệu của FPT Polytechnic để chọn lọc ra những dữ liệu cụ thể phù hợp với việc xây dựng mô hình dự đoán sinh viên bỏ học.
- Tiến hành tiền xử lý và chuẩn hóa dữ liệu để phù hợp với model.
- Thử nghiệm một vài mô hình Machine Learning khác nhau và so sánh kết quả của chúng.
- Từ kết quả xử lý và dự đoán của mô hình, đưa ra ưu điểm và hạn chế của chương trình học hiện tại và những giải pháp để cải thiện cho kỳ học tiếp theo.
- Xây dựng trở thành một phần mở rộng cho hệ thống chăm sóc sinh viên của nhà trường nếu khả năng.

### 1.1.3. Đối tượng nghiên cứu

Đối tượng nghiên cứu được quan tâm là cơ sở dữ liệu và bộ dữ liệu được nhà trường FPT Polytechnic cung cấp. Từ cơ sở dữ liệu này, ta cần chọn lọc ra những bộ dữ liệu được ưu tiêu sử dụng cho quá trình dự đoán sinh viên sẽ nghỉ học và nghiên cứu chúng. Từ đó, kết hợp với những bảng dữ liệu phù hợp để cho ra kết quả là một bộ dữ liệu mới thỏa mãn yêu cầu dự đoán.

Dữ liệu sẽ tập trung vào các kết quả theo dõi sinh viên từ nhà trường trong các kỳ học 1, 2 và 3; lấy trong khoảng thời gian từ kỳ Summer 2018 cho tới Spring 2024 tại cơ sở FPT Polytechnic Hà Nội.

### 1.1.4. Phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài giới hạn trong đơn vị nhà trường FPT Polytechnic cơ sở Hà Nội.

## 1.2. Giới thiệu các công nghệ sử dụng

### 1.2.1. Python

#### 1.2.1.1. Giới thiệu về Python

Python là một ngôn ngữ lập trình được sử dụng rộng rãi trong các ứng dụng web, phát triển phần mềm, khoa học dữ liệu và máy học (ML). Các nhà phát triển sử dụng Python vì nó hiệu quả, dễ học và có thể chạy trên nhiều nền tảng khác nhau. Phần mềm Python được tải xuống miễn phí, tích hợp tốt với tất cả các loại hệ thống và tăng tốc độ phát triển.

#### 1.2.1.2. Những lợi ích của Python

- Các nhà phát triển có thể dễ dàng đọc và hiểu một chương trình Python vì ngôn ngữ này có cú pháp cơ bản giống tiếng Anh.
- Python giúp cải thiện năng suất làm việc của các nhà phát triển vì so với những ngôn ngữ khác, họ có thể sử dụng ít dòng mã hơn để viết một chương trình Python.
- Python có một thư viện tiêu chuẩn lớn, chứa nhiều dòng mã có thể tái sử dụng cho hầu hết mọi tác vụ. Nhờ đó, các nhà phát triển sẽ không cần phải viết mã từ đầu.
- Các nhà phát triển có thể dễ dàng sử dụng Python với các ngôn ngữ lập trình phổ biến khác như Java, C và C++.
- Cộng đồng Python tích cực hoạt động bao gồm hàng triệu nhà phát triển nhiệt tình hỗ trợ trên toàn thế giới. Nếu gặp phải vấn đề, bạn sẽ có thể nhận được sự hỗ trợ nhanh chóng từ cộng đồng.
- Trên Internet có rất nhiều tài nguyên hữu ích nếu bạn muốn học Python. Ví dụ: bạn có thể dễ dàng tìm thấy video, chỉ dẫn, tài liệu và hướng dẫn dành cho nhà phát triển.
- Python có thể được sử dụng trên nhiều hệ điều hành máy tính khác nhau, chẳng hạn như Windows, macOS, Linux và Unix.

#### 1.2.1.3. Sử dụng Python trong các lĩnh vực phát triển ứng dụng

- **Phát triển web phía máy chủ**

Phát triển web phía máy chủ bao gồm những hàm backend phức tạp mà các trang web thực hiện để hiển thị thông tin cho người dùng. Ví dụ: các trang web phải tương tác với cơ sở dữ liệu, giao tiếp với các trang web khác và bảo vệ dữ liệu khi truyền qua mạng.

Python hữu ích trong việc lập trình mã phía máy chủ bởi vì ngôn ngữ này cung cấp nhiều thư viện bao gồm mã viết sẵn cho các hàm backend phức tạp. Các nhà phát triển

cũng sử dụng một loạt các khung Python cung cấp tất cả những công cụ cần thiết để xây dựng ứng dụng web một cách nhanh chóng và dễ dàng hơn. Ví dụ: các nhà phát triển có thể tạo ứng dụng web khung trong nháy mắt bởi vì họ không cần phải lập trình nó từ đầu. Sau đó, họ có thể kiểm tra ứng dụng web này bằng cách sử dụng các công cụ kiểm thử của khung, mà không cần phụ thuộc vào những công cụ kiểm thử bên ngoài.

- **Tự động hóa bằng các tập lệnh Python**

Ngôn ngữ tập lệnh là một ngôn ngữ lập trình tự động hóa các tác vụ mà thường được con người thực hiện. Các lập trình viên thường xuyên sử dụng các tập lệnh Python để tự động hóa nhiều tác vụ hàng ngày như:

- Đổi tên một số lượng lớn tệp cùng lúc.
- Chuyển đổi một tệp sang một loại tệp khác.
- Loại bỏ các từ trùng lặp trong tệp văn bản.
- Thực hiện các phép tính toán cơ bản.
- Gửi email.
- Tải xuống nội dung.
- Thực hiện phân tích nhật ký cơ bản.
- Tìm kiếm lỗi trong nhiều tệp.

- **Khoa học dữ liệu và máy học**

Khoa học dữ liệu trích xuất thông tin quý giá từ dữ liệu và máy học (ML) dạy máy tính tự động học hỏi từ dữ liệu và đưa ra các dự đoán chính xác. Các nhà khoa học dữ liệu sử dụng Python cho các tác vụ khoa học dữ liệu sau:

- Sửa và loại bỏ dữ liệu không chính xác, còn được gọi là làm sạch dữ liệu.
- Trích xuất và chọn lọc các đặc điểm đa dạng của dữ liệu.
- Ghi nhãn dữ liệu gán tên có ý nghĩa cho dữ liệu.
- Tìm các số liệu thống kê khác nhau từ dữ liệu.
- Trực quan hóa dữ liệu bằng cách sử dụng các biểu đồ và đồ thị, chẳng hạn như biểu đồ đường, biểu đồ cột, biểu đồ tần suất và biểu đồ tròn.

Các nhà khoa học dữ liệu sử dụng những thư viện ML của Python để đào tạo các mô hình ML và xây dựng các công cụ phân loại giúp phân loại dữ liệu một cách chính xác. Các chuyên gia từ nhiều lĩnh vực sử dụng những công cụ phân loại dựa trên Python để thực hiện các tác vụ phân loại, chẳng hạn như phân loại hình ảnh, văn bản cũng như lưu lượng truy cập mạng, nhận dạng giọng nói và nhận diện khuôn mặt. Các nhà khoa học dữ liệu cũng sử dụng Python cho deep learning, một kỹ thuật ML nâng cao.

- **Phát triển phần mềm**

Các nhà phát triển phần mềm thường sử dụng Python cho những tác vụ phát triển và ứng dụng phần mềm khác nhau, chẳng hạn như:

- Theo dõi lỗi trong mã của phần mềm.
- Tự động xây dựng phần mềm.
- Đảm nhận quản lý dự án phần mềm.
- Phát triển nguyên mẫu phần mềm.
- Phát triển các ứng dụng máy tính bằng cách sử dụng những thư viện Giao diện đồ họa người dùng (GUI).
- Phát triển từ các trò chơi văn bản đơn giản cho đến những trò chơi điện tử phức tạp.

- **Tự động hóa kiểm thử phần mềm**

Kiểm thử phần mềm là quy trình kiểm tra xem kết quả thực tế từ phần mềm có khớp với kết quả mong đợi không để đảm bảo rằng phần mềm không có lỗi.

- Các nhà phát triển sử dụng khung kiểm thử đơn vị Python, chẳng hạn như Unittest, Robot và PyUnit, để kiểm thử các hàm do họ viết.
- Các kỹ sư kiểm thử phần mềm sử dụng Python để viết các trường hợp kiểm thử cho nhiều tình huống khác nhau. Ví dụ: họ sử dụng ngôn ngữ này để kiểm thử giao diện người dùng của một ứng dụng web, nhiều thành phần của phần mềm và những tính năng mới.

Các nhà phát triển có thể sử dụng một số công cụ để tự động chạy tập lệnh kiểm thử. Những công cụ này có tên gọi là công cụ Tích hợp liên tục/Triển khai liên tục (CI/CD). Các kỹ sư kiểm thử phần mềm cũng như những nhà phát triển sử dụng các công cụ CI/CD như Travis CI và Jenkins để tự động hóa quy trình kiểm thử. Công cụ CI/CD tự động chạy các tập lệnh kiểm thử Python và báo cáo kết quả kiểm thử bất kỳ khi nào nhà phát triển thêm vào những dòng mã mới.

#### 1.2.1.4. Đặc điểm của Python

Các đặc điểm sau tạo nên sự độc đáo của ngôn ngữ lập trình Python:

- **Python là một ngôn ngữ thông dịch:** Python là một ngôn ngữ thông dịch, điều này nghĩa là ngôn ngữ này trực tiếp chạy từng dòng mã. Nếu có lỗi trong mã chương trình, nó sẽ ngừng chạy. Do đó, lập trình viên có thể nhanh chóng tìm ra lỗi trong đoạn mã.

- **Python là một ngôn ngữ dễ sử dụng:** Python sử dụng từ ngữ giống trong tiếng Anh. Không giống như các ngôn ngữ lập trình khác, Python không sử dụng dấu ngoặc ôm. Thay vào đó, ngôn ngữ này sử dụng thụt đầu dòng.
- **Python là một ngôn ngữ linh hoạt:** Các lập trình viên không cần phải khai báo loại biến khi viết mã bởi vì Python sẽ xác định chúng vào thời điểm chạy. Vì vậy, bạn có thể viết các chương trình Python một cách nhanh chóng hơn.
- **Python là một ngôn ngữ cấp cao:** Python gần gũi với ngôn ngữ con người hơn các ngôn ngữ lập trình khác. Do đó, các lập trình viên không cần phải lo lắng về những chức năng cơ bản của nó như kiến trúc và quản lý bộ nhớ.
- **Python là một ngôn ngữ lập trình hướng đối tượng:** Python coi mọi thứ đều là đối tượng, nhưng ngôn ngữ này cũng hỗ trợ các phương thức lập trình khác như lập trình hàm và lập trình cấu trúc.

#### 1.2.1.5. Thư viện Python được sử dụng

- **Matplotlib:** Các nhà phát triển sử dụng Matplotlib để hiển thị dữ liệu dưới dạng đồ họa hai và ba chiều (2D và 3D) chất lượng cao. Thư viện này thường được sử dụng trong các ứng dụng khoa học. Với Matplotlib, bạn có thể trực quan hóa dữ liệu bằng cách hiển thị dữ liệu dưới dạng các biểu đồ khác nhau, chẳng hạn như biểu đồ cột và biểu đồ đường. Bạn cũng có thể hiển thị nhiều biểu đồ cùng lúc và các chi tiết đồ họa có thể được di chuyển qua mọi nền tảng.
- **Pandas:** Pandas cung cấp cấu trúc dữ liệu được tối ưu hóa và linh hoạt mà bạn có thể sử dụng để thao tác với dữ liệu chuỗi thời gian và dữ liệu có cấu trúc, chẳng hạn như bảng và nhóm. Ví dụ, bạn có thể sử dụng Pandas để đọc, ghi, hợp nhất, lọc và nhóm dữ liệu. Thư viện này được nhiều người sử dụng cho các tác vụ khoa học dữ liệu, phân tích dữ liệu và ML.
- **NumPy:** NumPy là một thư viện phổ biến mà các nhà phát triển sử dụng để dễ dàng tạo và quản lý nhóm, thao tác với các hình dạng logic và thực hiện các phép toán đại số tuyến tính. NumPy hỗ trợ tích hợp với nhiều ngôn ngữ như C và C ++.
- **Seaborn:** Seaborn là một thư viện trực quan hóa dữ liệu dựa trên Matplotlib, được thiết kế để tạo ra các biểu đồ trực quan đẹp mắt với một số dòng mã ngắn. Nó cung cấp các chức năng đơn giản để vẽ biểu đồ phân phối, biểu đồ dữ liệu đôi, và biểu đồ hồi quy, giúp nhanh chóng khám phá mối quan hệ trong dữ liệu và trực quan hóa kết quả của phân tích. Seaborn cũng hỗ trợ việc tùy chỉnh biểu đồ và làm cho chúng trông chuyên nghiệp hơn, thích hợp cho các dự án khoa học dữ liệu và trình bày kết quả.

- **Scikit-learn (sklearn):** Scikit-learn là một thư viện mã nguồn mở phổ biến cho machine learning trong Python. Nó cung cấp một loạt các thuật toán học máy cho các tác vụ như phân loại, hồi quy, gom cụm và giảm chiều dữ liệu. Scikit-learn còn đi kèm với các công cụ hữu ích cho tiền xử lý dữ liệu, tăng cường và đánh giá mô hình. Với một cú phát, bạn có thể áp dụng các thuật toán machine learning phổ biến vào dữ liệu của mình và đánh giá hiệu suất của chúng, làm cho quá trình phát triển mô hình trở nên nhanh chóng và dễ dàng hơn.

## 1.2.2. Machine Learning

### 1.2.2.1. Tổng quan Machine Learning

Các nhà khoa học Machine Learning (Máy học) cố gắng dạy cho máy tính nhận biết mẫu và tạo ra các kết nối bằng cách hiển thị cho máy tính một lượng lớn các ví dụ và dữ liệu liên quan. Machine Learning cũng cho phép các hệ thống máy tính theo dõi và cảm nhận các hoạt động trong môi trường của chúng để máy tính có thể điều chỉnh hành vi của mình để xử lý các thay đổi trong môi trường. Công nghệ cũng có thể được sử dụng để dự đoán hiệu suất, để cấu hình lại các chương trình dựa trên các điều kiện thay đổi và nhiều hơn nữa. Nói một cách kỹ thuật, Machine Learning là một lĩnh vực khoa học quan tâm đến việc thiết kế và phát triển các thuật toán cho phép máy tính học dựa trên dữ liệu từ các cảm biến, cơ sở dữ liệu và các nguồn khác. Việc học này sau đó được sử dụng để dự đoán, nhận dạng mẫu và hỗ trợ người ra quyết định.

Có 3 yếu tố chính trong Machine Learning:

- **Task:** Một Task được xác định là vấn đề chính mà ta quan tâm. Vấn đề này có thể liên quan tới việc dự đoán hoặc đưa ra những đề xuất, ước lượng về một đối tượng, vấn đề cụ thể.
- **Experience:** Được xác định là việc học từ dữ liệu lịch sử hoặc dữ liệu quá khứ và được sử dụng để ước tính và giải quyết các nhiệm vụ trong tương lai.
- **Performance:** Được xác định là khả năng của bất kỳ máy móc nào để giải quyết bất kỳ nhiệm vụ hoặc vấn đề học máy nào và cung cấp kết quả tốt nhất. Tuy nhiên, hiệu suất phụ thuộc vào loại vấn đề mà nó giải quyết.

### 1.2.2.2. Các kỹ thuật trong Machine Learning

- **Supervised Learning**

Học có giám sát được áp dụng khi một máy tính có dữ liệu mẫu, tức là dữ liệu đầu vào cũng như dữ liệu đầu ra với các nhãn đúng. Các nhãn đúng được sử dụng để kiểm tra tính chính xác của mô hình bằng cách sử dụng một số nhãn và thẻ. Kỹ thuật học có giám sát giúp chúng ta dự đoán các sự kiện trong tương lai với sự trợ giúp của kinh nghiệm quá khứ và các ví dụ được gán nhãn. Ban đầu, nó phân tích tập dữ liệu huấn

luyện đã biết, và sau đó giới thiệu một hàm suy luận tạo ra dự đoán về các giá trị đầu ra. Hơn nữa, nó cũng dự đoán các lỗi trong toàn bộ quá trình học này và cũng sửa chữa những lỗi đó thông qua các thuật toán.

- **Unsupervised Learning**

Trong học không giám sát, một máy tính được huấn luyện với một số mẫu đầu vào hoặc nhãn duy nhất, trong khi đầu ra không được biết trước. Thông tin huấn luyện không phân loại hoặc đánh nhãn; do đó, một máy tính có thể không luôn cung cấp đầu ra đúng so với học có giám sát.

Mặc dù học không giám sát ít phổ biến hơn trong cài đặt kinh doanh thực tế, nó giúp khám phá dữ liệu và rút ra những kết luận từ các bộ dữ liệu để mô tả các cấu trúc ẩn từ dữ liệu không được gắn nhãn.

- **Reinforcement Learning**

Học củng cố là một kỹ thuật học máy dựa trên phản hồi. Trong loại học này, các tác nhân (chương trình máy tính) cần khám phá môi trường, thực hiện các hành động và dựa trên các hành động của họ, họ nhận được phản hồi dưới dạng phần thưởng. Đối với mỗi hành động tốt, họ nhận được một phần thưởng tích cực, và đối với mỗi hành động xấu, họ nhận được một phần thưởng tiêu cực. Mục tiêu của một tác nhân học củng cố là tối đa hóa các phần thưởng tích cực. Vì không có dữ liệu được gắn nhãn, nên tác nhân buộc phải học từ kinh nghiệm của mình.

- **Semi-supervised Learning**

Học bán giám sát là một kỹ thuật trung gian của cả hai học có giám sát và không giám sát. Nó thực hiện các hành động trên các tập dữ liệu có một số nhãn cũng như dữ liệu không được gắn nhãn. Tuy nhiên, nó thường chứa dữ liệu không được gắn nhãn. Do đó, nó cũng giảm chi phí của mô hình học máy vì nhãn có giá, nhưng cho mục đích doanh nghiệp, nó có thể có một số nhãn. Hơn nữa, nó cũng tăng độ chính xác và hiệu suất của mô hình học máy.

Học bán giám sát giúp các nhà khoa học dữ liệu vượt qua nhược điểm của học có giám sát và không giám sát. Phân tích giọng nói, phân loại nội dung web, phân loại chuỗi protein, các bộ phân loại tài liệu văn bản, v.v., là một số ứng dụng quan trọng của học bán giám sát.

#### 1.2.2.3. Logistic Regression – Hồi quy Logistic

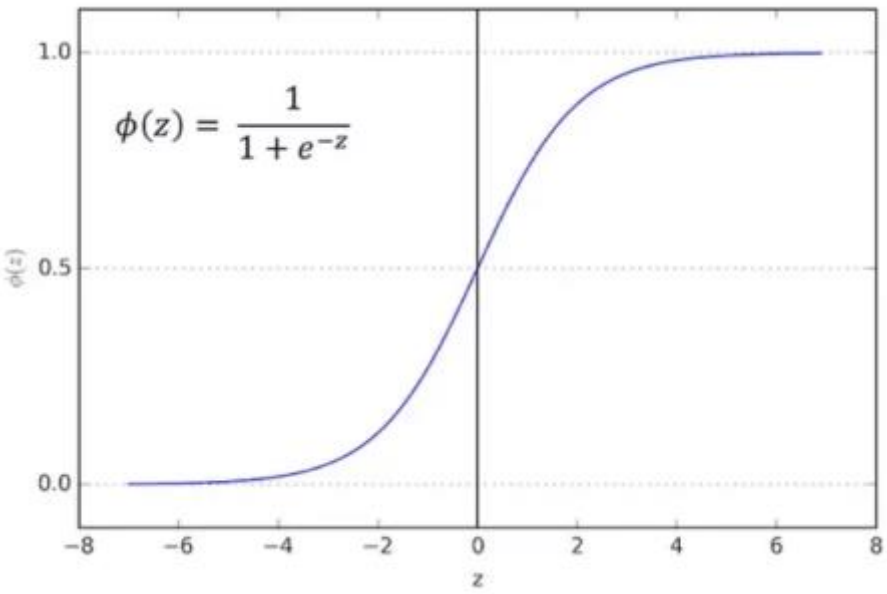
Hồi quy Logistic là một thuật toán phân loại rất phổ biến, đồng thời cũng là một phương pháp thống kê, dựa trên xác suất và sử dụng học có giám sát. Nó được phát triển vào những năm 1940 như một bổ sung cho các phương pháp hồi quy tuyến tính và phân

loại tuyến tính. Hồi quy Logistic đã được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau, bao gồm y học và khoa xã hội.

Hồi quy Logistic tương tự như hồi quy tuyến tính ở chỗ cùng nhằm mục tiêu hồi quy về một hàm toán học để giải thích mối quan hệ giữa biến phản ứng và các biến giải thích bằng cách sử dụng một mẫu các quan sát trước đó (dữ liệu huấn luyện). Tuy nhiên, hồi quy Logistic khác biệt với hồi quy tuyến tính ở một điểm quan trọng: đầu ra của nó (biến phản ứng) là một lớp thay vì một biến số. Điều này có nghĩa là, trong khi hồi quy tuyến tính được sử dụng để ước lượng một biến số liên tục, hồi quy Logistic được sử dụng để phân loại một biến số hạng mục. Mặc dù hình thức ban đầu của hồi quy Logistic được phát triển cho một biến đầu ra nhị phân (ví dụ: 1/0, có/không, đạt/không đạt, chấp/nhận), phiên bản hiện đại ngày nay có khả năng dự đoán các biến đầu ra đa lớp (tức là hồi quy Logistic đa mục tiêu). Nếu chỉ có một biến dự đoán và một biến được dự đoán, phương pháp được gọi là hồi quy Logistic đơn giản (tương tự như việc gọi các mô hình hồi quy tuyến tính với chỉ một biến độc lập là hồi quy tuyến tính đơn giản).

Hồi quy Logistic làm việc dựa trên nguyên tắc của hàm Sigmoid – một hàm phi tuyến tự chuyển đầu vào của nó thành xác suất thuộc về một trong hai lớp nhị phân. Hàm Sigmoid nhận đầu vào là một giá trị z bất kỳ, và trả về đầu ra là một giá trị xác suất nằm trong khoảng [0, 1], được biểu diễn bởi công thức:

$$\phi(z) = \frac{1}{1+ e^{-z}}$$




**Hình 1.2.** Đồ thị biểu diễn hàm Sigmoid

Khi áp dụng vào mô hình Hồi quy Logistic với đầu vào là ma trận dữ liệu X và trọng số w, ta có z = Xw. Việc huấn luyện của mô hình là tìm ra bộ trọng số w sao cho đầu ra dự đoán của hàm Sigmoid gần với kết quả thực tế nhất. Quy trình này bắt đầu với một giải pháp khởi đầu tạm thời, sau đó điều chỉnh các tham số với giá trị thay đổi nhỏ để xem liệu giải pháp có thể được cải thiện không, và lặp lại quá trình điều chỉnh lặp này cho đến khi không thể đạt được sự cải thiện hoặc là rất tối thiểu, tại đó quá trình được coi là đã hoàn tất/hội tụ.

#### 1.2.2.4. Random Forest – Rừng quyết định

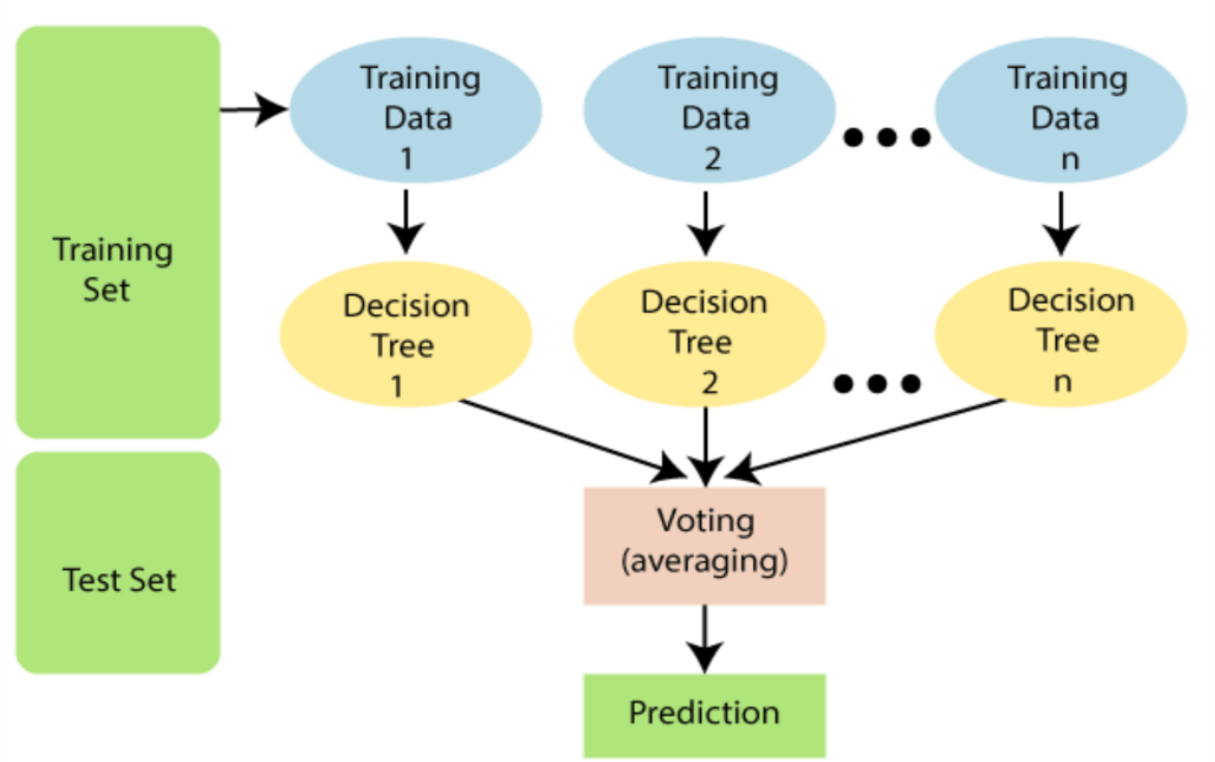
Random Forest (rừng ngẫu nhiên) là phương phân lớp thuộc tính được phát triển bởi Leo Breiman tại đại học California, Berkeley. Để hiểu kỹ thuật RF, ta cần tìm hiểu hai phương pháp là Boostrap và Bagging.

Bootstrap là một phương pháp nổi tiếng trong thống kê được giới thiệu bởi Bradley Efron vào năm 1979. Phương pháp này chủ yếu dùng để ước lượng lỗi chuẩn (standard errors), độ lệch (bias) và tính toán khoảng tin cậy (confidence interval) cho các tham số. Phương pháp này được thực hiện như sau: Từ một quần thể ban đầu lấy ra một mẫu L = (x1, x2,..xn) gồm n thành phần, tính toán các tham số mong muốn. Trong các bước tiếp theo lặp lại b lần việc tạo ra mẫu Lb cũng gồm n phần tử từ L bằng cách lấy lại mẫu với sự thay thế các thành phần trong mẫu ban đầu sau đó tính toán các tham số mong muốn.

Bagging được xem như là một phương pháp tổng hợp kết quả có được từ các bootstrap. Tư tưởng chính của phương pháp này như sau: Cho một tập huấn luyện D = {(xi, yi): i=1,2,…,n} và giả sử chúng ta muốn có một một dự đoán nào đó đối với biến x. Một mẫu gồm B tập dữ liệu, mỗi tập dữ liệu gồm n phần tử được chọn lựa ngẫu nhiên từ D với sự thay thế (giống như bootstrap). Do đó B = (D1, D2, ….,DB) trông giống như là một tập các tập huấn luyện được nhân bản. Tập huấn một máy hoặc một mô hình đối với mỗi tập Db (b=1, 2, …,B) và lần lượt thu thập các kết quả dự báo có được trên mỗi tập Db. Kết quả tổng hợp cuối cùng được tính toán bằng cách trung bình hóa (regression) hoặc thông qua số phiếu bầu nhiều nhất (classification).

Tóm tắt của giải thuật RF cho phân lớp được diễn giải như sau:

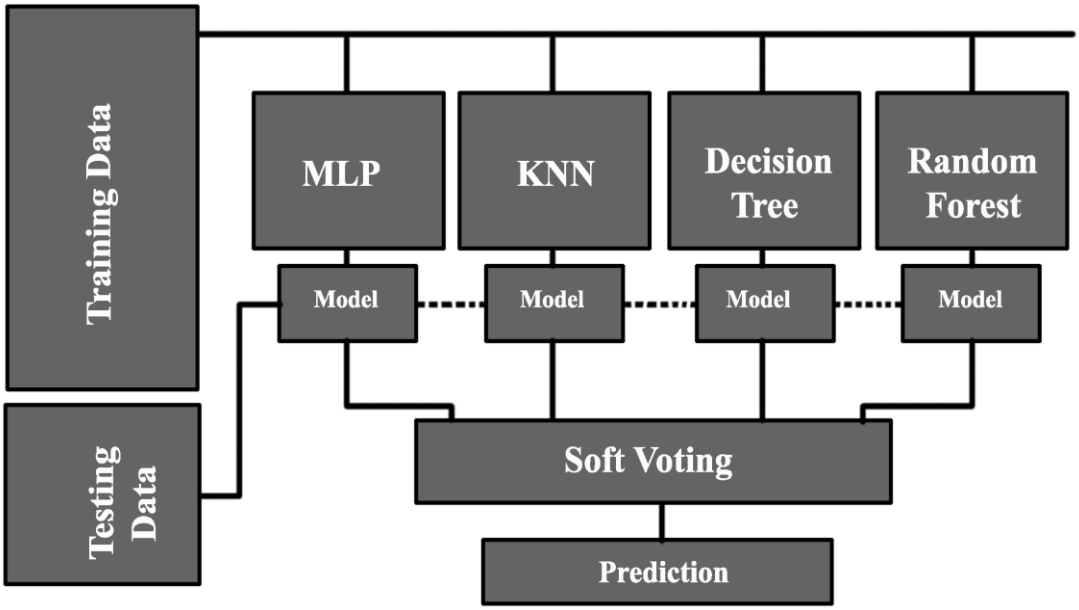
- Lấy ra K mẫu bootstrap từ tập huấn luyện.
- Đối với mỗi mẫu bootstrap xây dựng một cây phân lớp không được tỉa (unpruned tree) theo hướng dẫn sau: Tại mỗi nút thay vì chọn một phân chia tốt nhất trong tất cả các biến dự đoán, ta chọn ngẫu nhiên một mẫu m của các biến dự đoán sau đó chọn một phân chia tốt nhất trong các biến này.
- Đưa ra các dự đoán bằng cách tổng hợp các dự đoán của K cây.




**Hình 1.3.** Mô tả thuật toán Random Forest

#### 1.2.2.5. Voting Algorithm – Thuật toán bình chọn

Bình chọn hay bỏ phiếu là một phương pháp cho một nhóm để đưa ra quyết định tập thể hoặc bày tỏ ý kiến. Chúng ta có thể nói bỏ phiếu là một phương pháp kết hợp nhiều phương pháp phân loại đơn lẻ. Lý do để kết hợp các phương pháp phân loại là nhằm nâng cao tính hiệu quả và độ chính xác cho thuật toán đề xuất.




**Hình 1.4.** Mô tả thuật toán Voting

Trong đề tài này sẽ giới hạn ở Voting Classifier với mục đích cải thiện hiệu suất của mô hình dự đoán. Voting Classifier là một mô hình học máy học bằng cách huấn luyện trên một tập hợp nhiều mô hình và dự đoán đầu ra (lớp) dựa trên lớp có khả năng trở thành đầu ra cao nhất. Để dự đoán lớp đầu ra dựa trên số phiếu bầu lớn nhất, nó trung bình hóa kết quả của mỗi bộ phân loại được cung cấp vào Voting Classifier. Khái niệm là xây dựng một mô hình duy nhất học từ các mô hình khác nhau và dự đoán đầu ra dựa trên tổng số phiếu bầu cho mỗi lớp đầu ra, thay vì xây dựng các mô hình chuyên biệt riêng lẻ và xác định độ chính xác cho từng mô hình.

Có hai loại chính của Voting Classifier:

a. Hard Voting: Trong Hard Voting, lớp đầu ra được dự đoán là lớp có số phiếu bầu cao nhất, tức là lớp có khả năng được dự đoán cao nhất bởi mỗi bộ phân loại. Ví dụ, giả sử các bộ phân loại dự đoán các lớp đầu ra là (Mèo, Chó, Chó). Vì các bộ phân loại dự đoán lớp "Chó" nhiều nhất, chúng ta sẽ chọn Chó là dự đoán cuối cùng của mình.

b. Soft Voting: Trong Soft Voting, các xác suất trung bình của các lớp xác định lớp nào sẽ là dự đoán cuối cùng. Ví dụ, giả sử các xác suất của lớp là "Chó" là (0.30, 0.47, 0.53) và "Mèo" là (0.20, 0.32, 0.40). Vì vậy, trung bình của lớp Chó là 0.4333 và của lớp Mèo là 0.3067, từ đó, chúng ta có thể xác nhận dự đoán cuối cùng là Chó vì nó có xác suất trung bình cao nhất.

# CHƯƠNG 2: TỔNG QUAN VỀ DỮ LIỆU

## 2.1 Mô tả dữ liệu

Toàn bộ dữ liệu được xử lý và sử dụng trong đề tài sẽ được tập trung giới hạn vào các đặc điểm sau:

- Các kết quả nhập học của sinh viên được xác định từ năm 2018 trở đi do các dữ liệu trước 2018 chứa nhiều sai sót lớn và không đầy đủ để sử dụng.
- Sinh viên thuộc các ngành liên quan tới Công nghệ thông tin là chủ yếu (chiếm số lượng lớn nhất).
- Sử dụng dữ liệu của kỳ học 1, 2 và 3 của sinh viên (dữ liệu của kỳ 1, 2 sẽ được sử dụng để dự đoán trạng thái kỳ 3).

**Bộ dữ liệu gồm có:**

- Dữ liệu lịch sử trạng thái sinh viên.
- Dữ liệu điểm trung bình của sinh viên của cả kỳ 1 và 2.
- Dữ liệu tỉ lệ điểm danh của sinh viên của cả kỳ 1 và 2.
- Dữ liệu điểm trung bình theo từng nhóm môn (sử dụng tiền tố) và theo từng môn.
- Dữ liệu về môn học theo các kỳ của khung chương trình.

**Chi tiết từng bộ dữ liệu:**

- Lịch sử trạng thái sinh viên

**Bảng 2.1.** Bảng Lịch sử trạng thái sinh viên

| Tên trường | Mô tả |
|---|---|
| Student_code | Mã sinh viên |
| Status | Trạng thái học của sinh viên |
| Semester | Kỳ thứ của sinh viên (kỳ học) |
| Major_ID | Mã ngành học của sinh viên |
| Campus_Code | Mã cơ sở đào tạo |
| Term_ID | Kỳ triển khai |

- Dữ liệu điểm trung bình của sinh viên

**Bảng 2.2.** Bảng dữ liệu điểm trung bình của sinh viên

| Tên trường | Mô tả |
|---|---|
| User_Login | Tài khoản sinh viên |
| User_Code | Mã sinh viên |
| Term_ID | Kỳ triển khai |
| Major_ID | Mã ngành học của sinh viên |
| Total_Grade | Tổng điểm |
| Total_Creadit | Tổng tín chỉ |
| Average_Grade | Điểm trung bình |

- Dữ liệu tỉ lệ điểm danh của sinh viên

**Bảng 2.3.** Bảng dữ liệu tỉ lệ điểm danh của sinh viên

| Tên trường | Mô tả |
|---|---|
| User_Login | Tài khoản sinh viên |
| User_Code | Mã sinh viên |
| Term_ID | Kỳ triển khai |
| Total_Attendance | Tổng số lần điểm danh |
| Total_Activity | Tổng số buổi theo yêu cầu môn học |
| Percentage | Tỉ lệ điểm danh |

- Dữ liệu điểm trung bình theo từng nhóm môn/môn

**Bảng 2.4.** Bảng dữ liệu điểm trung bình theo từng nhóm môn/môn

| Tên trường | Mô tả |
|---|---|
| Student_Code | Mã sinh viên |
| Semester | Kỳ học |
| Term_ID | Kỳ triển khai |
| Major_ID | Mã ngành học của sinh viên |
| Value | Cờ đánh giá qua môn |
| Average_Grade | Điểm trung bình |
| Prefix_Subject | Mã nhóm môn |

- Dữ liệu về môn học theo khung chương trình

**Bảng 2.5.** Bảng dữ liệu về môn học theo khung chương trình

| Tên trường | Mô tả |
|---|---|
| Major | Tên ngành |
| Specialized_Major | Tên chuyên ngành |
| Period_Name | Kỳ học |
| Subject_Name | Tên môn học |
| Subject_Code | Mã môn học |
| Number_Of_Credit | Số tín chỉ |

**Đặc điểm của dữ liệu:**

- **Student Code:** Mỗi sinh viên chỉ có duy nhất một mã sinh viên, trừ trường hợp sinh viên chuyển ngành với mã số mới hoặc nhập học từ đầu sẽ được cấp mã số mới.
- **Semester:** Kỳ học của sinh viên được quyết định bởi khung chương trình đào tạo tương ứng mà sinh viên đăng ký học. Các sinh viên bị nợ môn sẽ được định nghĩa là ở trạng thái học quá thời gian quy định nhưng chưa thể tốt nghiệp.
- **Status:** Trạng thái học của sinh viên gồm nhiều loại, trong đó THO được định nghĩa là nghỉ học, HDI là trạng thái lên kỳ bình thường. Ngoài ra còn có TN là tạm ngưng, CHO là chờ xếp lớp. Để phù hợp với phạm vi của nghiên cứu, trạng thái học sẽ chỉ được giới hạn ở 2 trạng thái chính là THO và các trạng thái còn lại xếp chung là HDI.

- **Completion Status:** Sinh viên có thể có duy nhất một trạng thái học trong kỳ học hoặc có nhiều hơn một nếu sinh viên có quyết định nghỉ học ngay trong kỳ học đó.

## 2.2. Tiền xử lý dữ liệu

- Lịch sử trạng thái sinh viên

**Bảng 2.6.** Bảng kết quả kiểm tra dữ liệu lịch sử trạng thái sinh viên

| Trường dữ liệu | Số dữ liệu không hợp lệ |
|---|---|
| Student_Code | 0 |
| Status | 0 |
| Semester | 0 |
| Major_ID | 0 |
| Campus_Code | 0 |
| Term_ID | 0 |

- Dữ liệu tỉ lệ điểm danh của sinh viên

**Bảng 2.7.** Bảng kết quả kiểm tra dữ liệu tỉ lệ điểm danh của sinh viên

| Trường dữ liệu | Số dữ liệu không hợp lệ |
|---|---|
| Term | 0 |
| User_Login | 0 |
| User_Code | 0 |
| Total_Attendance | 0 |
| Total_Activity | 0 |
| Percentage | 0 |
| Up_To_Semester | 0 |

- Dữ liệu điểm trung bình của sinh viên

**Bảng 2.8.** Bảng kết quả kiểm tra dữ liệu điểm trung bình của sinh viên

| Trường dữ liệu | Số dữ liệu không hợp lệ |
|---|---|
| User_Login | 0 |
| User_Code | 0 |
| Term_ID | 0 |
| Major_ID | 0 |
| Campus | 0 |
| Value | 0 |
| Average_Grade | 101 |
| Prefix_Subject | 0 |

Ta tiến hành lọc bớt các dữ liệu Null hoặc NaN ở trường dữ liệu Average_Grade dựa theo kết quả thu được từ kiểm tra.

- Dữ liệu điểm trung bình theo từng nhóm môn

**Bảng 2.9.** Bảng kết quả kiểm tra dữ liệu điểm trung bình theo từng nhóm môn

| Trường dữ liệu | Số dữ liệu không hợp lệ |
|---|---|
| User_Login | 0 |
| User_Code | 0 |
| Up_To_Semester | 0 |
| Term_ID | 0 |
| Major_ID | 0 |
| Campus | 0 |
| Value | 0 |
| Total_Grade | 0 |
| Total_Credit | 0 |
| Average_Grade | 19 |

Ta tiến hành lọc bớt các dữ liệu Null hoặc NaN ở trường dữ liệu Average_Grade dựa theo kết quả thu được từ kiểm tra.

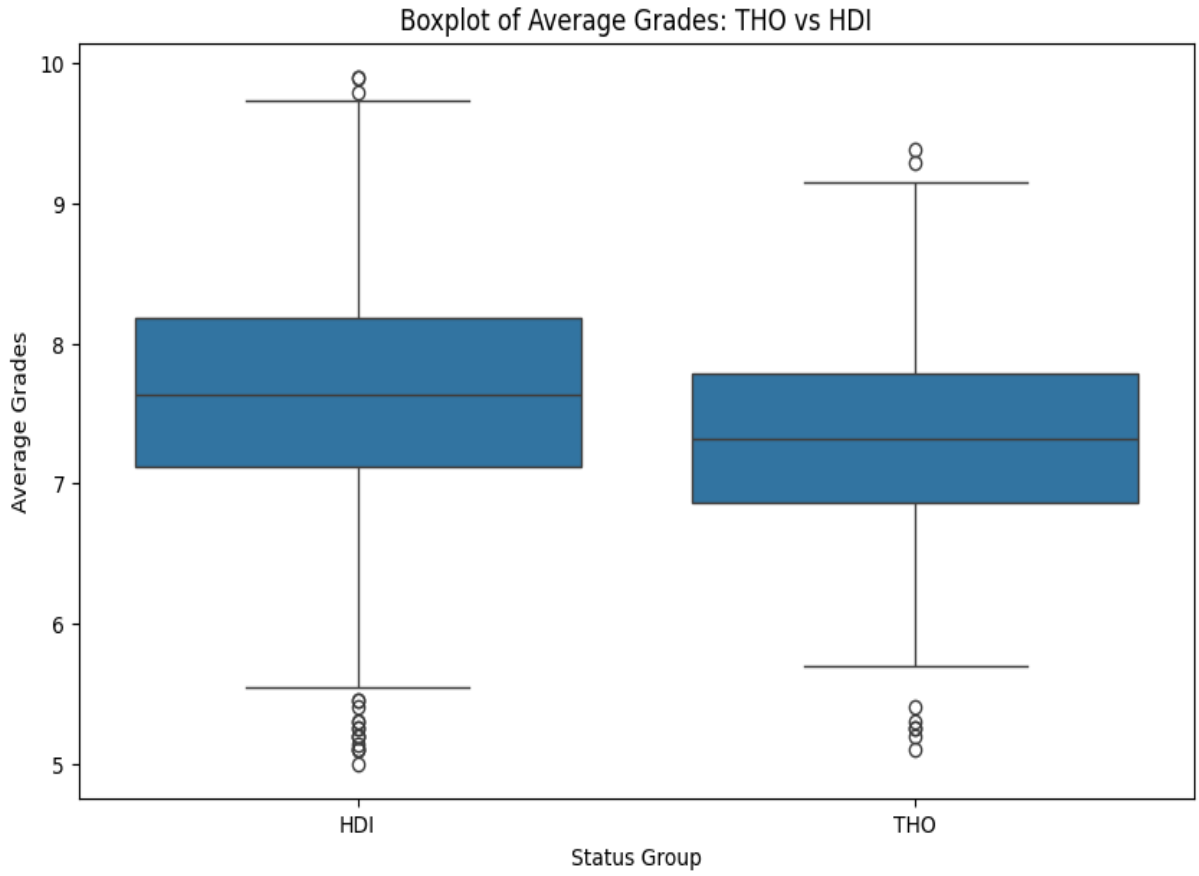
- Dữ liệu môn học theo khung chương trình

**Bảng 2.10.** Bảng kết quả kiểm tra dữ liệu môn học theo khung chương trình

| Trường dữ liệu | Số dữ liệu không hợp lệ |
|---|---|
| ID | 0 |
| Subject_Name | 0 |
| Subject_Code | 0 |
| Skill_Code | 0 |
| Number_Of_Credit | 0 |

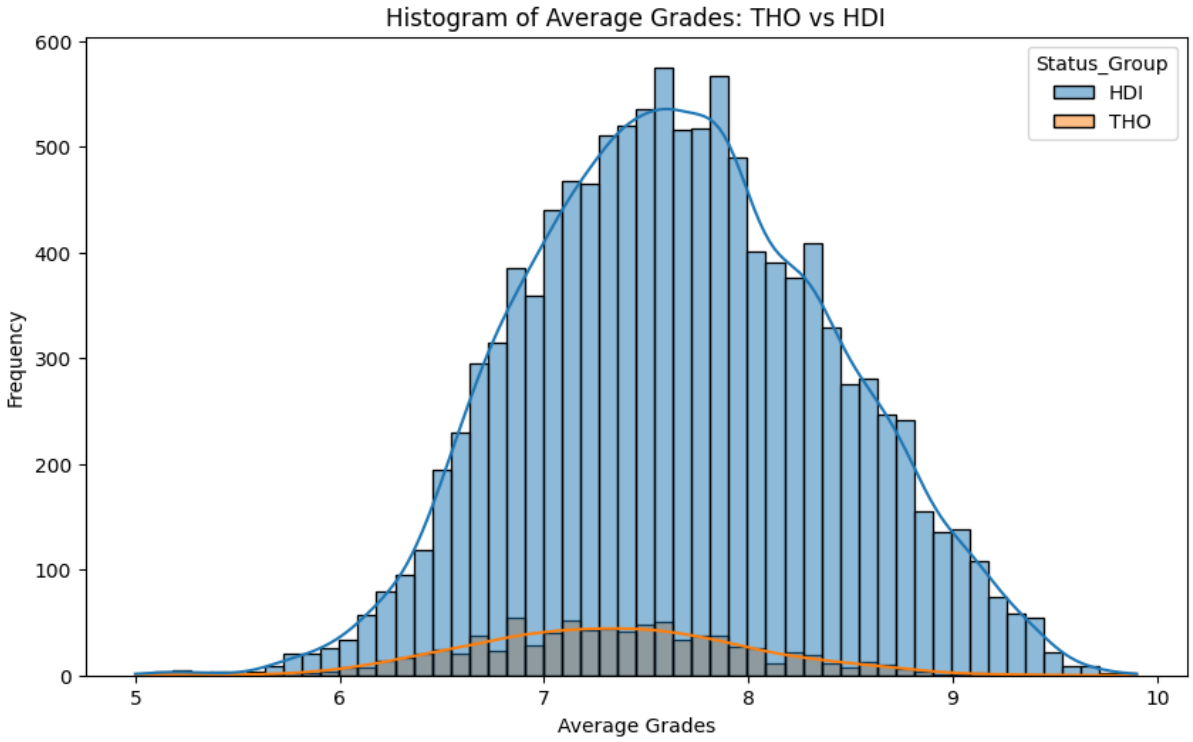
## 2.3. Trực quan hóa dữ liệu

- **Average Score – Điểm trung bình môn học**




**Hình 2.1.** Biểu đồ Boxplot Average Score giữa hai nhóm sinh viên

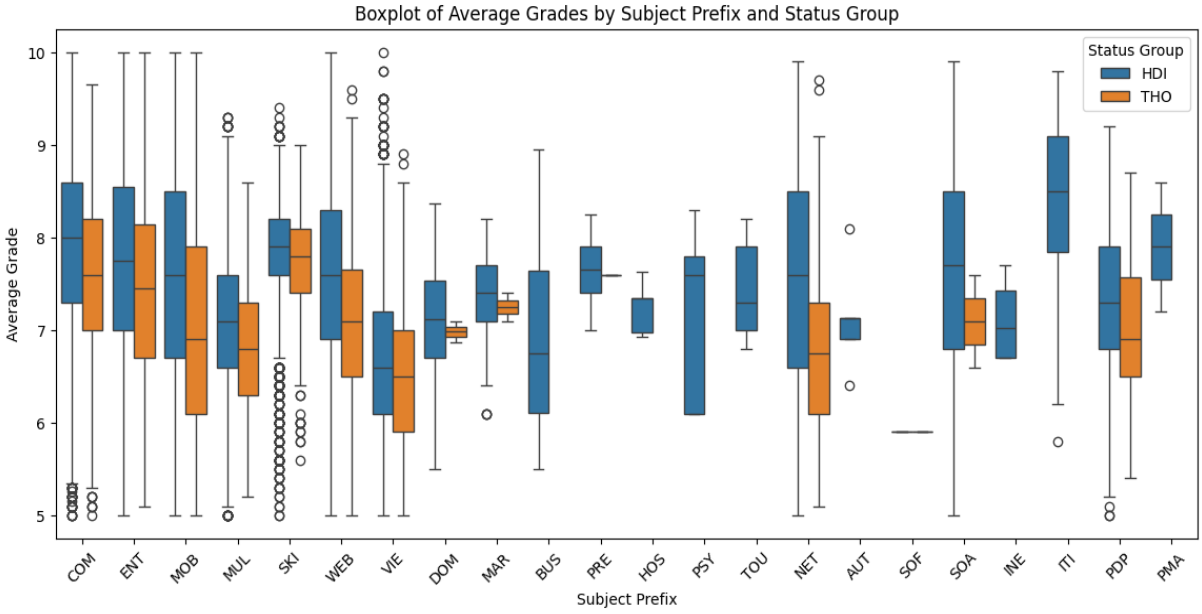
Dữ liệu điểm trung bình cho thấy nhóm bỏ học (THO) có điểm trung bình thấp hơn so với nhóm tiếp tục theo học (HDI) một khoảng chênh lệch chưa thực sự rõ ràng. Khoảng tứ phân vị IQR cũng cho thấy khoảng điểm trung bình của sinh viên THO thấp hơn so với sinh viên HDI. Ta có thể tạm thời đánh giá rằng, mức điểm trung bình sẽ có ảnh hưởng tới quyết định bỏ học của sinh viên khi mà số điểm càng thấp sẽ càng phổ biến ở những sinh viên bỏ học.



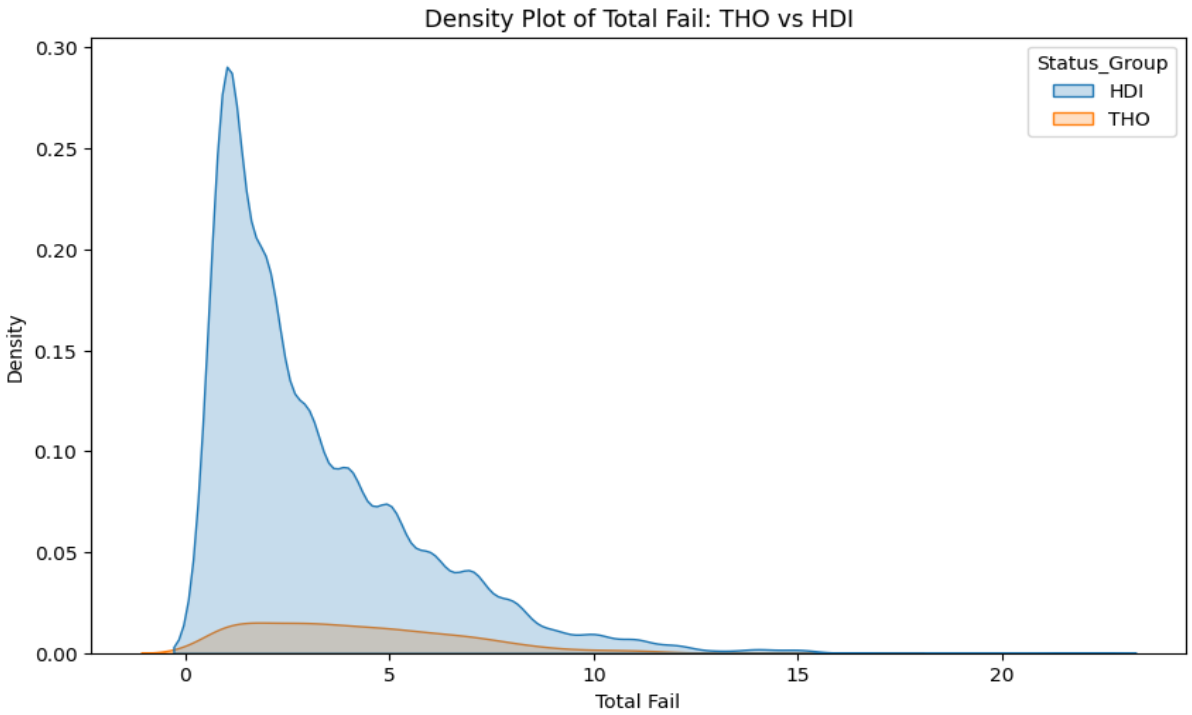

**Hình 2.2.** Biểu đồ Histogram Average Score giữa hai nhóm sinh viên

Biểu đồ tần suất cũng cho thấy mức điểm của nhóm THO có sự phân bố lệch về phía bên trái hơn so với HDI một mức nhỏ và tần suất phân bố của nhóm THO chỉ chiếm khoảng 1/10 so với HDI.

Đối với từng nhóm môn, ta thấy một số môn nhất định có khoảng điểm chênh lệch rõ ràng giữa 2 nhóm là các môn thuộc ngành Công nghệ thông tin (COM, MOB, WEB, NET, …) và các môn Tiếng Anh (ENT), Kỹ năng mềm (PDP). Ta sẽ đặc biệt quan tâm tới các nhóm môn này ở phần sau.



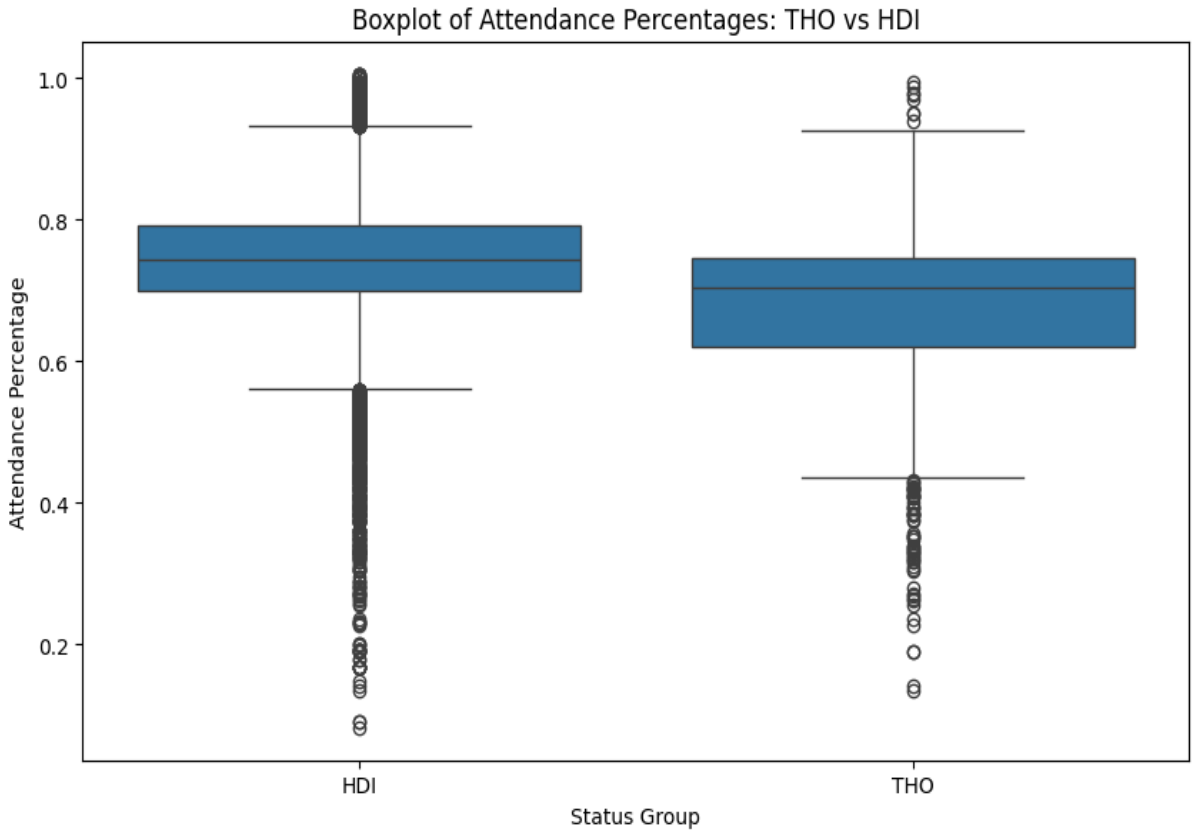

**Hình 2.3.** Biểu đồ Boxplot Average Score với từng nhóm môn giữa hai nhóm sinh viên




**Hình 2.4.** Biểu đồ mật độ số lần trượt môn giữa hai nhóm sinh viên

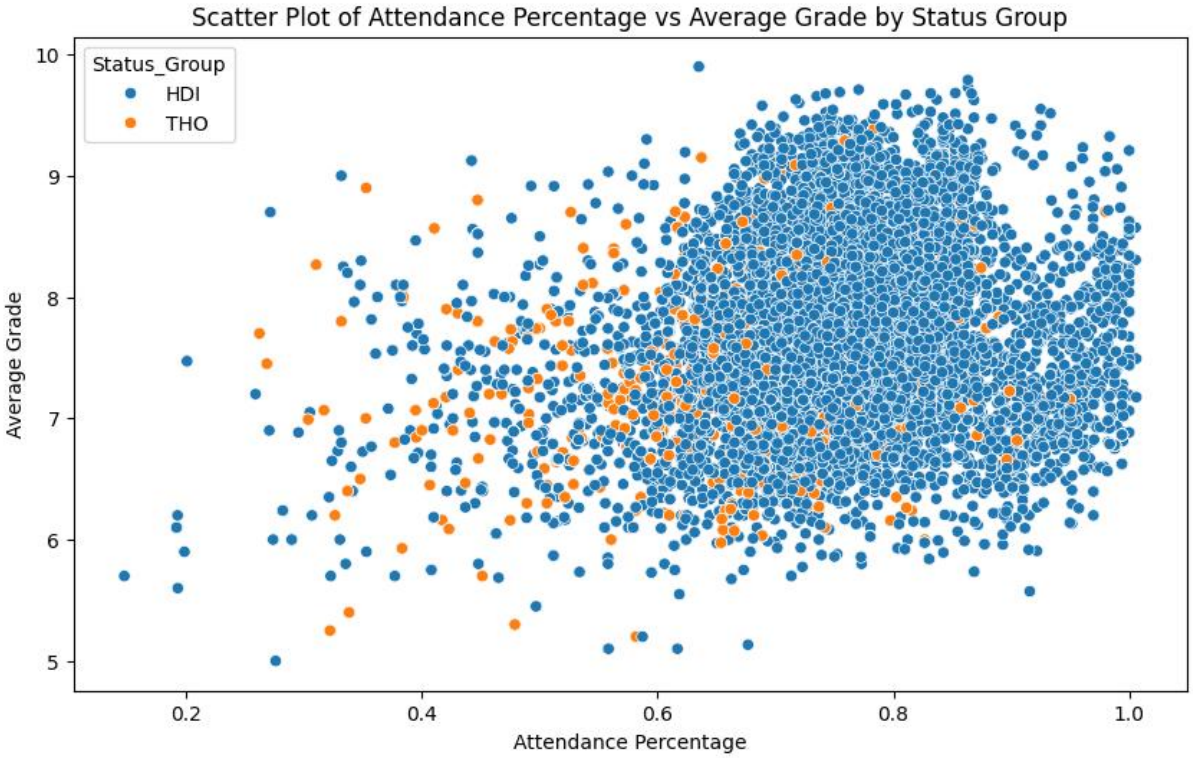
Dựa vào biểu đồ phân bố mật độ số lần trượt môn của 2 nhóm, ta có thể thấy nhóm THO thường có xu hướng trượt môn nhiều lần là kết quả từ sự chênh lệch điểm số được đánh giá ở trên.

- **Attendance Rate – Tỉ lệ điểm danh**

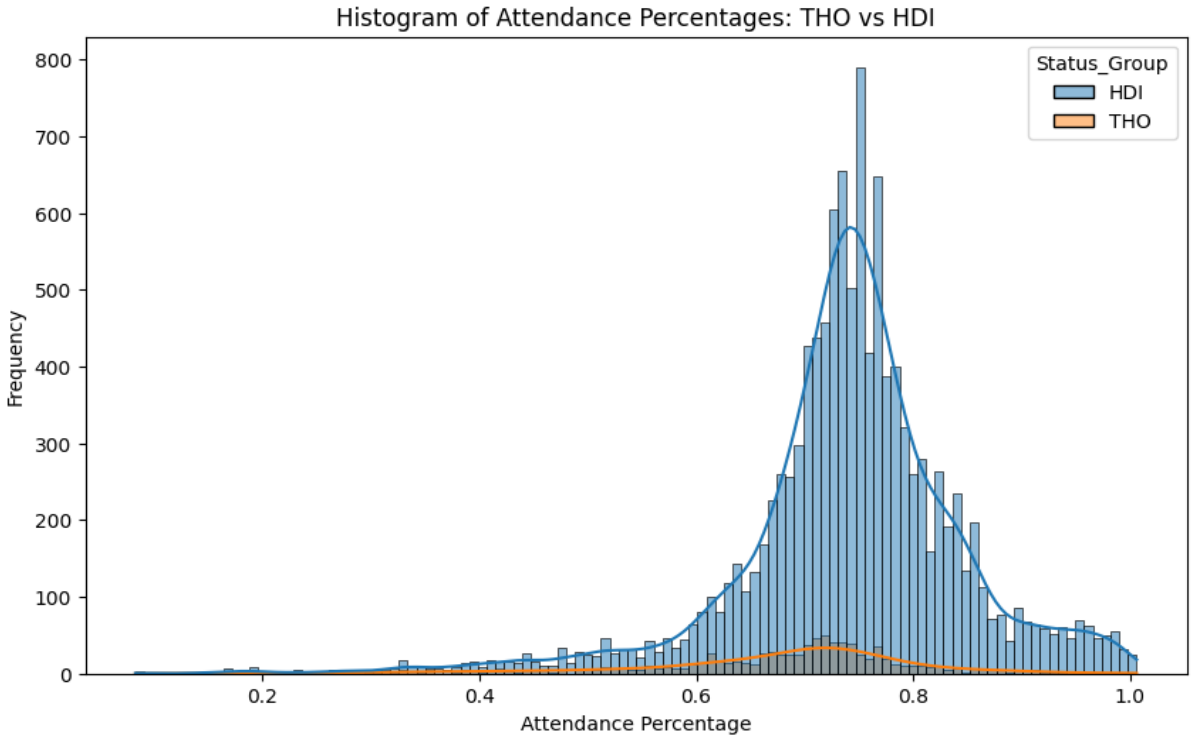



**Hình 2.5.** Biểu đồ Boxplot Attendance Percentages giữa hai nhóm sinh viên

Dữ liệu điểm danh cho thấy nhóm HDI có tỉ lệ điểm danh cao hơn so với nhóm THO và tỉ lệ này cũng dày đặc hơn so với sự phân bố của THO. Ta có thể có đánh giá sớm rằng tỉ lệ điểm danh có ảnh hưởng tới quyết định bỏ học cuối cùng của sinh viên,từ đó nhấn mạnh tầm quan trọng của việc đảm bảo sinh viên đi học đầy đủ để cho ra kết quả học tập cải thiện hơn.




**Hình 2.6.** Biểu đồ phân tán Average Score và Attendance Percentage giữa hai nhóm sinh viên




**Hình 2.7.** Biểu đồ tần suất Attendance Percantage giữa hai nhóm sinh viên

Biểu đồ tần suất cho thấy tỉ lệ điểm danh của nhóm THO chỉ khoảng từ 60 đến 80% nhưng có khá nhiều trường hợp phân bố ở các khoảng thấp từ từ 20 đến 50% tỉ lệ điểm danh. Số lượng chênh lệch giữa 2 nhóm dựa trên tỉ lệ điểm danh cũng có sự tương đồng nhất định với tần suất điểm trung bình của 2 nhóm.

Bởi lý do này, khi kiểm tra biểu đồ phân tán mối quan hệ giữa điểm trung bình và tỉ lệ điểm danh, ta có thể bước đầu phân lớp được dữ liệu thành 2 nhóm dựa trên trạng thái học cuối cùng của sinh viên dù chưa thực sự rõ ràng.

- **Ma trận tương quan (Correlation Matrix Heatmap)**


**Hình 2.8.** Ma trận tương quan giữa Attendance Rate và Average Score với trạng thái của sinh viên

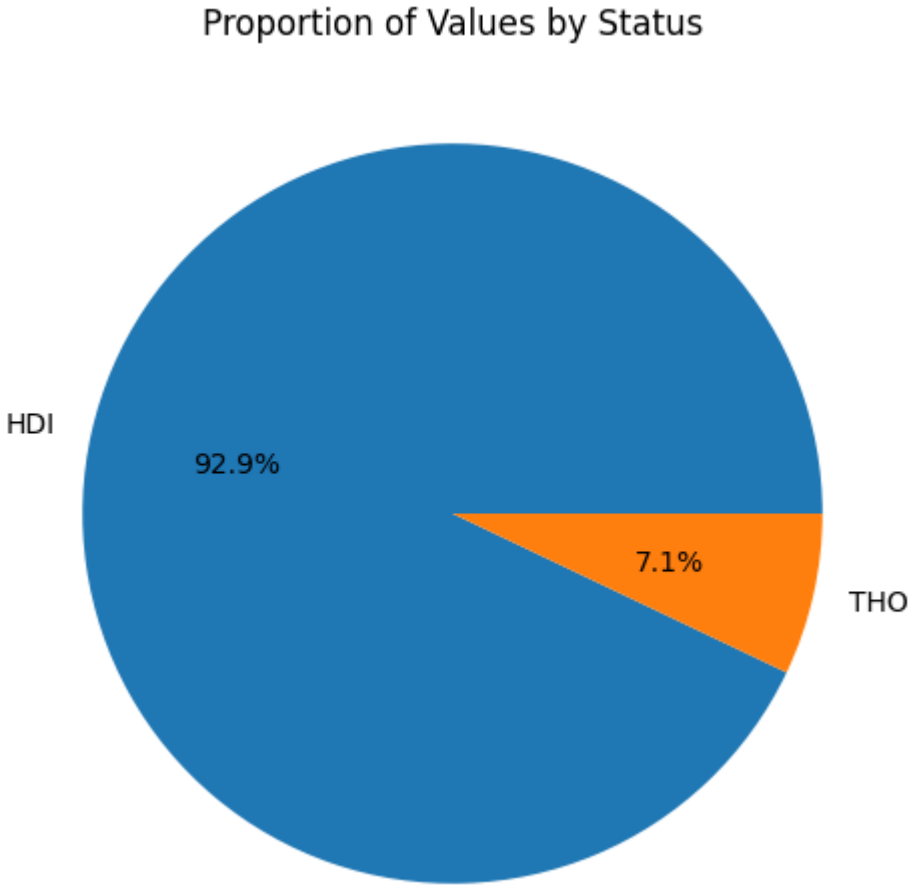
Ta có thể thấy không thực sự có yếu tố nào ảnh hưởng mạnh tới kết quả trạng thái học cuối cùng của sinh viên. Điều này có thể giải thích bởi số lượng dữ liệu lớn, chưa thực sự được cô lập các yếu tố quan trọng hoặc số lượng dữ liệu ngoại lệ gây ảnh hưởng tới kết quả đánh giá cuối cùng. Vấn đề này sẽ được đề cập và thảo luận ở chương tiếp theo.

# CHƯƠNG 3: NHỮNG VẤN ĐỀ LIÊN QUAN ĐẾN DỮ LIỆU VÀ HƯỚNG XỬ LÝ

## 3.1 Vấn đề liên quan tới dữ liệu

### 3.1.1. Đặt vấn đề

Sau khi quan sát bộ dữ liệu được cung cấp và thông qua các hình ảnh dữ liệu được trực quan hóa xuất hiện một vấn đề lớn là sự mất cân bằng dữ liệu giữa số lượng sinh viên của hai trạng thái tiếp tục theo học (HDI) và bỏ học (THO).




**Hình 3.1.** Tỉ lệ giữa hai nhóm sinh viên

Biểu đồ cho thấy số lượng sinh viên THO chỉ chiếm chưa tới 10% tổng số lượng sinh viên theo học. Kết hợp cùng số lượng ngoại lệ xuất hiện đã khiến cho việc trực quan hóa gặp một số vấn đề, cụ thể nhất là chưa làm rõ được mức độ ảnh hưởng của yếu tố tới kết quả trạng thái cuối cùng của sinh viên.

### 3.1.2. Mất cân bằng dữ liệu

Mất cân bằng dữ liệu (Imbalanced Dataset) là một trong những hiện tượng phổ biến của bài toán phân loại nhị phân (binary classification). Trong trường hợp tỷ lệ giữa

dữ liệu 2 classes là 50:50 thì được coi là cân bằng. Khi có sự khác biệt trong phân phối giữa 2 classes, chẳng hạn 60:40 thì dữ liệu có hiện tượng mất cân bằng.

Hầu hết các bộ dữ liệu đều khó đạt được trạng thái cân bằng mà luôn có sự khác biệt về tỷ lệ giữa 2 classes. Đối với các trường hợp mất cân bằng nhẹ (Ví dụ: tỉ lệ 60:40) thì sẽ không ảnh hưởng đáng kể tới khả năng dự báo của mô hình. Tuy nhiên nếu hiện tượng mất cân bằng nghiêm trọng xảy ra (Ví dụ: tỉ lệ 90:10) thường sẽ dẫn tới ngộ nhận về chất lượng của mô hình. Khi đó các thước đo đánh giá mô hình là độ chính xác (accuracy) có thể đạt được rất cao mà không cần tới mô hình.

Ngoài ra, mất cân bằng dữ liệu nghiêm trọng thường dẫn tới dự báo kém chính xác trên nhóm thiểu số bởi đa phần kết quả dự báo ra thường thiên về 1 nhóm là nhóm đa số và rất kém trên nhóm thiểu số (biased learning). Trong khi đó tầm quan trọng của việc dự báo được chính xác mẫu thuộc nhóm thiểu số lại lớn hơn nhiều so với dự báo mẫu thuộc nhóm đa số.

Để cải thiện kết quả dự đoán, chúng ta cần những điều chỉnh thích hợp để mô hình đạt được một độ chính xác cao trên nhóm thiểu số.

### 3.1.3. Các phương pháp giải quyết mất cân bằng

#### 3.1.3.1. Thay đổi metric để giải quyết vấn đề ngộ nhận chất lượng (Misleading Accuracy)

Accuracy không nên là đơn vị duy nhất để đánh giá chất lượng khi làm việc với những bộ dữ liệu mất cân bằng. Ngoài Accuracy ta có thể sử dụng một số đại lượng khác để đánh giá:

- Confusion Matrix: Là một phương pháp đánh giá kết quả của những bài toán phân loại với việc xem xét cả những chỉ số về độ chính xác và độ bao quát của các dự đoán cho từng lớp. Confusion Matrix gồm 4 chỉ số để đánh giá là True Positive (TP) – số lượng dự đoán chính xác , True Negative (TN) – số lượng dự đoán chính xác một cách gián tiếp, False Positive (FP) – số lượng các dự đoán sai lệch và False Negative (FN) – số lượng các dự đoán sai lệch một cách gián tiếp.
- Precision: Được định nghĩa là tỉ lệ số điểm Positive mô hình dự đoán đúng trên tổng số điểm mô hình dự đoán là Positive.
- Recall: Được định nghĩa là tỉ lệ số điểm Positive mô hình dự đoán đúng trên tổng số điểm thật sự là Positive.
- F1 Score (or F-score): Được định nghĩa là tham số trung bình điều hòa (Harmonic mean) giữa Precision và Recall và có công thức:

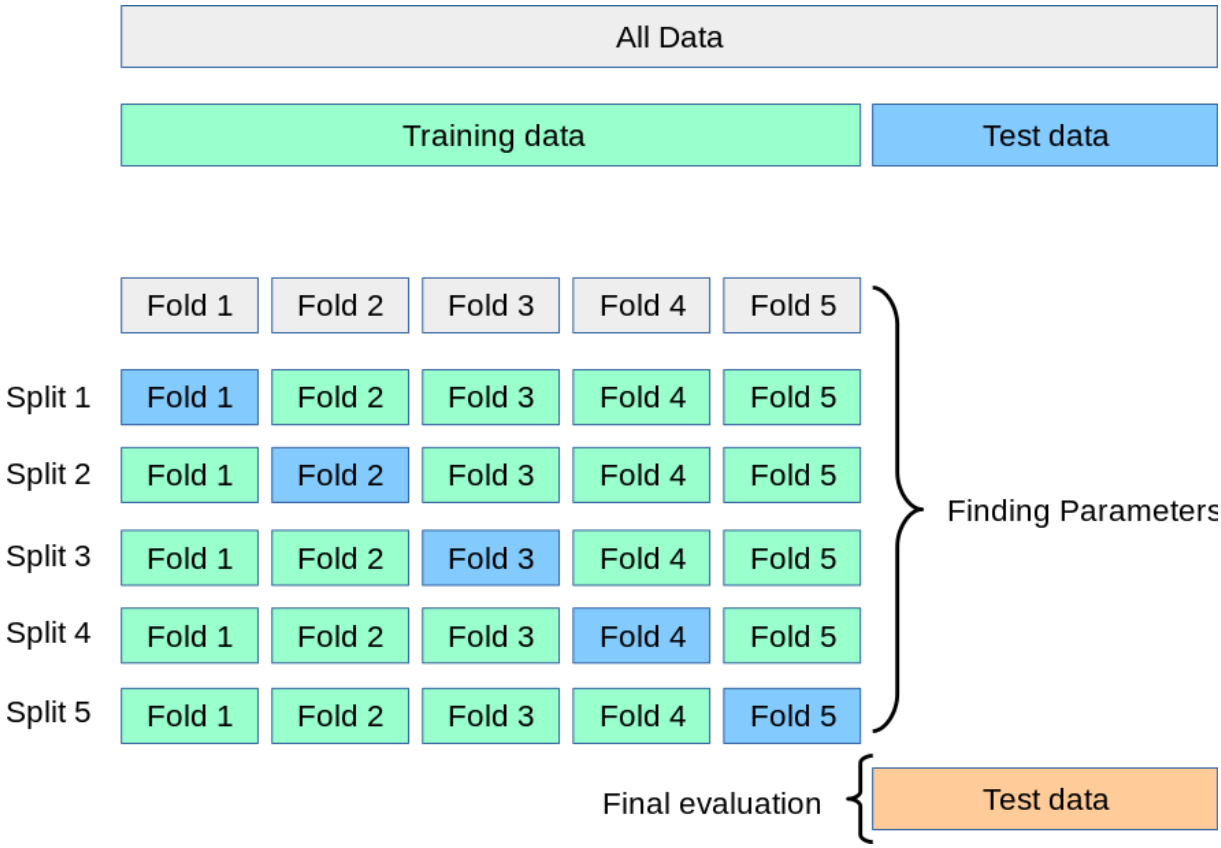
$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precesion} + \text{Recall}}$$

### 3.1.3.2. Resampling và Ensembling để giải quyết vấn đề thiên kiến dữ liệu (Biased Learning)

#### a. Resampling

Resampling là một phương pháp liên quan đến việc lặp lại việc lấy mẫu từ tập dữ liệu huấn luyện. Các mẫu này sau đó được sử dụng để làm mới lại một mô hình cụ thể để thu thập thêm thông tin về mô hình đã điều chỉnh. Mục tiêu là thu thập thêm thông tin về một mẫu và cải thiện độ chính xác và ước lượng không chắc chắn.

- Undersampling (Giảm mẫu): Undersampling là một nhóm các kỹ thuật được thiết kế để cân bằng lớp cho một tập dữ liệu có phân phối lớp bị lệch. Các kỹ thuật lấy mẫu loại bỏ các mẫu dữ liệu khỏi tập dữ liệu ở lớp đa số để cân bằng tốt hơn với lớp thiểu số, chẳng hạn như giảm độ lệch từ 1:100 xuống 1:10, 1:2 hoặc thậm chí là 1:1.
- Oversampling (Tăng mẫu): Các kỹ thuật Oversampling bổ sung thêm mẫu ở lớp thiểu số để đưa tập dữ liệu đạt mức cân bằng. Oversampling làm tăng số lượng mẫu của lớp thiểu số bằng cách sao chép các mẫu của lớp thiểu số. Mặc dù nó không làm tăng thông tin mẫu nhưng với một số kỹ thuật khác nhau, nó làm tăng thêm các trường hợp khác nhau ở tập mẫu mới.
- Cross-validation (Kiểm chứng chéo): Trong nhiều trường hợp, chúng ta có số lượng dữ liệu hạn chế để xây dựng mô hình. Nếu lấy quá nhiều dữ liệu từ tập training để làm dữ liệu validation, phần dữ liệu còn lại trong tập training sẽ không đủ để xây dựng mô hình. Do đó, tập validation cần phải thật nhỏ để đảm bảo lượng dữ liệu training đủ lớn. Tuy nhiên, điều này lại dẫn đến một vấn đề khác là khi tập validation quá nhỏ, hiện tượng overfitting có thể xảy ra với phần dữ liệu training còn lại. Cross-validation sẽ là giải pháp cho vấn đề này. Một cách thường được sử dụng là chia tập training ra *k* tập con không có phần tử chung, có kích thước gần bằng nhau. Tại mỗi lần kiểm thử, được gọi là run, một trong số *k* tập con được lấy ra làm validate set. Mô hình sẽ được xây dựng dựa vào hợp của *k-1* tập con còn lại. Mô hình cuối được xác định dựa trên trung bình các train error và validation error. Cách làm này còn có tên gọi là **K-Fold Cross-Validation**.




**Hình 3.2.** Mô tả thuật toán K-Fold Cross-Validation

**b. Ensembling**

Ensemble learning là một kỹ thuật học máy tăng cường độ chính xác và tính linh hoạt trong dự báo bằng cách kết hợp các dự đoán từ nhiều mô hình. Nó nhằm mục đích giảm thiểu các lỗi hoặc thiên hướng có thể tồn tại trong các mô hình cá nhân bằng cách tận dụng trí tuệ tập thể của bộ hợp.

Khái niệm cơ bản đằng sau Ensemble learning là kết hợp các đầu ra của các mô hình đa dạng để tạo ra một dự đoán chính xác hơn. Bằng cách xem xét nhiều góc độ và sử dụng các điểm mạnh của các mô hình khác nhau, Ensemble learning cải thiện hiệu suất tổng thể của hệ thống học. Phương pháp này không chỉ cải thiện độ chính xác mà còn cung cấp tính linh hoạt đối với các không chắc chắn trong dữ liệu. Bằng cách kết hợp các dự đoán từ nhiều mô hình một cách hiệu quả, Ensemble learning đã được chứng minh là một công cụ mạnh mẽ trong nhiều lĩnh vực, mang lại các dự báo mạnh mẽ và đáng tin cậy hơn.

- Bagging: xây dựng một lượng lớn các models (thường là cùng loại) trên những subsamples khác nhau từ tập training dataset một cách song song nhằm đưa ra dự đoán tốt hơn.
- Boosting: xây dựng một lượng lớn các models (thường là cùng loại). Tuy nhiên quá trình huấn luyện trong phương pháp này diễn ra tuần tự theo chuỗi (sequence).

Trong chuỗi này mỗi model sau sẽ học cách sửa những errors của model trước (hay nói cách khác là dữ liệu mà model trước dự đoán sai).

- Stacking: xây dựng một số models (thường là khác loại) và một mô hình supervisor model, mô hình này sẽ học cách kết hợp kết quả dự báo của một số mô hình một cách tốt nhất.

**c. Phạt mô hình để trực tiếp cải thiện tính chính xác của mô hình**

Việc dự báo sai một quan sát thuộc mẫu đa số sẽ ít nghiêm trọng hơn so với dự báo sai một quan sát thuộc mẫu thiểu số. Xuất phát từ ý tưởng đó chúng ta sẽ phạt nặng hơn đối với sai số dự báo thuộc nhóm thiểu số bằng cách gán cho nó một trọng số lớn hơn trong công thức của hàm loss function.

**d. Thu thập thêm dữ liệu quan sát**

Thông thường với các mô hình mà số lượng quan sát trong nhóm thiểu số quá nhỏ sẽ không thể đại diện cho toàn bộ các nguyên nhân dẫn đến quan sát rơi vào nhóm thiểu số. Để mô hình học được bao quát hơn các khả năng, chúng ta cần gia tăng kích thước mẫu thiểu số bằng cách thu thập thêm các quan sát thực tế thuộc nhóm thiểu số.

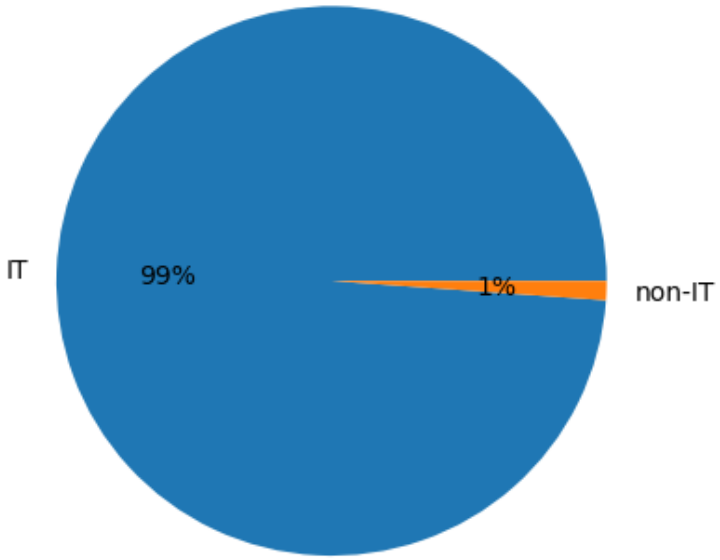
**e. Thu thập thêm biến**

Mô hình có kết quả kém có thể là do đang thiếu những biến quan trọng có ảnh hưởng lớn tới xác định hành vi của nhóm thiểu số. Để bổ sung thêm biến, thu thập ý kiến chuyên gia là một biện pháp quan trọng để tạo ra bộ dữ liệu chất lượng cho huấn luyện mô hình. Các chuyên gia là những người có kinh nghiệm và hiểu biết chuyên sâu về đặc tính của các nhóm. Do đó họ sẽ đưa ra nhiều rules nhận diện giúp ích cho phân loại.

## 3.2. Áp dụng vào bộ dữ liệu thực tế cung cấp

### 3.2.1. Tái xử lý bộ dữ liệu

Quan sát bộ dữ liệu, ta nhận thấy số lượng sinh viên theo học các ngành về Công nghệ thông tin chiếm đa số các bản ghi. Để giải quyết vấn đề mất cân bằng dữ liệu, ta sẽ thử khoanh vùng lại bộ dữ liệu mới chỉ bao gồm các sinh viên thuộc ngành Công nghệ thông tin có thời gian bắt đầu học từ kỳ Summer 2018 cho tới kỳ Spring 2024.




**Hình 3.3.** Tỉ lệ giữa sinh viên ngành IT và các ngành khác

Bộ dữ liệu mới sẽ có cấu trúc mỗi bản ghi là kết quả học tập của sinh viên theo từng kỳ trải dài từ kỳ 1 cho tới kỳ 3. Dưới đây là khái quát về bộ dữ liệu mới này:

- Tổng số bản ghi được xét: 6840
- Tổng số cột: 11
- Có tổng số 524 bản ghi là các sinh viên có trạng thái kỳ học thứ 3 là bỏ học (THO) và 6316 bản ghi là các sinh viên tiếp tục học (HDI).

**Cấu trúc các cột:**

- Student_code (String): Các giá trị độc nhất là mã sinh viên cho từng sinh viên theo học.
- Semester_1 (Object): Cấu trúc dưới dạng một chuỗi JSON gồm thông tin của từng môn học trong kỳ học 1 bao gồm:
  - Full_subject_code: Mã môn học đầy đủ.
  - Attendance_rate: Tỉ lệ điểm danh cho môn học đó.
  - Passed: Cờ trạng thái sinh viên qua hay trượt môn.
  - Average_score: Điểm trung bình của môn học.
  - Number_of_credit: Số tín chỉ của môn học.
  - Learnt_times: Số lần học của sinh viên đối với môn đó (mặc định là 1).

- Semester_2 (Object): Giống như Semester_1 nhưng là thông tin của kỳ học 2.
- Semester_3 (Object): Giống như Semester_1 nhưng là thông tin của kỳ học 3.
- Semester_3_status (String): Trạng thái của sinh viên ở cuối kỳ học thứ 3, bao gồm 2 trạng thái được xét là HDI (High Distinction – Giỏi, tiếp tục được lên kỳ) và THO (Bỏ học). Một sinh viên có thể tồn tại đồng thời cả 2 trạng thái HDI và THO là 2 bản ghi khác nhau trong trường hợp sinh viên bỏ học sau khi được xét duyệt trạng thái hoặc sinh viên theo học lại và có kết quả kỳ thứ 3 mới.
- Semester_1_attendance_rate (Float64): Tỉ lệ điểm danh của kỳ học 1.
- Semester_1_average_score (Float64): Điểm trung bình của kỳ học 1.
- Semester_2_attendance_rate (Float64): Tỉ lệ điểm danh của kỳ học 2.
- Semester_2_average_score (Float64): Điểm trung bình của kỳ học 2.
- Semester_3_attendance_rate (Float64): Tỉ lệ điểm danh của kỳ học 3.
- Semester_3_average_score (Float64): Điểm trung bình của kỳ học 3.

Ví dụ bộ dữ liệu:

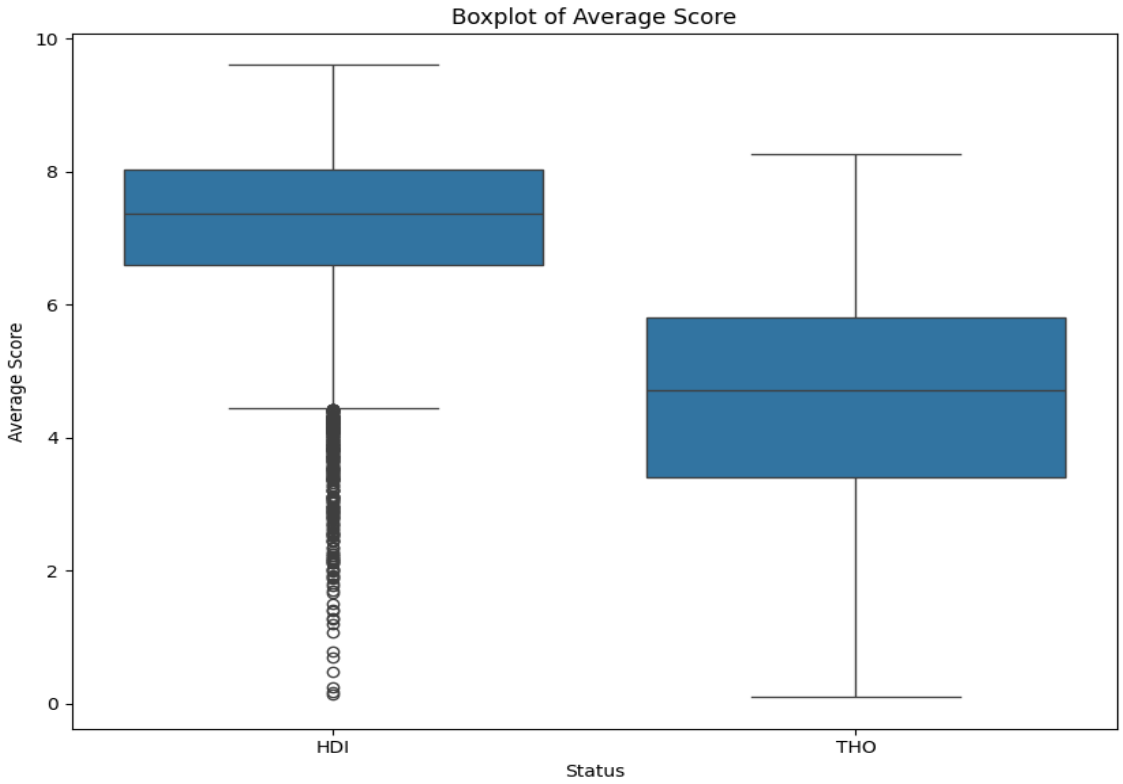
**Bảng 3.1.** Mô tả bảng dữ liệu mẫu

| student_code | semester_1 | semester_2 | semester_3 | semester_1_attendance_rate | semester_1_average_score |
|---|---|---|---|---|---|
| A1000 | {‘S1’: {…}} | {‘S2’: {…}} | {‘S3’: {…}} | 94.117 | 8.3 |
| A1001 | {‘S1’: {…}} | {‘S2’: {…}} | {‘S3’: {…}} | 96.125 | 7.1 |
| A1002 | {‘S1’: {…}} | {‘S2’: {…}} | {‘S3’: {…}} | 63.439 | 6.3 |

| semester_2_attendance_rate | semester_2_average_score | semester_3_attendance_rate | semester_3_average_score | semester_3_status |
|---|---|---|---|---|
| 95.558 | 8.2 | 82.089 | 8.7 | HDI |
| 89.126 | 7.6 | 87.603 | 8.4 | THO |
| 75.984 | 8.4 | 98.230 | 9.0 | HDI |

### 3.2.2. Trực quan hóa dữ liệu và đánh giá

- **Average Scores giữa 2 trạng thái**



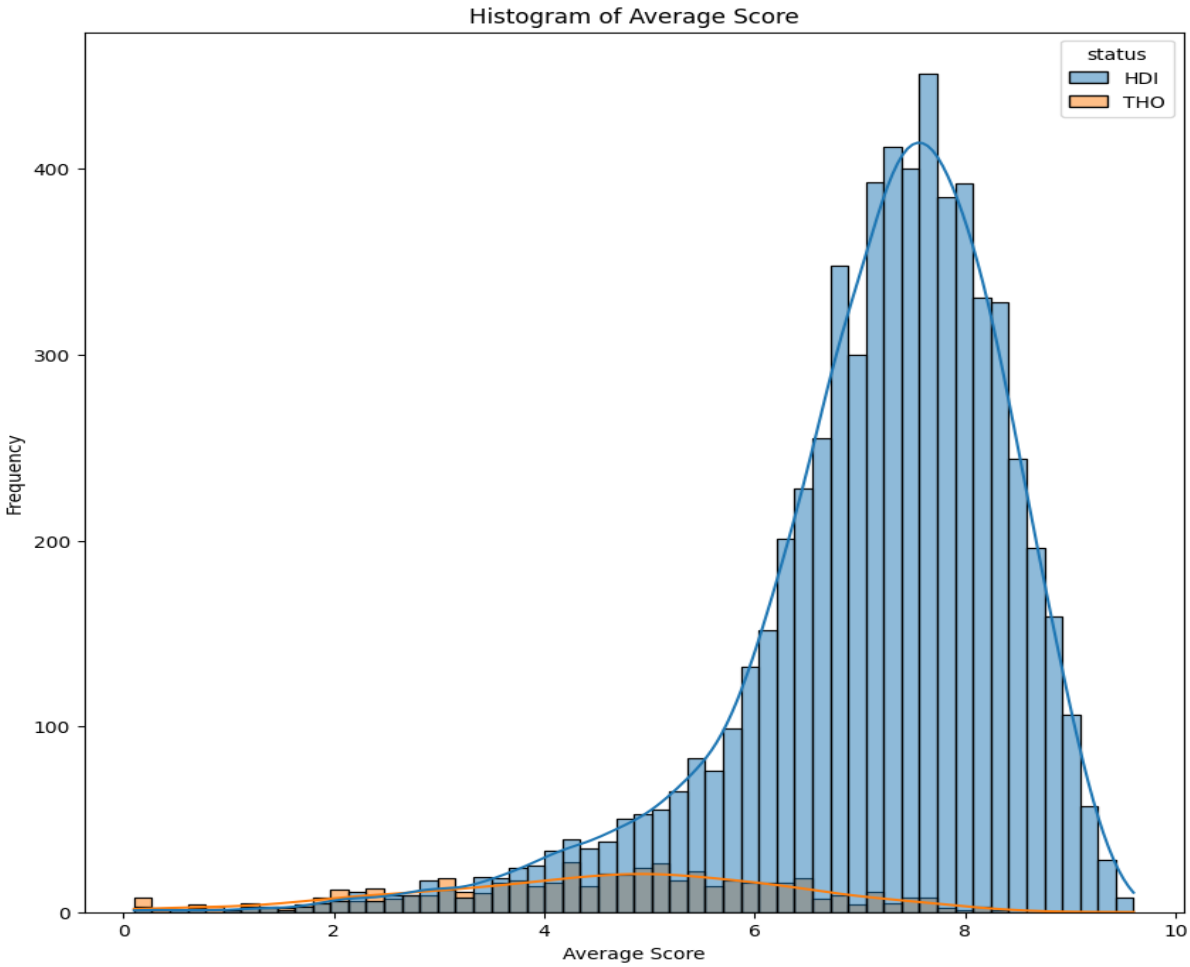

**Hình 3.4.** Biểu đồ Boxplot Average Score sau khi xử lý lại

Biểu đồ cho thấy giá trị trung vị của điểm trung bình đối những sinh viên tiếp tục học (HDI) là khoảng điểm 7, trong khi đó đối với những sinh viên bỏ học (THO) con số này chỉ là khoảng điểm 5.

Giá trị cận dưới đối với nhóm HDI là khoảng điểm 4 với số lượng lớn các điểm ngoại lệ nằm dưới giá trị cận này, cho thấy rằng có một tập hợp nhỏ các sinh viên vẫn sẽ tiếp tục học dù có điểm số thấp. Đối với nhóm THO, giá trị cận dưới này mở rộng xuống khoảng điểm 0.

Khoảng tứ phân vị (IQR) đối với nhóm HDI khá hẹp cho thấy số lượng điểm ổn định trong mức điểm khá trở lên (6 đến 8). Ngược lại, IQR đối với nhóm THO lại có sự trải rộng hơn, phản ánh mức độ đa dạng hơn về điểm số.

Biểu đồ boxplot đã cho thấy với mức điểm trung bình cao, sinh viên sẽ có xu hướng tiếp tục việc học trong khi với mức điểm thấp, sinh viên lại có xu hướng bỏ học nhiều hơn.




**Hình 3.5.** Biểu đồ tần suất Average Score sau khi xử lý lại

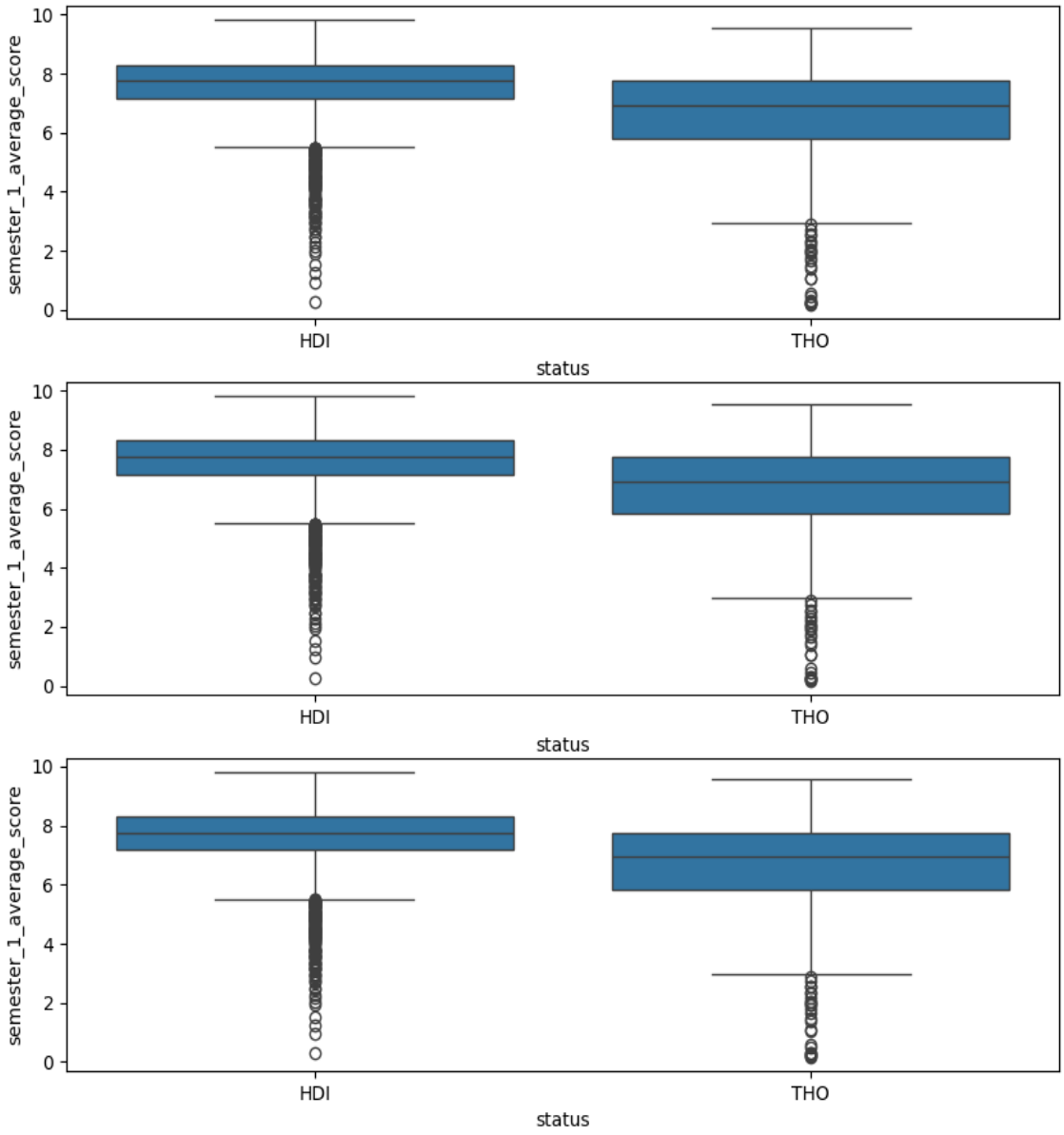
Biểu đồ tần suất cũng cho thấy phổ điểm giữa 2 nhóm có sự phân hóa rõ rệt hơn, cho ra kết quả khá tương đồng với những gì được thể hiện ở biểu đồ Boxplot. Tuy nhiên, sự chênh lệch giữa tần suất của 2 nhóm vẫn khá lớn.

- **Average Scores theo từng kỳ học**

So sánh từng kỳ học, ta thấy nhóm HDI sẽ có số điểm ổn định trải dài qua cả 3 kỳ với giá trị trung vị là khoảng 8 điểm. Trong khi đó, nhóm THO có sự giảm nhẹ qua từng kỳ với mức điểm chỉ khoảng 7 điểm.

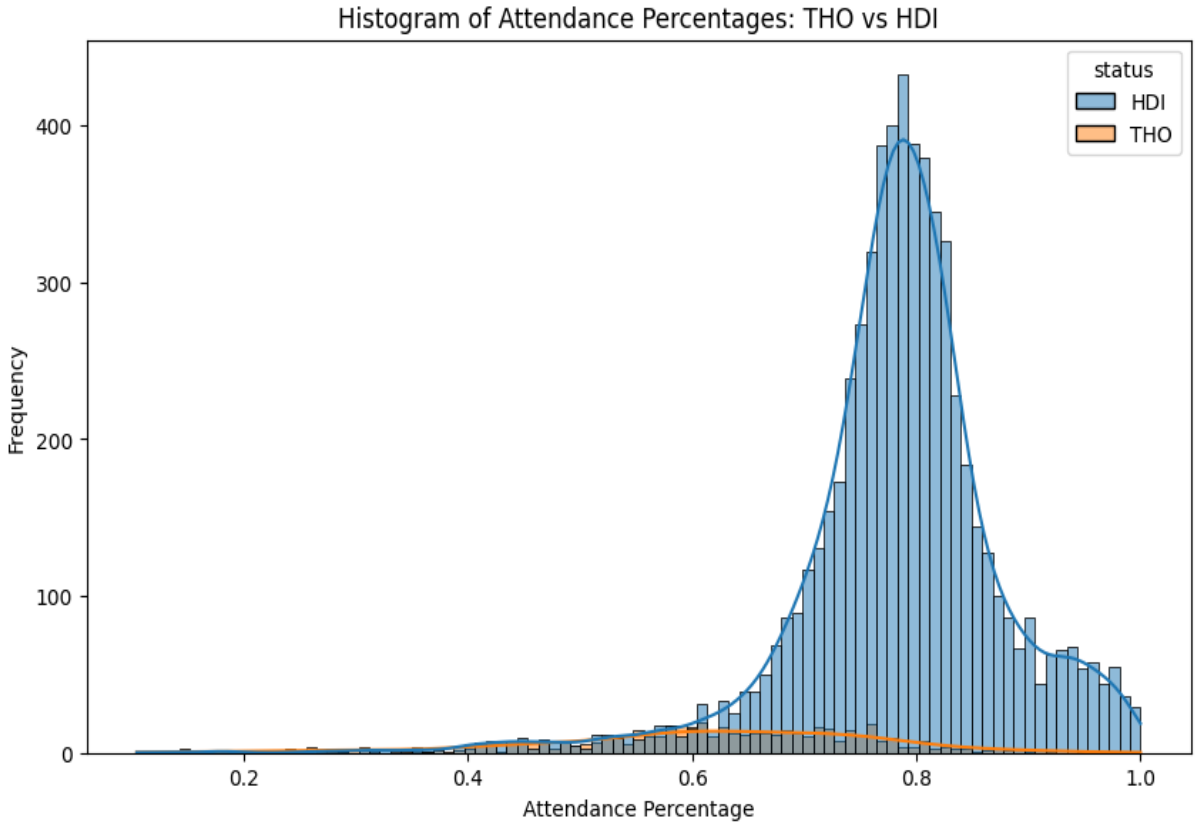
Khoảng tứ phân vị IQR cho thấy với nhóm HDI có mức điểm ổn định ít biến động (từ 7 tới 8.8) trong khi đó nhóm THO có sự phân hóa đa dạng hơn và càng mở rộng qua các kỳ (từ 6 tới 8).

Từ các đánh giá trên, ta có thể nhận xét rằng: Khoảng giá trị điểm trung bình của nhóm sinh viên THO có thể mở rộng xuống khoảng mức điểm yếu (2 – 4 điểm) và cần được đặc biệt quan tâm từ sớm để cải thiện mức điểm.




**Hình 3.6.** Biểu đồ Boxplot Average Score theo từng kỳ học

- **Attendance Rates giữa 2 trạng thái**




**Hình 3.7.** Biểu đồ Boxplot Attendance Percentage sau khi xử lý lại

Quan sát ta thấy nhóm sinh viên HDI có giá trị trung vị của tỉ lệ điểm danh vào khoảng 80% so với khoảng 65% của nhóm sinh viên THO.

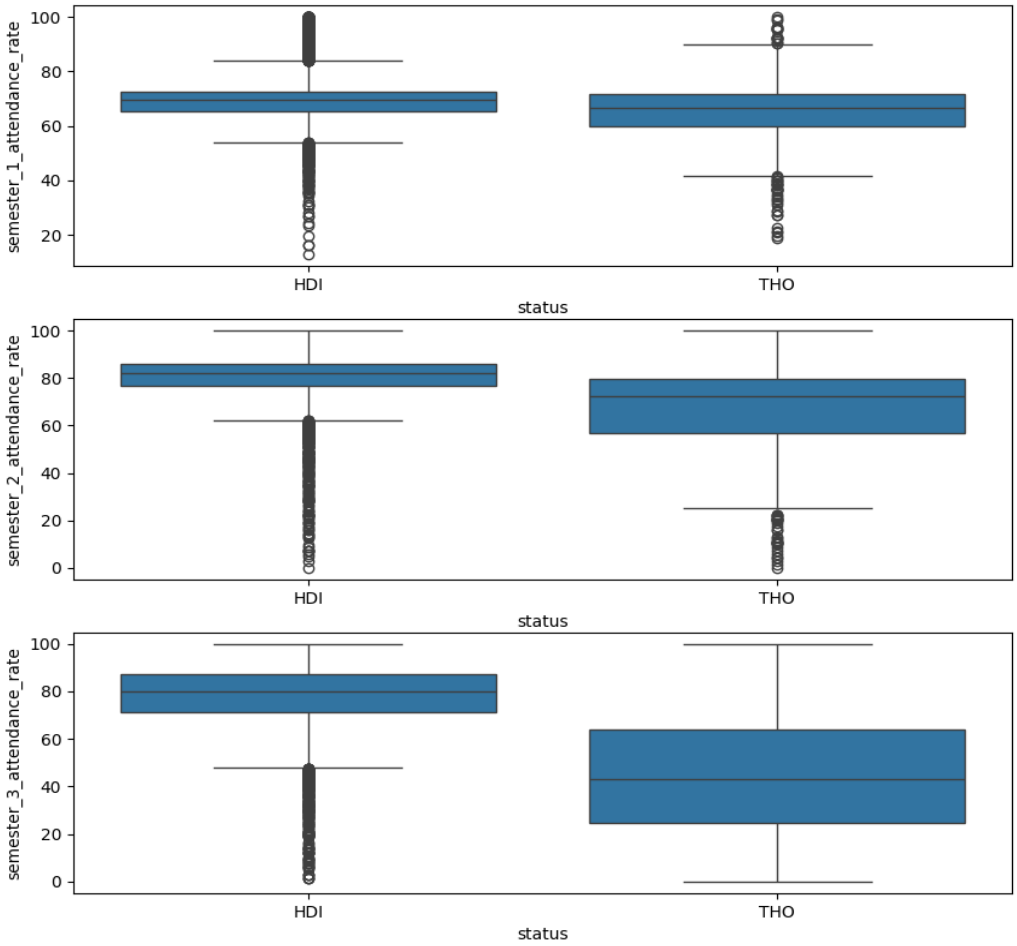
Khoảng IQR của nhóm HDI chỉ dao động trong khoảng hẹp từ 75% đến 85% trong khi đó với nhóm THO cho thấy sự đa dạng hơn về tỉ lệ điểm danh từ 55% đến 75%.

Ta có thể đánh giá rằng với mức chênh lệch tỉ lệ điểm danh như trên cho thấy những sinh viên đi học thường xuyên sẽ ít có khả năng bỏ học hơn. Vì vậy cần đặc biệt quan tâm tới việc khuyến khích sinh viên đi học đầy đủ để cải thiện kết quả giáo dục.

Biểu đồ tần suất cho thấy nhóm sinh viên bỏ học (THO) hầu hết chỉ đáp ứng được hơn phân nửa số buổi điểm danh cần thiết trong kỳ học. Điều này cũng củng cố thêm những nhận xét được đề cập ở phần trên.

- **Attendance Rates theo từng kỳ học**

Nhóm sinh viên HDI có giá trị trung vị ổn định ở khoảng khá cao dù có biến động qua các kỳ (khoảng từ 70% đến 80%) so sánh với nhóm THO thì lại có sự giảm dần rõ rệt qua các kỳ (từ khoảng 70-75% tụt xuống chỉ còn khoảng 40% ở kỳ 3).

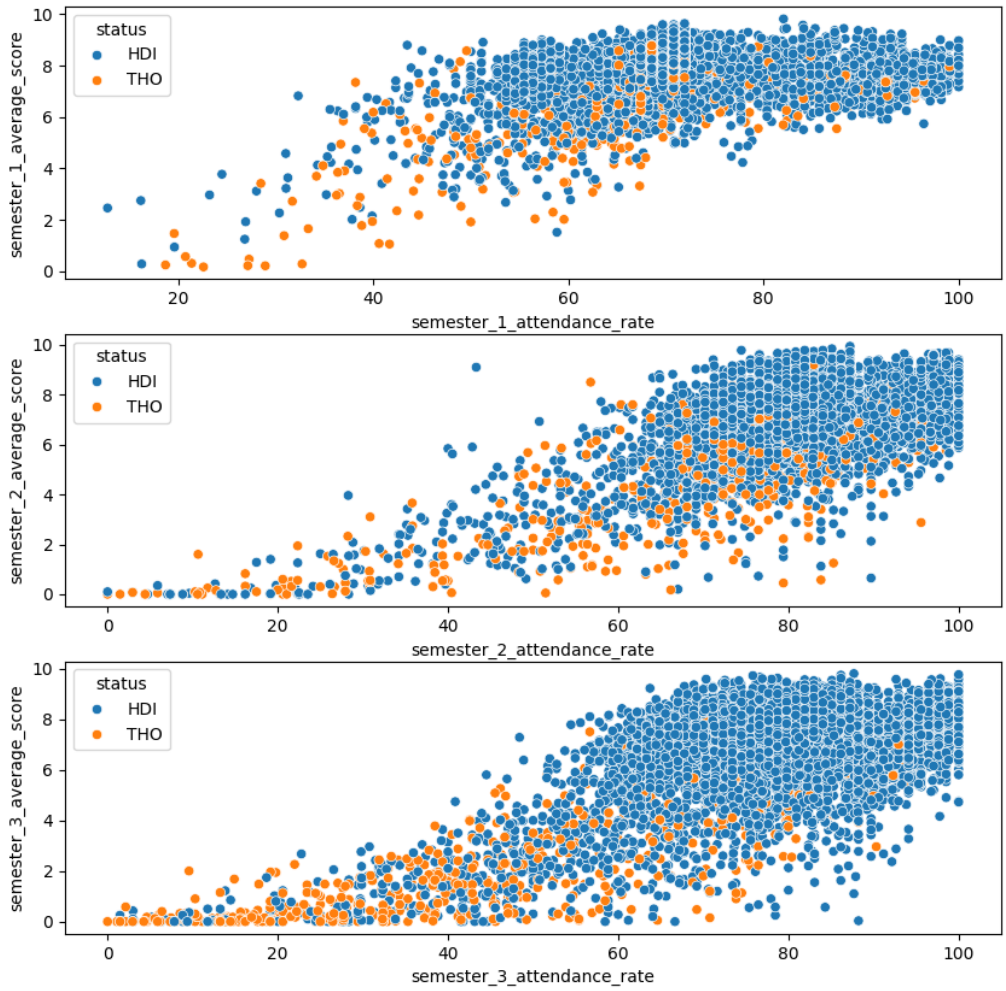



**Hình 3.8.** Biểu đồ tần suất Attendance Percentage sau khi xử lý lại

Cả 2 trạng thái đều cho thấy sự mở rộng dần về khoảng tỉ lệ điểm danh tuy nhiên đối với nhóm HDI chỉ tăng nhẹ độ rộng khoảng 5 đến 10%. Ngược lại đối với nhóm THO, tỉ lệ điểm danh tụt giảm nghiêm trọng từ khoảng 60-70% xuống chỉ còn 25-65% ở kỳ thứ 3. Với khoảng IQR mở rộng và dần nghiêng về cận 0 cho thấy tỉ lệ điểm danh ở nhóm THO trở nên thiếu ổn định và cần được đặc biệt quan tâm để cải thiện và giảm thiểu khả năng bỏ học của sinh viên. Việc sinh viên có tỉ lệ đi học đầy đủ đồng nghĩa với việc sinh viên sẽ có thể có kết quả giáo dục tốt hơn.

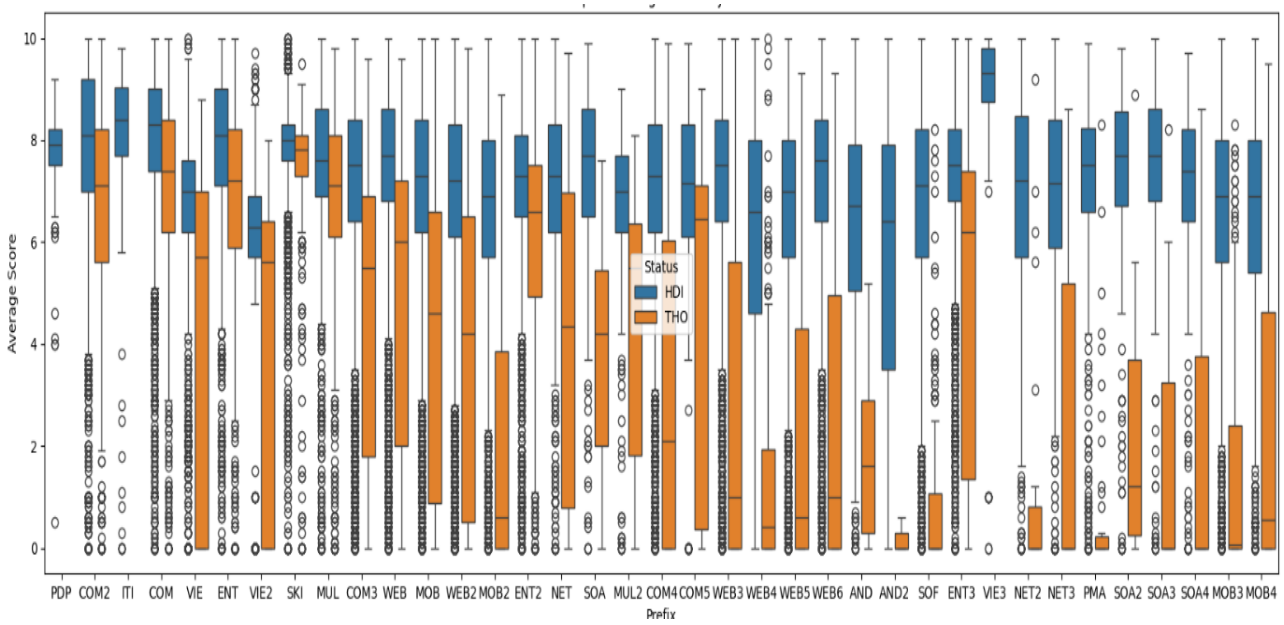
- **Attendance Rates so với Average Scores theo từng kỳ**

Cả 3 biểu đồ phân tán đều chỉ ra mối tương quan rõ ràng giữa tỉ lệ điểm danh và điểm trung bình của sinh viên giữa 2 nhóm. Điều này cho thấy rằng những sinh viên đi học thường xuyên và đầy đủ thường sẽ có điểm số trung bình cao hơn. Tuy nhiên, mức độ tương quan này chỉ thực sự rõ ràng ở các kỳ sau so với kỳ đầu.




**Hình 3.10.** Biểu đồ phân tán Attendance Rate và Average Score theo từng kỳ học

- **Mối quan hệ giữa Average Score từng nhóm môn với kết quả trạng thái học**




**Hình 3.11.** Biểu đồ Boxplot Average Score từng nhóm môn sau khi xử lý lại

Ở hầu hết các nhóm môn, những sinh viên nhóm HDI đều có số điểm cao hơn so với nhóm THO. Khoảng IQR cũng cho thấy nhóm HDI có màn thể hiện về điểm số tốt hơn và ít biến động hơn so với nhóm THO.

Với một số nhóm môn nhất định:

- Các nhóm môn COM, WEB, SOA, MOB cho thấy sự phân hóa rõ ràng giữa 2 nhóm khi mà những sinh viên tiếp tục học đều có điểm số cao hơn đáng kể. Đây cũng đồng thời là các nhóm môn thuộc về chuyên ngành của ngành Công nghệ thông tin.
- Các nhóm môn như ENT hay VIE cũng cho thấy sự khác biệt rõ ràng giữa 2 nhóm khi đây là các nhóm môn Tiếng Anh và các môn cơ bản.
- Các nhóm môn như ITI và COM2 lại cho thấy ít sự ảnh hưởng hơn một phần do ít sinh viên lựa chọn các nhóm môn này để học. Ta có thể cân nhắc loại bỏ các nhóm môn này khi xét mô hình.

Hầu hết các nhóm môn đều có số lượng trường hợp ngoại lệ khá nhiều ở cả 2 nhóm HDI và THO, thể hiện một số trường hợp sinh viên cá biệt với điểm số cao hoặc thấp bất thường. Đối với riêng nhóm THO có thể được giải thích bởi việc sinh viên chỉ học tốt các môn này hoặc đặc biệt không tốt các môn này trước khi bỏ học.

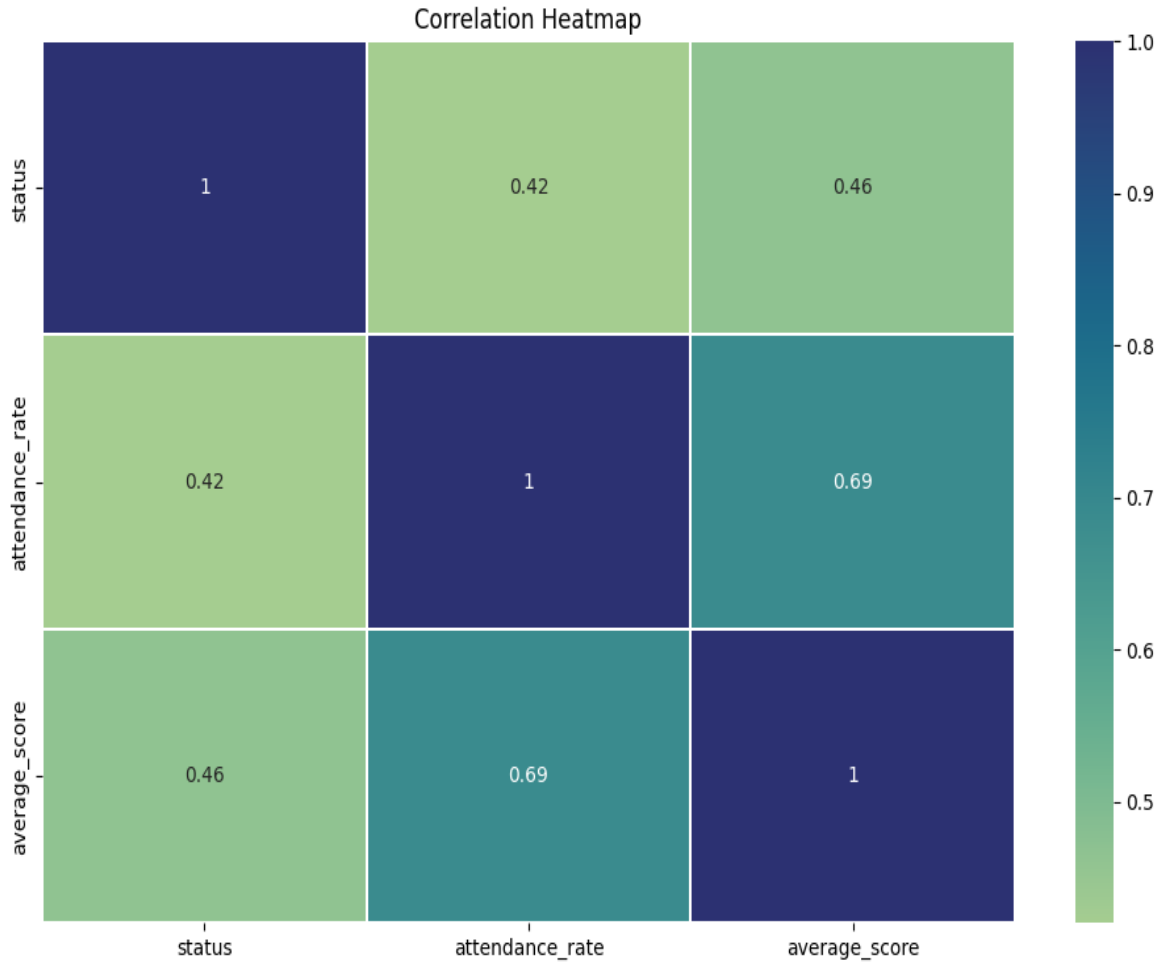
Chốt lại, sự khác biệt giữa 2 nhóm trạng thái với các nhóm môn được liệt kê có thể được cân nhắc lựa chọn sử dụng cho việc xây dựng mô hình dự đoán cuối cùng.

- **Ma trận tương quan**

Giữa Status và Attendance Rate (0.42): Ta có thể thấy mối tương quan dương, thể hiện rằng những sinh viên có tỉ lệ điểm danh cao hơn sẽ tiếp tục việc học của mình hơn những sinh viên có tỉ lệ điểm danh thấp. Tuy trọng số không thực sự ấn tượng nhưng số liệu tương quan này vẫn cho thấy một số thông tin hữu ích về hành vi của sinh viên

Giữa Status và Average Score (0.46): Tương tự như với mối tương quan giữa Status và Attendance Rate, ta cũng có thể thấy những sinh viên với điểm số trung bình cao hơn sẽ có ý định tiếp tục việc học của mình. Trọng số của mối tương quan này cũng không thực sự quá cao nhưng so với những yếu tố khác liên quan trực tiếp tới Status có thể coi là một mối tương quan mạnh và có thể được sử dụng cho việc dự đoán hành vi của sinh viên.

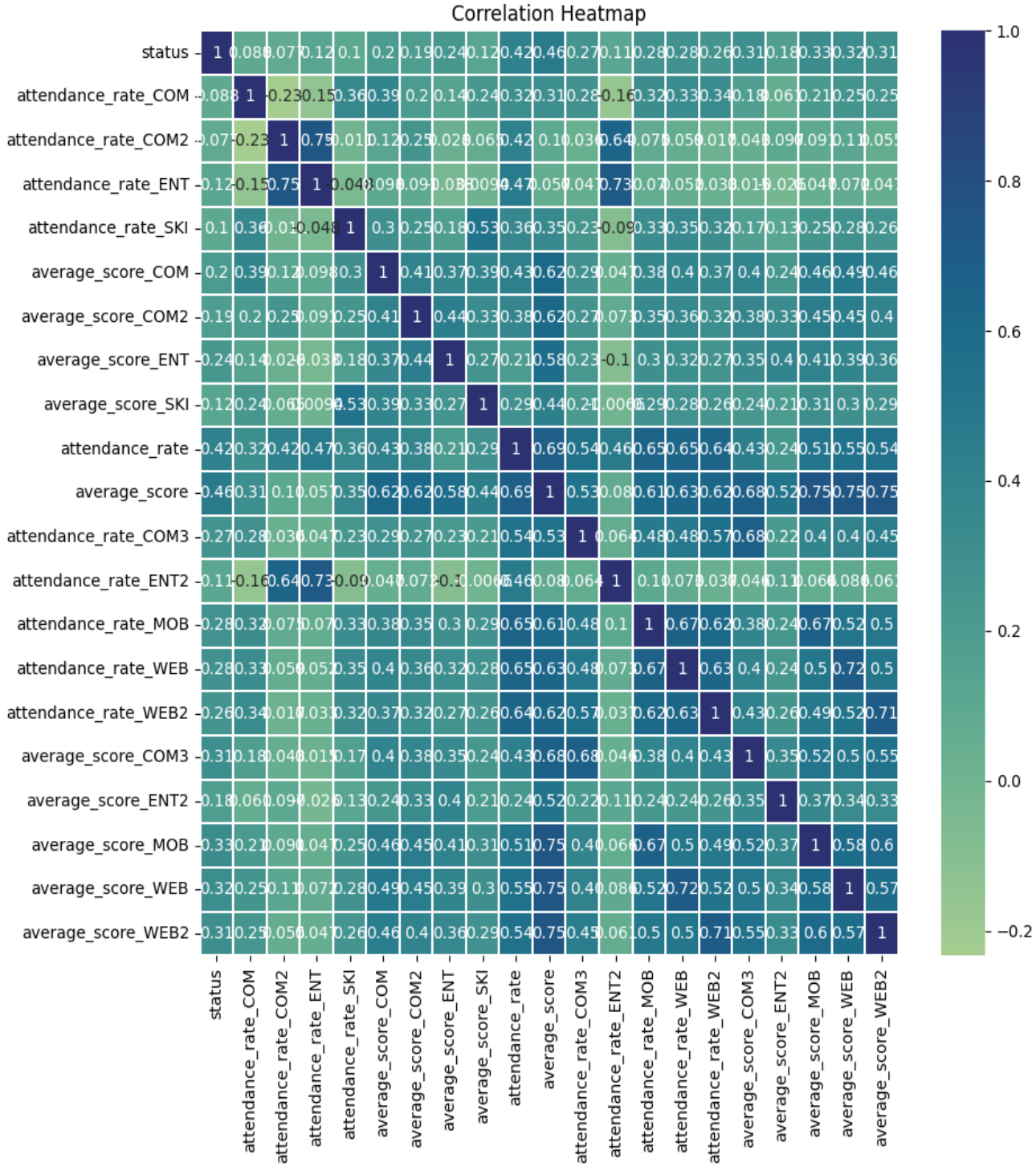
Giữa Attendance Rate và Average Score (0.69): Tuy không phải mối tương quan trực tiếp đối với Status, ta vẫn có thể đánh giá giữa tỉ lệ điểm danh và điểm trung bình có mối quan hệ tương đối mạnh. Điều này chỉ ra rằng việc điểm danh tác động khá mạnh tới kết quả đánh giá giáo dục của sinh viên.




**Hình 3.12.** Ma trận tương quan Attendance Rate và Average Score sau khi xử lý lại

- **Ma trận tương quan với từng nhóm môn**

Kết quả của ma trận tương quan đối với từng nhóm môn đều cho ra mối tương quan dương với tỉ lệ dao động chỉ từ 0.1 đến 0.4 là chủ yếu, cá biệt một số trường hợp con số này có thể lên tới khoảng 0.6 do là các môn đặc thù có ít bản ghi được ghi nhận dẫn tới tính không ổn định của mối tương quan này.




**Hình 3.13.** Ma trận tương quan đối với từng nhóm môn

Tuy nhiên, kết quả này vẫn sẽ được cân nhắc để đưa vào làm các yếu tố tác động tới mô hình và được xem xét ở phần sau của đề tài.

### 3.2.3. Tái chọn mẫu và mở rộng khác

Bên cạnh việc tái xử lý bộ dữ liệu để tìm ra bộ dữ liệu khả dụng và phù hợp với yêu cầu, ta cũng cân nhắc tới việc tái chọn mẫu (Resampling).

Ta sẽ sử dụng hàm **resample** từ thư viện sklearn.utils để giải quyết vấn đề tái chọn mẫu. Hàm **sklearn.resample** sẽ thực hiện lấy mẫu lại các mảng hoặc ma trận theo một cách nhất quán. Nó sẽ thực hiện việc tái chọn mẫu một cách ngẫu nhiên bằng cách sinh thêm hoặc lọc bớt dữ liệu ban đầu mà không thực sự thay thế bộ dữ liệu gốc. Điều này sẽ ngăn mô hình học nghiêng về phía lớp đa số để cho ra kết quả. Để sử dụng, ta sẽ thực hiện import function từ thư viện **sklearn.utils**.

Ta thu được 2 bộ dữ liệu cân bằng với tỉ lệ lý tưởng xấp xỉ 1:1 để tiếp tục thực nghiệm ở phần xây dựng mô hình tiếp theo. Trong quá trình xây dựng và kiểm thử, ta sẽ điều chỉnh thông số **samples** để cho ra các kết quả gần sát với thực tế hơn là những kết quả Overfitting/Underfitting.

Cả 2 phương pháp Undersampling/Oversampling trên đều thuộc được coi là các kỹ thuật Resampling ngẫu nhiên (Random Resampling Techniques). Các mẫu được lựa chọn thêm hoặc bớt từ tập dữ liệu một cách ngẫu nhiên và mỗi mẫu trong tập đều được coi là tương đương nhau, độc lập với nhau và có xác suất được lựa chọn giống nhau. Tuy nhiên, nhược điểm lớn nhất của kỹ thuật này là việc nó chỉ làm tăng kích thước tập huấn luyện thông qua việc lặp lại các mẫu sẵn có được quan sát trên tập thiểu số mà không làm tăng tính đa dạng của bộ dữ liệu huấn luyện. Để giải quyết vấn đề này, ta sẽ đề cập tới SMOTE – Synthetic Minority Over-sampling như một phương pháp thay thế.

SMOTE – Synthetic Minority Over-sampling là phương pháp sinh mẫu nhằm gia tăng kích thước mẫu của nhóm thiểu số trong trường hợp xảy ra mất cân bằng mẫu. Để gia tăng mẫu, với mỗi một mẫu thuộc nhóm thiểu số, ta sẽ lựa chọn ra *k* mẫu láng giềng gần nhất với nó và sau đó thực hiện tổ hợp tuyến tính để tạo ra mẫu giả lập. Ta sẽ sử dụng hàm **over_sampling.SMOTE** từ thư viện **imblearn** để áp dụng kỹ thuật này.

Ta cũng sẽ sử dụng tới Stratified K-Fold Cross-Validation thay cho K-Fold Cross-Validation. Stratified K-Fold Cross-Validation có cơ chế hoạt động gần giống như K-Fold Cross-Validation, chỉ khác ở chỗ thay vì sử dụng Random Sampling thì sẽ sử dụng Stratified Sampling nhằm đảm bảo mỗi fold quan sát đều được phân bố đầy đủ mẫu quan sát với các nhãn được đưa ra vào xét. Stratified K-Fold Cross-Validation cũng sẽ đảm bảo mỗi fold đều có thể thể hiện tính chất của tập dữ liệu mà không bị thiên kiến về bất kỳ một nhãn nào. Với điều này, các chỉ số để đánh giá như Accuracy, Precision, Recall sẽ trở nên đáng tin hơn. Để áp dụng, ta sẽ sử dụng tới hàm **StratifiedKFold** được cung cấp từ thư viện **sklearn.model_selection** với các tham số tương ứng (n_splits, shuffle, random_state).

# CHƯƠNG 4: XÂY DỰNG MÔ HÌNH DỰ ĐOÁN HÀNH VI BỎ HỌC CỦA SINH VIÊN

## 4.1 Các lựa chọn mô hình

Dưới đây là một số hàm sẽ có thể được sử dụng trong việc xây dựng:

- **sklearn.model_selection**: train_test_split, GridSearchCV, StratifiedKFold.
- **sklearn.linear_model**: LogisticRegression.
- **sklearn.ensemble**: RandomForestClassifier, VotingClassifier.
- **sklearn.preprocessing**: StandardScaler, MinMaxScaler.
- **sklearn.metrics**: accuracy_score, precision_score, recall_score, f1_score, confusion_matrix, roc_curve, auc, classification_report, log_loss.
- **sklearn.utils**: resample.
- **imblearn.over_sampling**: SMOTE.
- **imblearn.pipeline**: Pipeline.

### 4.1.1. Hồi quy Logistic - Logistic Regression

Ta sẽ áp dụng mô hình hồi quy Logistic với thư viện scikit-learn. Trong thư viện scikit-learn hỗ trợ cho Python đã có sẵn lớp LogisticRegression để giúp chúng ta giải quyết bài toán liên quan một cách nhanh chóng và hiệu quả hơn.

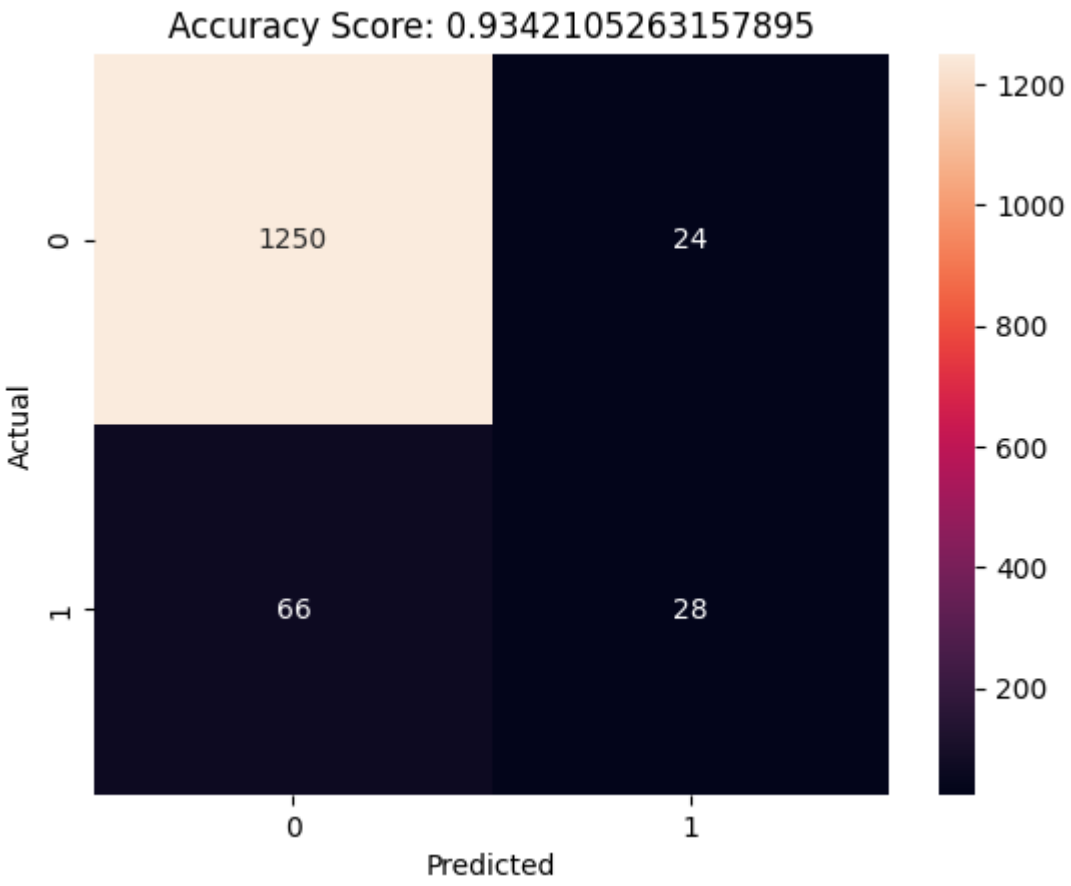
Trước hết, ta sẽ xem xét với bộ dữ liệu khi chưa được giải quyết mất cân bằng dữ liệu với 2 đặc trưng chính được xem xét là Attendance Rate (tỉ lệ điểm danh) và Average Score (điểm trung bình chung). Ta tiến hành việc chia dữ liệu thành 2 tập: tập huấn luyện và tập kiểm tra với tỉ lệ 80/20 bằng cách sử dụng **train_test_split**

Tạo 1 instance LogisticRegression và tiến hành truyền dữ liệu đầu vào và tiến hành training model. Ta thu được kết quả:

```
[ ]: 0.9342105263157895
```

Ta thực hiện testing model và đánh giá mức độ hiệu quả:

```
[ ]: Accuracy: 0.9342105263157895
     Precision:  0.5384615384615384
     Recall:  0.2978723404255319
     F1:  0.3835616438356164
```




**Hình 4.1.** Confusion Matrix đối với Logistic Regression

Quan sát kết quả, ta rút ra được một số nhận xét như sau:

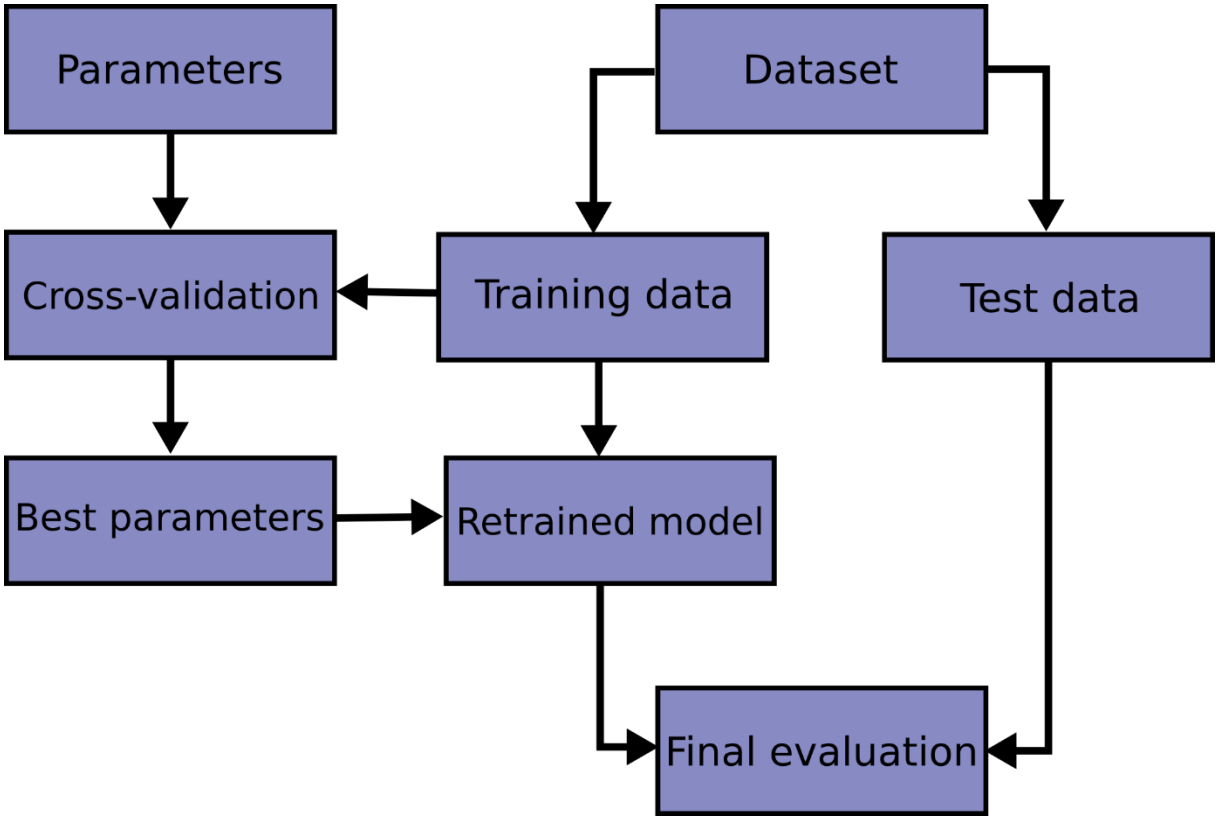
- Chỉ số Accuracy cho ra kết quả khá cao là khoảng 93.42%. Tuy nhiên, khi xét tới phân loại của từng dữ liệu, phần lớn những giá trị dự đoán cho nhóm 1 (nhóm THO) đều cho ra kết quả sai so với kỳ vọng và số kết quả dự đoán cho nhóm 0 (nhóm HDI) chiếm số lượng lớn.
- Với chỉ số Precision, mô hình chỉ cho ra kết quả trên trung bình khoảng 53.8% cho thấy trong tổng số những dự đoán True, chỉ có khoảng một nửa số dự đoán là thực sự đúng.
- Chỉ số Recall chỉ đạt khoảng 29.7% cho thấy mô hình bỏ sót khá nhiều dự đoán Positive khi mà True Positive Rate khá thấp.
- Cuối cùng, chỉ số F1 là sự kết hợp đồng thời giữa Precision và Recall chỉ đạt khoảng 38.35% cho thấy mô hình không thực sự hoạt động tốt khi xem xét cả 2 chỉ số này cùng nhau.

Tất cả những điều này đã phản ảnh đúng vấn đề mà bộ dữ liệu gặp phải là việc mất cân bằng dữ liệu nghiêm trọng. Ta sẽ tiến hành một số phương pháp đã được đề cập ở chương 3 để cải thiện kết quả trên.

Tiếp theo, ta sẽ thử tiến hành cân bằng dữ liệu bằng cách Resampling và sau đó thực hiện Tuning mô hình. Trước hết ta sẽ chia lại bộ dữ liệu để phù hợp với việc Tuning sau này như sau:

- Chia bộ dữ liệu thành *training* và *testing* với tỉ lệ 80/20
- Tiếp tục chia bộ dữ liệu *training* thành hai bộ *training* và *validation* với tỉ lệ 80/20

Khi này, quy trình xây dựng mô hình sẽ được thêm vào bước validating và tuning để tìm ra parameters tốt nhất rồi mới cho ra kết quả cuối cùng.




**Hình 4.2.** Quy trình xây dựng mô hình sau khi được chỉnh sửa

Ta sẽ xây dựng Pipeline để tái chọn mẫu và đưa vào mô hình hồi quy bằng các hàm **SMOTE**, **Pipeline**, **StratifiedKFold**.

Tiếp theo, ta sẽ tối ưu *hyperparameter* (siêu tham số) của mô hình. Trong học máy, siêu tham số là các tham số được thiết lập trước khi quá trình học bắt đầu. Các tham số này điều khiển chính quá trình học và ảnh hưởng đến hiệu suất của mô hình học máy đang được huấn luyện. Không giống như các tham số của mô hình, được học trong quá trình huấn luyện, các siêu tham số thường được thiết lập thủ công trước khi bắt đầu huấn luyện. Để tối ưu, ta sẽ sử dụng tới **GridSearchCV**. Phương pháp này được coi là một phương pháp "vét cạn" để tối ưu hóa siêu tham số. Ta sẽ cài đặt mô hình bằng tất cả các cách kết hợp có thể sau khi tạo ra một lưới các giá trị siêu tham số rời rạc có tiềm năng. Ta ghi lại hiệu suất của mỗi bộ tham số và sau đó chọn ra kết hợp cho ra kết quả tốt nhất.

Tuy nhiên, phương pháp này cũng có nhược điểm là việc tiêu tốn nhiều tài nguyên và thời gian. Ta thường sẽ mất nhiều nguồn tài nguyên để xử lý và thời gian để cài đặt mô hình với mọi kết hợp tiềm năng. Trong phạm vi nghiên cứu này, ta sẽ giới hạn điều chỉnh hyperparameter C để kiểm soát hiệu suất của mô hình trên dữ liệu kiểm tra.

Ta tiến hành cài đặt với hàm **GridSearchCV** từ **sklearn.model_selection** với đầu vào là pipeline được xây dựng ở phần trước. Đặt tiêu chí đánh giá theo chỉ số *f1-score*, ta tiến hành tuning trên tập dữ liệu training để tìm ra hyperparameter tốt nhất. Ta thu được kết quả:

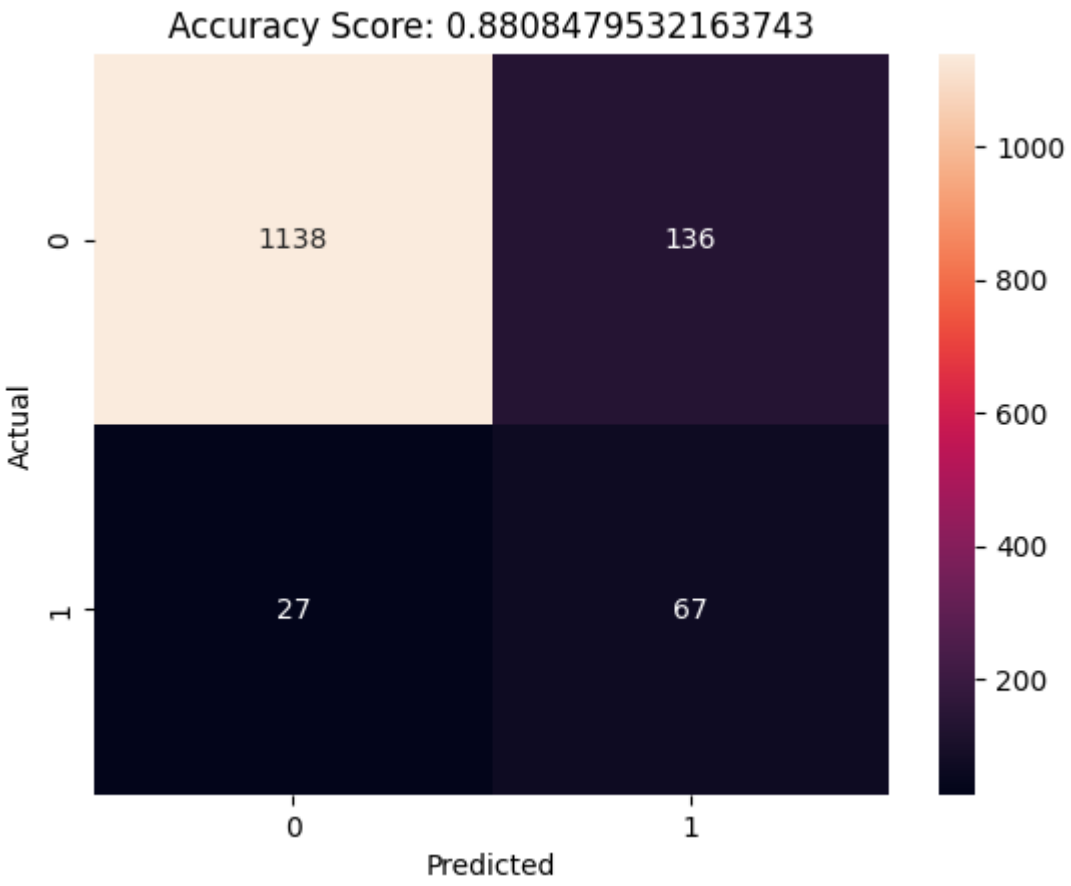
```
[ ]: Best Hyperparameters: {'lr__C': 0.01}
```

Với thông số thu được, ta tiến hành đánh giá trên tập dữ liệu validation:

```
[ ]: Validation Set Evaluation
[[896 111]
 [ 24  64]]
              precision    recall  f1-score   support

           0       0.97      0.89      0.93      1007
           1       0.37      0.73      0.49        88

    accuracy                           0.88      1095
   macro avg       0.67      0.81      0.71      1095
weighted avg       0.93      0.88      0.89      1095
```

Ta có thể thấy hầu hết các chỉ số đều cho ra kết quả được cải thiện so với mô hình gốc. Ta sẽ tiến hành đánh giá cuối cùng trên tập dữ liệu testing. Kết quả thu được không chênh lệch quá nhiều so với trên tập dữ liệu validation

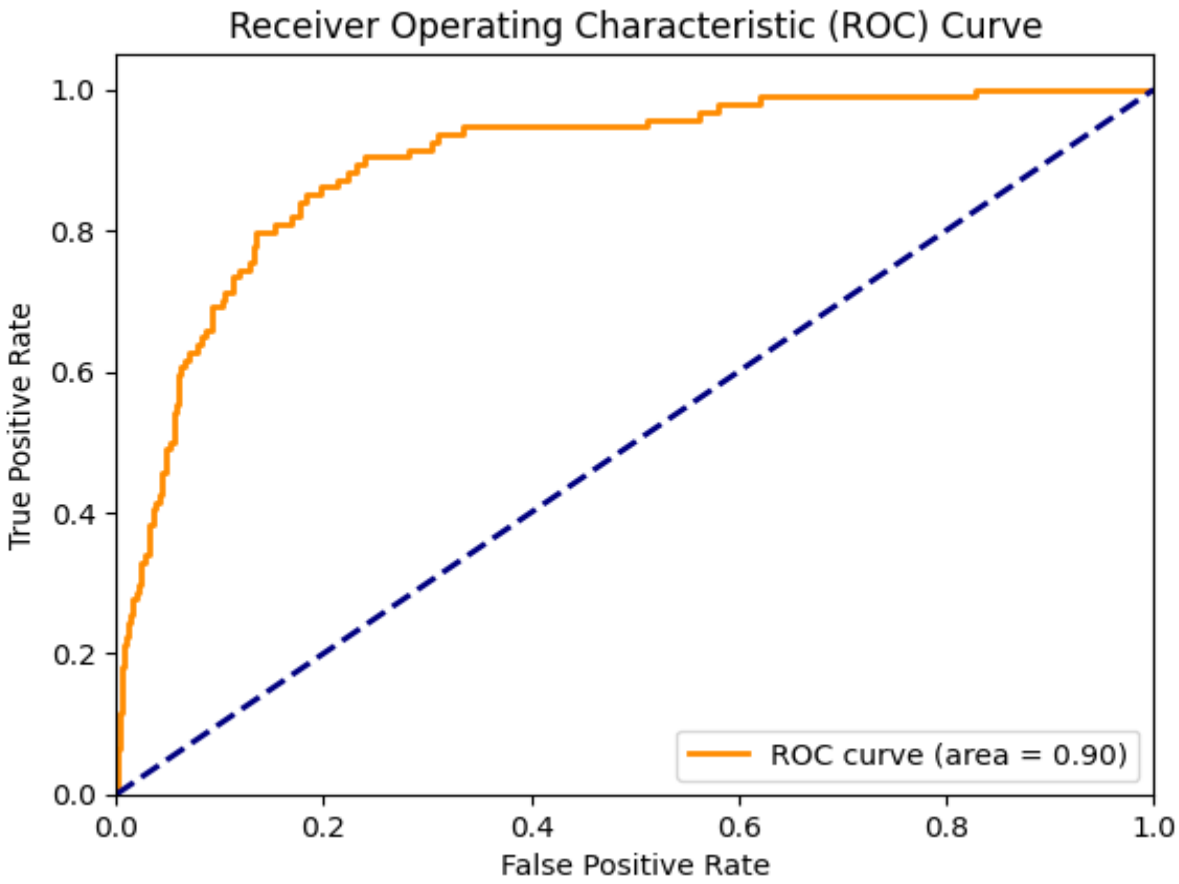
```
[ ]: Accuracy: 0.8808479532163743
Precision:  0.33004926108374383
Recall:  0.7127659574468085
F1:  0.4511784511784511
```




**Hình 4.3.** Confusion Matrix đối với Logistic Regression sau khi Tuning

Quan sát biểu đồ Heatmap được xây dựng từ kết quả dự đoán, ta có thể thấy tổng số lượng các trường hợp True Positive và False Positive cao hơn nhiều so với số các trường hợp False Negative. Giới hạn trong nghiên cứu này, ta sẽ chấp nhận tỉ lệ báo động giả (False Alarm Rate) ở mức trung bình. Nguyên do là bởi, mô hình chưa thể phát hiện được đủ số lượng nhãn cần thiết để đối phó với việc mất cân bằng dữ liệu một cách triệt để. Ta sẽ công nhận các kết quả dự đoán sinh viên sẽ bỏ học nhưng thực tế là tiếp tục học để có thể bao phủ hầu hết các trường hợp thực tế, tránh số lượng những sinh viên bỏ học thực sự nhưng mô hình dự đoán sai và không bắt được những trường hợp này.

Từ đồ thị ROC, ta có thể thấy chỉ số AUC ở mức khá cao (0.9) cho thấy hiệu suất phân loại của mô hình ở mức ổn, mô hình phân loại khá chính xác. Tuy nhiên, ta cũng không nên quá tin tưởng và lạc quan vào kết quả này mà không có những đánh giá sâu hơn do mô hình đã được xử lý và hiệu chỉnh chưa thực sự chi tiết.




**Hình 4.4.** Đồ thị ROC của Logistic Regression sau khi Tuning

### 4.1.2. Rừng ngẫu nhiên – Random Forest Classifier

Ta sử dụng tập dữ liệu được chia ở phần trước để tiếp tục thử nghiệm với mô hình rừng ngẫu nhiên. Trong thư viện scikit-learn hỗ trợ cho Python cũng có sẵn lớp RandomForestClassifier để giúp chúng ta giải quyết bài toán liên quan một cách nhanh chóng và hiệu quả hơn.

Tạo 1 instance RandomForestClassifier và xây dựng Pipeline để tái chọn mẫu và đưa vào mô hình hồi quy. Ta cũng cài đặt GridSearchCV để hiệu chỉnh hyperparameter của mô hình:

Trong đó, các siêu tham số được điều chỉnh có ý nghĩa:

- n_estimators: số cây thuật toán xây dựng trước khi lấy phiếu bầu tối đa hoặc giá trị trung bình các dự đoán. Số lượng cây cao hơn làm tăng hiệu suất và làm cho các dự đoán ổn định hơn, nhưng nó cũng làm chậm quá trình tính toán.
- max_depth: độ sâu của mỗi cây trong rừng ngẫu nhiên. Nếu giá trị này được set là None thì tại mỗi nodes sẽ mở rộng tới khi kết quả trả về tại nodes lá chỉ thuộc về một lớp duy nhất (purity) hoặc mỗi lá chứa ít hơn min_samples_split samples.

- min_samples_split: số samples tối thiểu ở mỗi lần split. Giá trị này càng cao thì càng hữu ích khi ngăn việc overfitting.
- min_samples_leaf: số samples tối thiểu yêu cầu ở mỗi node lá. Giá trị này càng cao thì model càng tránh việc bị quá nhạy cảm đối với các giá trị ngoại lệ.

Đặt tiêu chí đánh giá theo chỉ số *f1-score*, ta tiến hành tuning trên tập dữ liệu training để tìm ra hyperparameter tốt nhất:

```
[ ]: Best Hyperparameters for Random Forest:
{'rf__max_depth': 10, 'rf__min_samples_leaf': 1, 'rf__min_samples_split': 5, 'rf__n_estimators': 50}
```

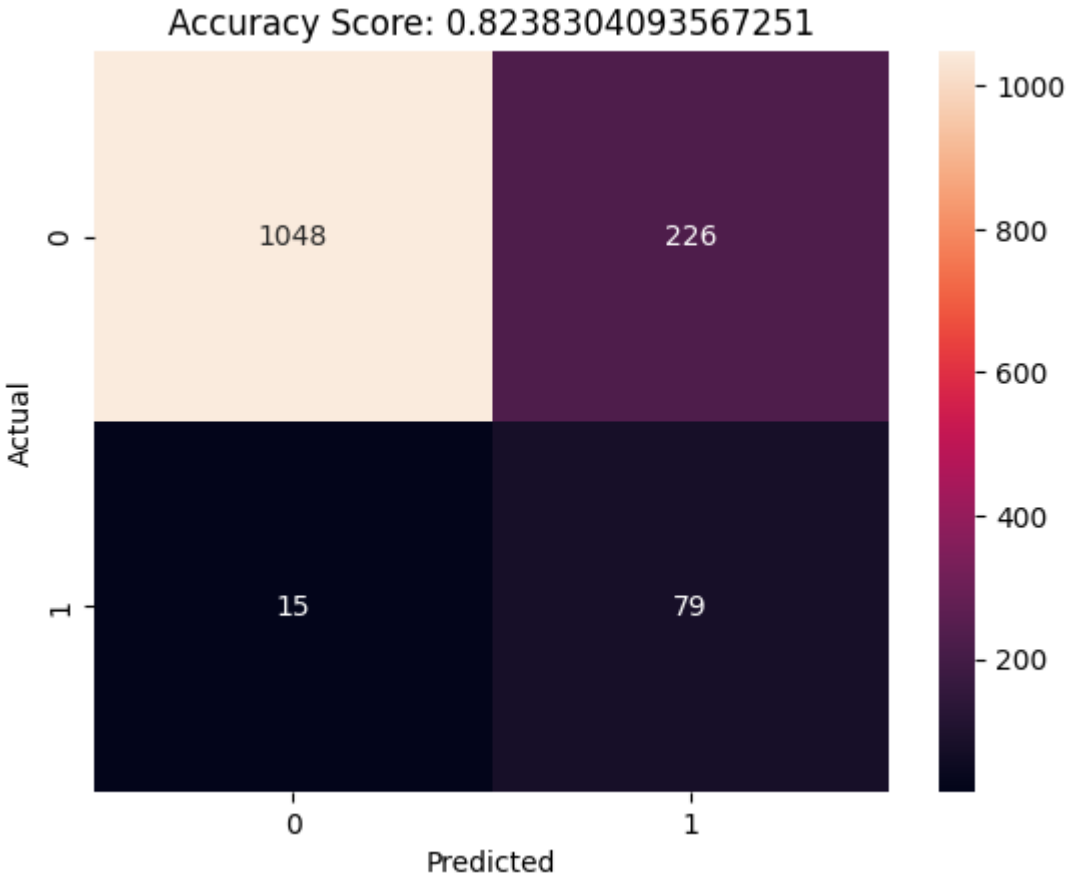
Với thông số thu được, ta tiến hành đánh giá trên tập dữ liệu validation. Ta có thể thấy các chỉ số khá tương quan với chỉ số cho ra từ mô hình Logistic Regression.

```
[ ]: Validation Set Evaluation
[[856 151]
 [ 22  66]]
              precision    recall  f1-score   support

           0       0.97      0.85      0.91      1007
           1       0.30      0.75      0.43        88

    accuracy                           0.84      1095
   macro avg       0.64      0.80      0.67      1095
weighted avg       0.92      0.84      0.87      1095
```

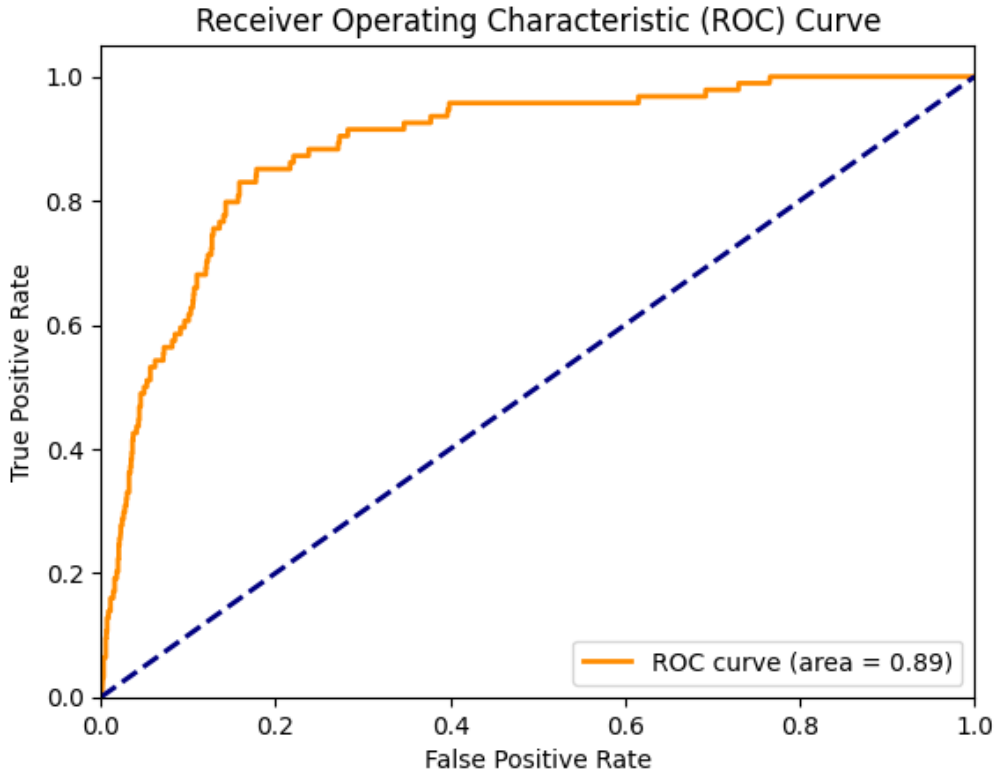
Ta sẽ tiến hành đánh giá cuối cùng trên tập dữ liệu testing:

```
[ ]: Accuracy: 0.8238304093567251
Precision:  0.25901639344262295
Recall:  0.8404255319148937
F1:  0.39598997493734345
```

Quan sát biểu đồ Heatmap, ta có thể thấy kết quả không chênh lệch nhiều so với những thông số mà ta đã rút ra được từ mô hình Logistic Regression ở trên. Vậy nên ta có thể tạm chấp nhận kết quả thu được để có những cải thiện trong tương lai.




**Hình 4.5.** Confusion Matrix đối với Random Forest Classifier sau khi Tuning




**Hình 4.6.** Đồ thị ROC của Random Forest Classifier sau khi Tuning

Từ đồ thị ROC, ta có thể thấy chỉ số AUC ở mức khá cao (0.89) cho thấy hiệu suất phân loại của mô hình ở mức ổn, mô hình phân loại khá chính xác. Tuy nhiên, ta cũng không nên quá tin tưởng và lạc quan vào kết quả này mà không có những đánh giá sâu hơn do mô hình đã được xử lý và hiệu chỉnh chưa thực sự chi tiết.

### 4.1.3. Phân loại Voting – Voting Classifier

Ta sẽ sử dụng hai model đã được trình bày là Logistic Regression và Random Forest Classifier để làm đầu vào cho mô hình Voting Classifier. Trong thư viện scikit-learn hỗ trợ cho Python cũng có sẵn lớp VotingClassifier để giúp chúng ta giải quyết bài toán liên quan một cách nhanh chóng và hiệu quả hơn.

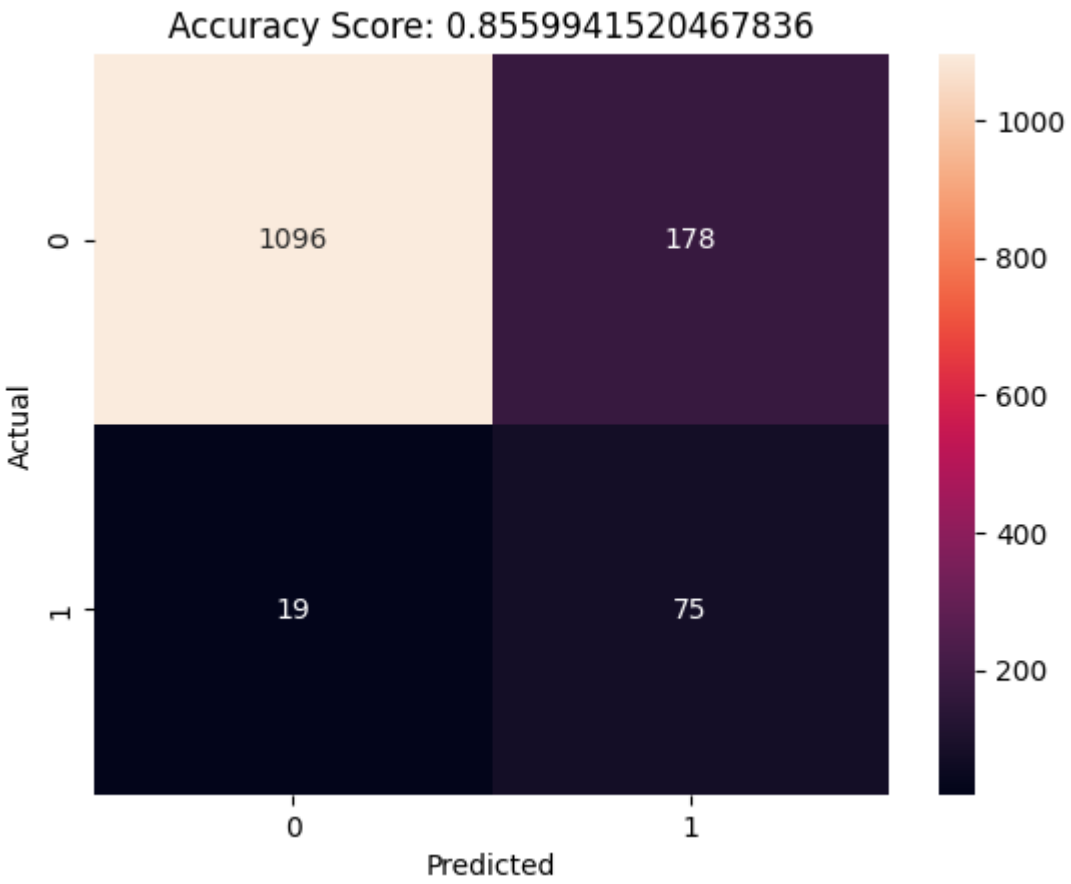
Tạo 1 instance VotingClassifier với tham số đầu vào là hai model thu được ở trên. Ta thực hiện đánh giá mô hình trên tập dữ liệu validation:

```
[ ]: Validation Set Evaluation with Voting Classifier
[[875 132]
 [ 20  68]]
              precision    recall  f1-score   support

           0       0.98      0.87      0.92      1007
           1       0.34      0.77      0.47        88

    accuracy                           0.86      1095
   macro avg       0.66      0.82      0.70      1095
weighted avg       0.93      0.86      0.88      1095
```

Đánh giá mô hình trên tập dữ liệu testing thu được kết quả:

```
[ ]: Test Set Evaluation with Voting Classifier
Accuracy: 0.8559941520467836
Precision:  0.2964426877470356
Recall:  0.7978723404255319
F1:  0.4322766570605187
```

Biểu đồ Heatmap của mô hình dự đoán:

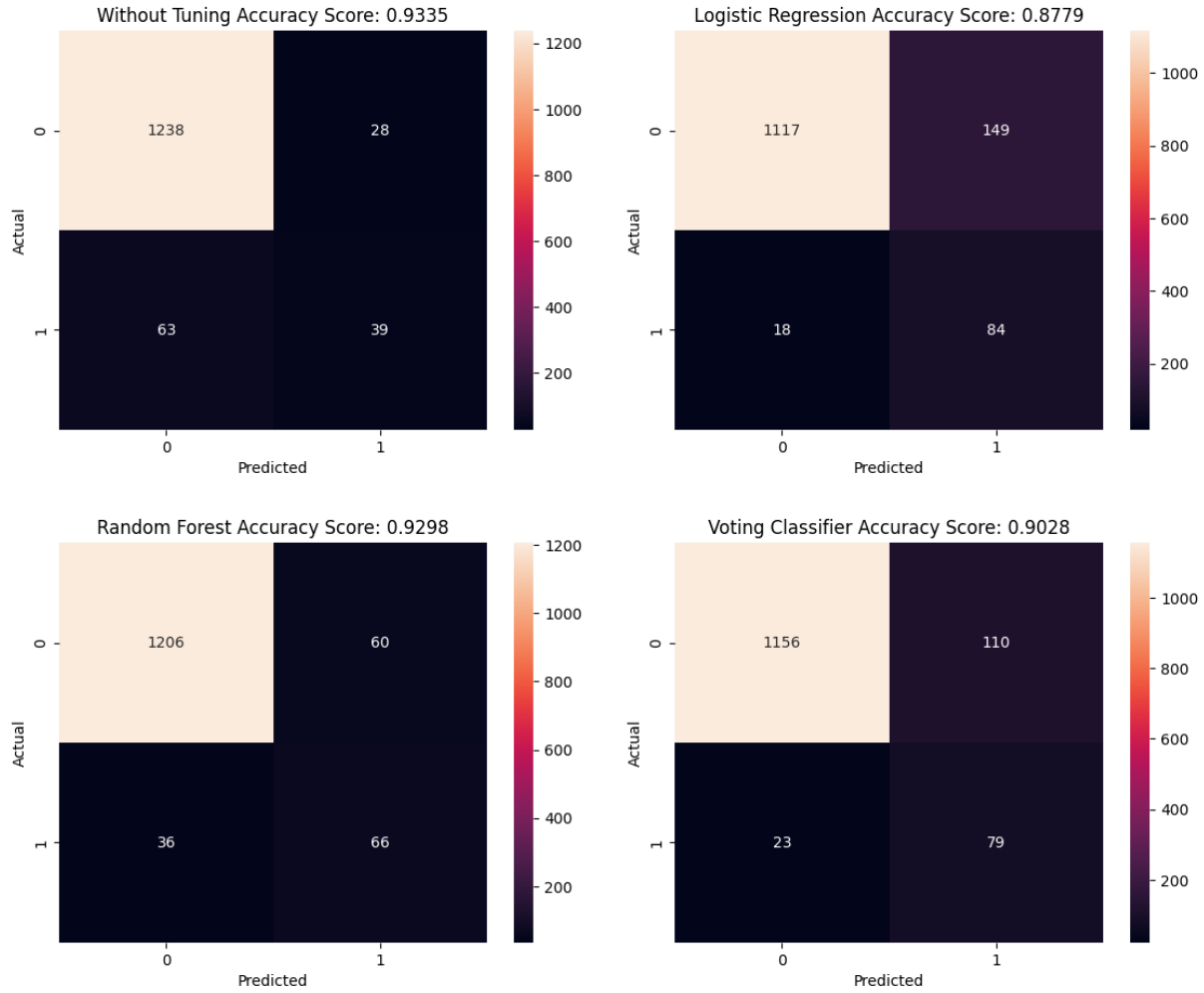



**Hình 4.7.** Confusion Matrix đối với Voting Classifier sau khi Tuning

Ta có thể thấy các kết quả thu được khá ít biến động so với những mô hình được đề cập trước đó. Kết quả thu được dừng ở mức có thể chấp nhận được và cần có sự chỉnh sửa và hoàn thiện để đầu ra được cải thiện hơn.

### 4.1.4. Xây dựng mô hình với những đặc trưng khác

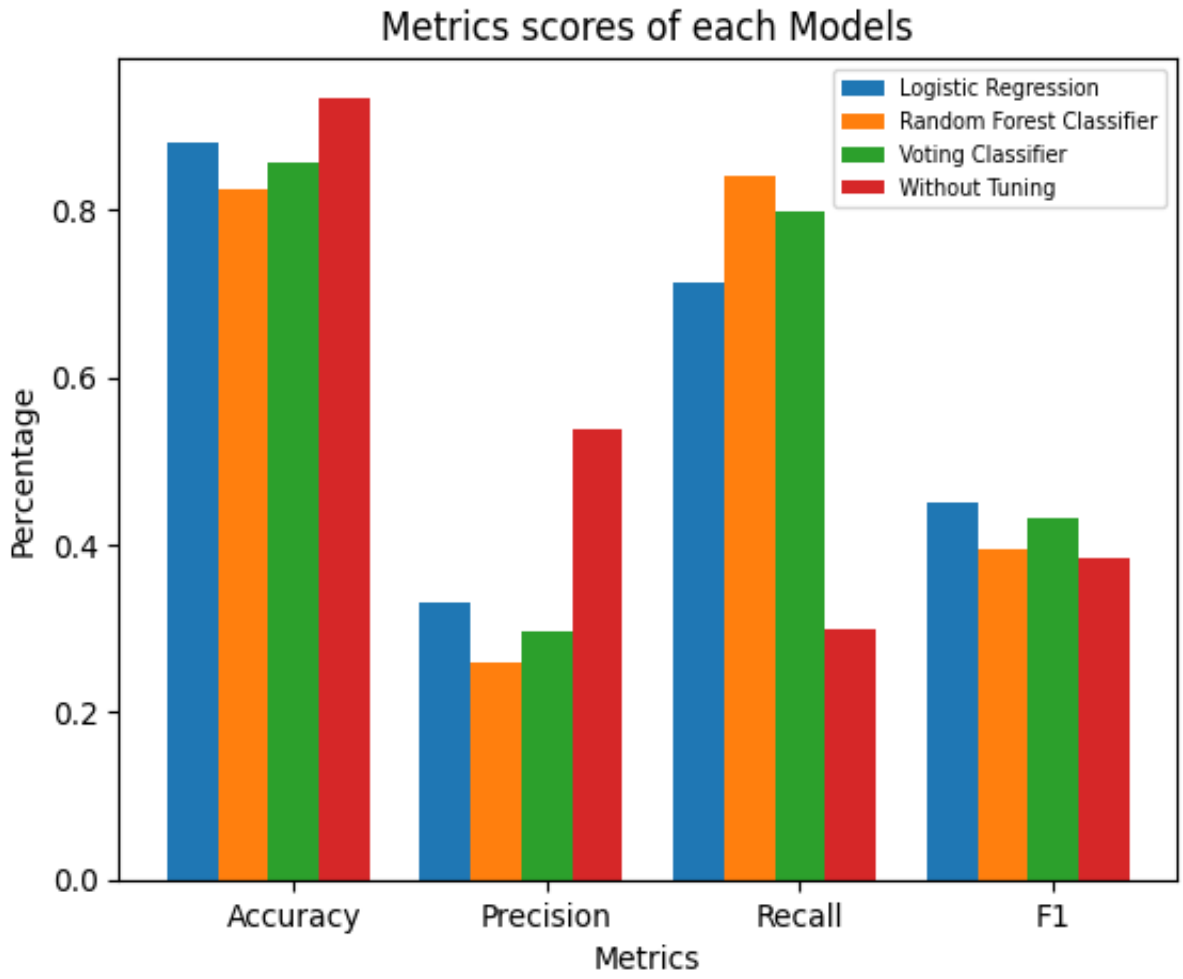
Ngoài bộ dữ liệu với hai đặc trưng chính là Average Score và Attendance Rate được sử dụng để xét trong các mô hình trên, ta cũng sẽ mở rộng thêm với bộ dữ liệu với các đặc trưng khác là thông tin đối với từng môn học. Đối với những dữ liệu này, một số môn học có số lượng sinh viên theo học chiếm số lượng rất nhỏ sẽ bị loại bỏ và một số sinh viên thiếu điểm môn học (Null hoặc NaN) sẽ được xử lý bằng cách điền bằng giá trị trung bình của cột được xét. Áp dụng tương tự các bước xây dựng mô hình như trên, ta thu được kết quả được trình bày dưới đây:




**Hình 4.8.** Đánh giá Confusion Matrix với các mô hình được thêm đặc trưng khác

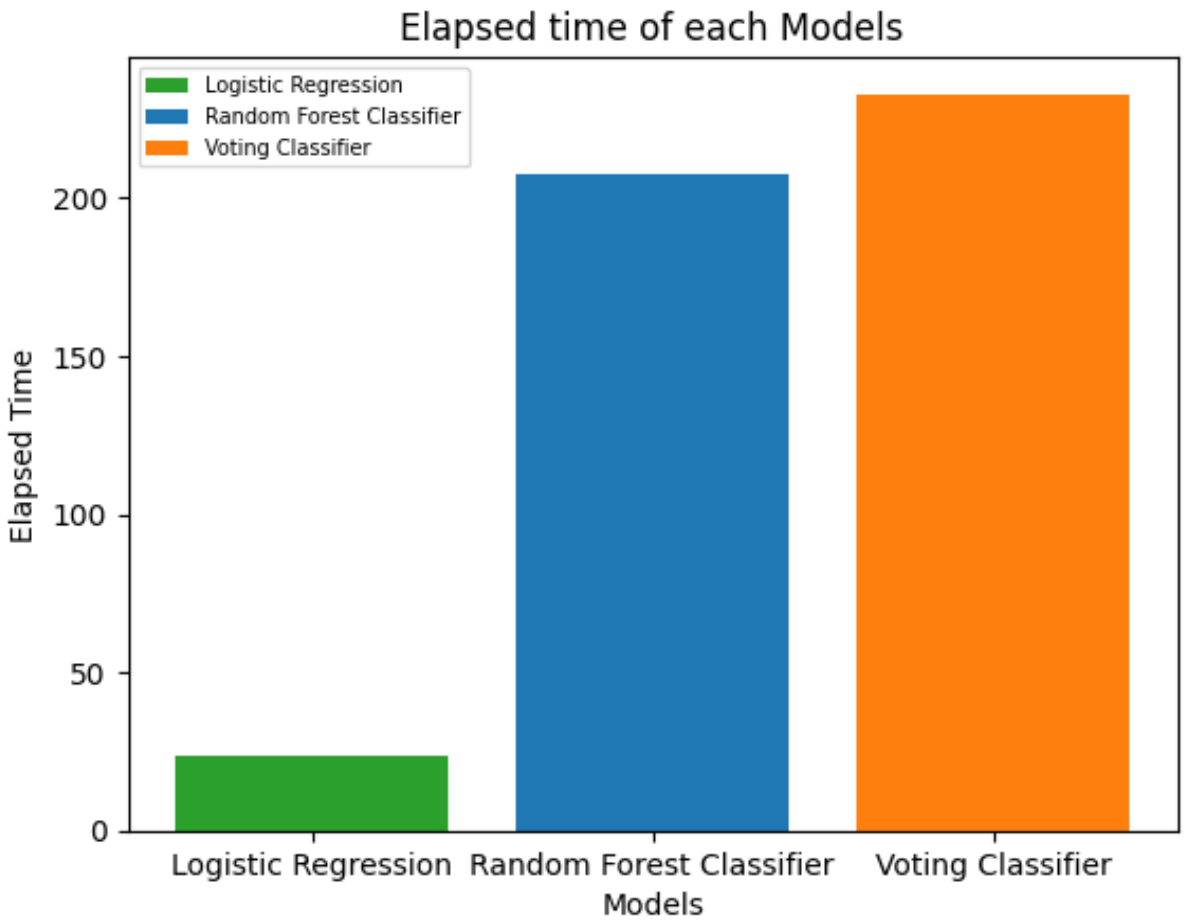
Có thể thấy, sự chênh lệch và tương quan tỉ lệ giữa trước và sau khi thêm vào các đặc trưng mới là chưa thực sự đáng kể, đặt ra yêu cầu về việc tối ưu lại chọn lọc các đặc trưng để đưa vào mô hình.

## 4.2. So sánh các mô hình




**Hình 4.9.** Biểu đồ so sánh chỉ số của các mô hình được xây dựng

Với mục tiêu đặt ra là dự đoán hành vi bỏ học của sinh viên, ta chấp nhận tỉ lệ báo động nhầm (False Alarm Rate) cao hơn tỉ lệ bỏ sót (Failure Detection Rate) để đảm bảo bao phủ được hầu hết các trường hợp. Điều này đồng nghĩa với việc ta sẽ cần duy trì Recall ở mức cao. Dựa vào biểu đồ thống kê, ta có thể thấy rằng, sau khi hiệu chỉnh mô hình thì cả ba mô hình đều cho ra kết quả Accuracy thấp hơn so với mô hình khi chưa thực hiện hiệu chỉnh. Đổi lại, các chỉ số Recall và F1 lại được cải thiện. Trong đó, Random Forest Classifier cho ra tỉ lệ Recall cao nhất (khoảng 84.04%) nhưng hai chỉ số còn lại là Precision và F1 thì lại là thấp nhất trong ba mô hình đồng nghĩa với việc mô hình này có thể không phải lựa chọn tối ưu nhất. Voting Classifier và Logistic Regression cho ra kết quả khá gần nhau về mặt dự đoán nhưng ta sẽ cần quan tâm tới cả hiệu suất và thời gian chạy của mô hình.




**Hình 4.10.** Biểu đồ so sánh thời gian thực thi của các mô hình được xây dựng

Biểu đồ cho thấy, thời gian để mô hình Logistic Regression chạy chỉ bằng 1/10 so với Random Forest Classifier. Do Voting Classifier là mô hình tổng hợp từ các mô hình con để đưa ra dự đoán cuối cùng nên thời gian chạy của mô hình phụ thuộc hoàn toàn vào tổng thời gian chạy của các mô hình con. Điều này cho thấy rằng Logistic Regression trở thành một trong những lựa chọn tốt nhất khi cân nhắc tới hiệu suất và thời gian tính toán.

Tổng kết lại, dù Random Forest Classifier có thể bao phủ hầu hết các trường hợp có khả năng bỏ học, Logistic Regression lại thể hiện được rằng là một lựa chọn hợp lý hơn khi xét tới các yếu tố hiệu suất và thời gian, đặc biệt là khi bộ dữ liệu tăng trưởng và trở nên phức tạp hơn. Voting Classifier có thể được xem xét nếu hiệu suất tổng thể của nó vượt trội và thời gian tính toán không phải là vấn đề lớn. Tuy nhiên, nếu tốc độ là ưu tiên thì Logistic Regression vẫn sẽ là lựa chọn ưu việt hơn.

# CHƯƠNG 5: KẾT LUẬN VÀ KIẾN NGHỊ

## 5.1. Kết quả đạt được

- Đề tài đã bước đầu thành công xây dựng được một số mô hình dự đoán hành vi nghỉ học của sinh viên.
- Áp dụng được các kỹ thuật nghiên cứu, xử lý, phân tích và đánh giá dữ liệu.
- Có thêm các kiến thức về nghiên cứu Machine Learning và các kỹ thuật liên quan của Data Science.

## 5.2. Những vấn đề còn tồn tại

- Các mô hình xây dựng được chưa thực sự tối ưu và mới chỉ áp dụng trên một bộ dữ liệu nhỏ được giới hạn cả về thời gian lấy mẫu và kích thước mẫu.
- Chưa phát hiện được thêm các đặc trưng trọng yếu để nâng cao hiệu suất của mô hình.

## 5.3. Hướng phát triển tiếp của đề tài

- Tiếp tục thu thập thêm bộ dữ liệu thống kê từ đơn vị trường để nghiên cứu và tìm ra các đặc trưng mới phù hợp hơn.
- Cải thiện các nhóm đặc trưng hiện có của mô hình.
- Nghiên cứu các cách hiệu chỉnh mô hình khác.

## 5.4. Những thuận lợi và khó khăn trong quá trình làm đề tài

- Về thuận lợi: Trong quá trình làm đồ án, dưới sự hướng dẫn của giảng viên hướng dẫn là TS. Vũ Huấn, em đã có thêm nhiều hướng tiếp cận và cách giải quyết bài toán đề tài đặt ra. Ngoài ra, em cũng đã nhận được sự giúp đỡ nhiệt tình từ phía đơn vị Trường Cao đẳng FPT Polytechnic đã cho phép em được sử dụng cơ sở dữ liệu mẫu của nhà trường và phòng IT đã giúp đỡ em nghiên cứu, phân tích cơ sở dữ liệu trên. Các nguồn tài liệu về Machine Learning và các vấn đề liên quan khá phong phú và đầy đủ cũng đã một phần giúp em thuận lợi hoàn thành đề tài này.
- Về khó khăn: Trong quá trình làm đồ án, khó khăn lớn nhất là việc em chưa có nhiều kinh nghiệm và các kiến thức liên quan tới lĩnh vực này nên đã mất khá nhiều thời gian để nghiên cứu và giải quyết các vấn đề phát sinh.

# DANH MỤC TÀI LIỆU THAM KHẢO


1. Amazon Web Services Developer Tools Documentation – What is Python? (https://aws.amazon.com/what-is/python/)
2. Python 3.12.3 Documentation (https://docs.python.org/3/)
3. Ramesh Sharda, Dursun Delen, Efraim Turban – ANALYTICS, DATA SCIENCE, & ARTIFICIAL INTELLIGENCE (SYSTEMS FOR DECISION SUPPORT) (11th Edition)
4. Basic Concepts in Machine Learning (https://www.javatpoint.com/basic-concepts-in-machine-learning/)
5. Google Foundational courses for Developers – Imbalanced Data (https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data/)
6. Haibo He, Yunqian Ma – Imbalanced Learning – Foundations, Algorithms and Applications (2013)
7. ThS. Lê Song Toàn, PGS.TS. Nguyễn Thanh Bình – Xử lý dữ liệu không cân bằng trong bài toán dự đoán lỗi phần mềm – KỶ YẾU HỘI THẢO KHOA HỌC QUỐC GIA CITA 2020 "CNTT VÀ ỨNG DỤNG TRONG CÁC LĨNH VỰC"
8. Scikit-learn Official Documentation (https://scikit-learn.org/stable/user_guide.html/)
9. Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer – SMOTE: Synthetic Minority Over-sampling Technique – Journal of Artificial Intelligence Research 16 (2002) 321–357
10. Phạm Đình Khánh – Machine Learning Algorithms to Practice – EBook
11. Phạm Thị Phương Trang – Ứng dụng phương pháp bình chọn các mô hình trí tuệ nhân tạo để phân loại hai lớp và đa lớp trong xây dựng – KỶ YẾU HỘI THẢO KHOA HỌC QUỐC GIA CITA 2020 "CNTT VÀ ỨNG DỤNG TRONG CÁC LĨNH VỰC"