ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA HỆ THỐNG THÔNG TIN

BÁO CÁO ĐỒ ÁN

MẠNG XÃ HỘI

ĐỀ TÀI:

Dự đoán kết quả hoàn thành khoá học của học viên trên bộ dữ liệu MOOCCubeX

GVHD: ThS. Nguyễn Thị Anh Thư

Nhóm thực hiện: 7

Sinh viên thực hiện:

Lê Thị Ánh Hồng	**-**	**21520245**
Trần Phước Chung	**-**	**21521893**
Nguyễn Duy Đạt	**-**	**21521936**
Lê Anh Duy	**-**	**21521994**
Lê Ngọc Yến Khoa	**-**	**21522224**
Nguyễn Thị Mai Liên	**-**	**21522283**
Trần Thị Mỹ Xoan	**-**	**21522815**

TP. Hồ Chí Minh, ngày 11 tháng 12 năm 2024

Mục lục

CHƯƠNG 1: GIỚI THIỆU BÀI TOÁN.

1.1. Giới thiệu về bài toán :

Sự phát triển của công nghệ thông tin và truyền thông đã dẫn đến sự bùng nổ của các khóa học trực tuyến mở rộng (MOOC - Massive Open Online Courses), cho phép học viên trên toàn thế giới tiếp cận giáo dục chất lượng cao mà không bị giới hạn bởi địa lý hay chi phí. Nhiều nền tảng học tập như Coursera, edX, Udacity đã mở ra các khóa học miễn phí hoặc có phí thấp, tạo ra một cuộc cách mạng trong cách thức truyền tải tri thức.

Tuy nhiên, một trong những thách thức lớn nhất mà các khóa học MOOC đang đối mặt là tỷ lệ hoàn thành khóa học thấp. Theo các nghiên cứu, nhiều học viên đăng ký khóa học nhưng không duy trì được động lực học tập đến khi kết thúc. Tỷ lệ hoàn thành khóa học thường chỉ khoảng 5-15%, gây lãng phí tài nguyên và làm giảm hiệu quả của phương thức học trực tuyến. Trong bối cảnh này, việc xây dựng các mô hình dự đoán khả năng hoàn thành khóa học của học viên trở nên quan trọng, giúp các nền tảng học tập có thể can thiệp kịp thời để hỗ trợ học viên.

Bộ dữ liệu MOOCCubeX là một tập dữ liệu lớn và phong phú, chứa thông tin về các khóa học MOOC, hành vi học tập của học viên và kết quả học tập. Dựa trên dữ liệu này, việc xây dựng mô hình dự đoán kết quả hoàn thành khóa học có thể giúp xác định những yếu tố ảnh hưởng lớn đến khả năng hoàn thành khóa học, từ đó hỗ trợ việc phát triển các chiến lược cải thiện hiệu suất học tập của học viên.

1.2. Loại bài toán :

Bài toán học có giám sát (supervised learning) trong đó mô hình được huấn luyện trên một tập dữ liệu đã được gán nhãn và bài toán gán nhãn nhị phân (binary labeling).

1.3. Mục tiêu của bài toán :

Xác định mối quan hệ giữa độ khó của khóa học và tỷ lệ hoàn thành, từ đó đưa ra dự đoán tỷ lệ hoàn thành của học viên dựa trên các yếu tố liên quan đến độ khó của khóa học giúp người thiết kế khóa học có thể điều chỉnh khóa học cho phù hợp hơn, tăng tỷ lệ hoàn thành nhưng vẫn duy trì chất lượng khóa học.

Phân tích đặc điểm hành vi học tập: Khám phá và hiểu rõ các yếu tố trong bộ dữ liệu MOOCCubeX có liên quan đến quá trình học tập của học viên. Qua đó, tìm hiểu các yếu tố phổ biến như số lượng bài giảng đã xem, thời gian học tập, số lượng bài tập đã nộp, điểm số, và thời gian hoàn thành bài tập.

Xác định các yếu tố quan trọng: Xác định các yếu tố chủ chốt có ảnh hưởng lớn đến khả năng hoàn thành khóa học của học viên. Điều này bao gồm việc phân tích tương quan giữa hành vi học tập và kết quả hoàn thành khóa học, từ đó làm rõ các yếu tố có khả năng dự đoán cao, chẳng hạn như mức độ tham gia hoạt động học tập, tiến trình học tập, và sự tham gia vào các bài tập kiểm tra.

Xây dựng và đánh giá mô hình dự đoán: Sử dụng các phương pháp học máy (machine learning) để xây dựng các mô hình dự đoán kết quả học tập. Nghiên cứu sẽ thử nghiệm các mô hình như hồi quy logistic, cây quyết định (decision tree), mạng nơ-ron (neural network), và các mô hình tiên tiến khác để so sánh độ chính xác và tính khả thi của từng phương pháp. Mục tiêu cuối cùng là lựa chọn được mô hình có độ chính xác cao và tính ổn định trong việc dự đoán kết quả hoàn thành khóa học của học viên.

Đề xuất cải thiện hiệu suất học tập: Dựa trên kết quả phân tích và dự đoán, đưa ra các đề xuất cải thiện hiệu suất học tập của học viên trên nền tảng MOOC. Điều này bao gồm việc đề xuất các biện pháp hỗ trợ học viên có nguy cơ không hoàn thành khóa học, chẳng hạn như cung cấp nhắc nhở học tập, hỗ trợ tương tác giữa các học viên và giảng viên, và các giải pháp tùy chỉnh theo nhu cầu học tập cá nhân.

1.4. Phạm vi nghiên cứu :

Phạm vi nghiên cứu của đề tài tập trung vào việc phân tích và xây dựng mô hình dự đoán dựa trên dữ liệu từ MOOCCubeX. Bộ dữ liệu này bao gồm nhiều thông tin chi tiết về hành vi học tập của học viên, từ các hoạt động học tập trên nền tảng MOOC cho đến kết quả học tập cụ thể. Phạm vi nghiên cứu bao gồm:

- Dữ liệu về hành vi học tập: Bộ dữ liệu MOOCCubeX cung cấp các thông tin về hành vi của học viên khi tham gia khóa học, như thời gian học tập, số lượng bài giảng đã xem, số bài tập hoàn thành, điểm số, và các hoạt động khác trong quá trình học.
- Phương pháp phân tích và xây dựng mô hình: Nghiên cứu sẽ sử dụng các phương pháp học máy phổ biến như hồi quy logistic, cây quyết định, và các mô hình học sâu để xây dựng mô hình dự đoán. Ngoài ra, các phương pháp tiền xử lý dữ liệu, lựa chọn đặc trưng (feature selection), và đánh giá mô hình cũng được thực hiện để đảm bảo độ chính xác và tính khả thi của mô hình.
- Giới hạn và phạm vi dữ liệu: Nghiên cứu này chỉ giới hạn trong phạm vi bộ dữ liệu MOOCCubeX và không bao gồm dữ liệu từ các nền tảng MOOC khác. Các kết quả và kết luận của nghiên cứu sẽ dựa trên những yếu tố có trong bộ dữ liệu này và có thể không áp dụng hoàn toàn cho các nền tảng học tập khác.

CHƯƠNG 2: BỘ DỮ LIỆU MOOCCUBEX

2.1. Giới thiệu bộ dữ liệu MOOCCubeX

MOOCCubeX được duy trì bởi Nhóm Kỹ thuật Tri thức của Đại học Thanh Hoa và được hỗ trợ bởi XuetangX, một trong những trang web MOOC lớn nhất ở Trung Quốc. Kho lưu trữ này bao gồm 4.216 khóa học, 230.263 video, 358.265 bài tập, 637.572 khái niệm chi tiết và hơn 296 triệu dữ liệu hành vi thô của 3.330.294 sinh viên, để hỗ trợ các chủ đề nghiên cứu về học tập thích ứng trong MOOCs.

Mức độ bao phủ cao: MOOCCubeX có được các tài nguyên MOOC đa dạng và tài nguyên giáo dục bên ngoài, cũng như các bản ghi dữ liệu về việc học tập, tập thể dục và thảo luận của học sinh.

Quy mô lớn: So với kho dữ liệu giáo dục truy cập mở khác, quy mô của MOOCCubeX lớn hơn, từ đó hỗ trợ việc khám phá các mô hình sâu với yêu cầu dữ liệu cao.

Lấy khái niệm làm trung tâm: Dữ liệu không đồng nhất được tổ chức bằng cách sử dụng các khái niệm chi tiết, giúp các tài nguyên trở nên phù hợp hơn và dễ trình bày, tìm kiếm và lập mô hình hơn.

Tác giả: Yu, Jifan and Wang, Yuquan and Zhong, Qingyang and Luo, Gan and Mao, Yiming and Sun, Kai and Feng, Wenzheng and Xu, Wei and Cao, Shulin and Zeng, Kaisheng and others

2.2. Mô tả các bảng:

2.2.1. Bảng user:

2.2.1.2. Mô tả:

Bảng User lưu trữ thông tin chi tiết về người dùng trong hệ thống, bao gồm các thông tin cá nhân cơ bản, dữ liệu về trường học, năm sinh, cùng với các thông tin liên quan đến việc tham gia các khóa học. Các trường dữ liệu trong bảng này giúp xác định người dùng, phân biệt giới tính, và theo dõi quá trình đăng ký các khóa học của họ.

STT	Tên biến	Kiểu dữ liệu	Ý nghĩa
1	id	string	Id của người dùng
2	name	string	Tên người dùng
3	gender	byte	Giới tính
4	school	string	Trường học
5	year_of_birth	int	Ngày sinh
6	course_order	array	Mảng số nguyên chứa id các khóa học đã đăng ký
7	enroll_time	array	Mảng chứa thời gian đăng ký khóa học

2.2.1.2. Phân tích bảng User:

Bộ dữ liệu có 1812921 dòng dữ liệu với đầy đủ thuộc tính cần thiết

– **Id:**

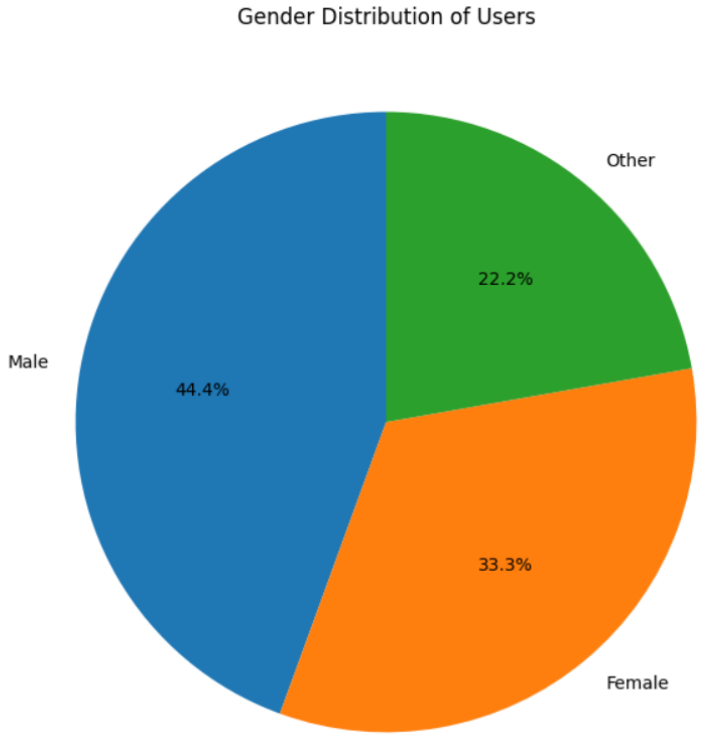
- Giải thích: Đây là mã định danh duy nhất của mỗi người dùng trong hệ thống. Giá trị này đảm bảo không trùng lặp để phân biệt mỗi người dùng.
- Cách sử dụng: ID này giúp truy xuất dữ liệu nhanh chóng cho từng người dùng, liên kết với các bảng khác trong cơ sở dữ liệu khi cần, và đảm bảo tính duy nhất.

– **Name**:

- Giải thích: Tên đầy đủ của người dùng, lưu dưới dạng chuỗi ký tự.
- Cách sử dụng: Trường này được dùng để hiển thị hoặc xác minh thông tin người dùng khi cần. Trong các báo cáo hoặc giao diện, tên giúp phân biệt và xác định người dùng.

– **Gender:**

- Giải thích: Trường này chứa thông tin giới tính, thường lưu dưới dạng mã số. Ví dụ: 0 có thể là Nam, 1 là Nữ, 2 là Khác. Định dạng byte giúp tiết kiệm không gian lưu trữ.
- Cách sử dụng: Được dùng để phân tích thống kê hoặc báo cáo, giúp phân nhóm người dùng theo giới tính hoặc phục vụ mục đích cá nhân hóa dịch vụ.
- Nhận xét:

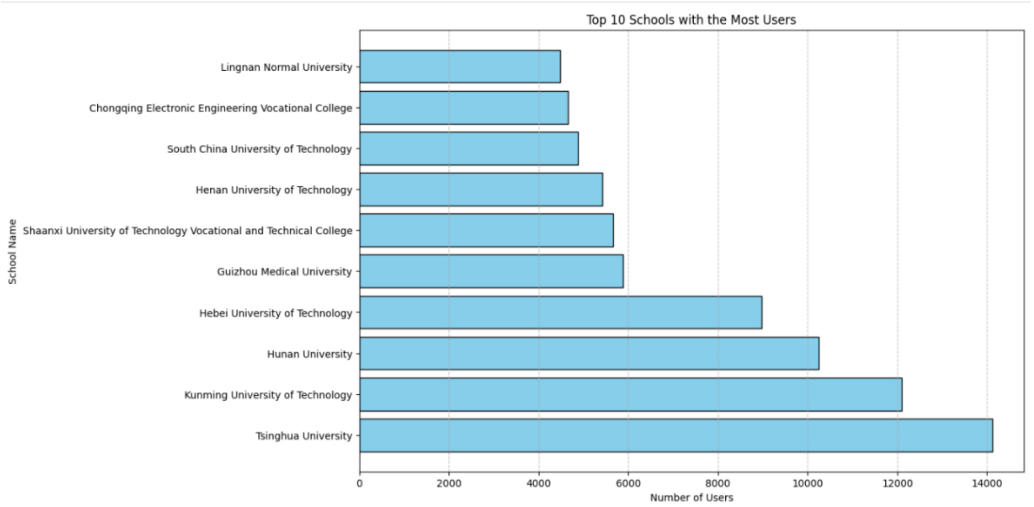


Dựa vào bảng phân bổ giới tính ta thấy có 44.4% giới tính nam và 33.3% giới tính nữ thường tham gia đăng ký học tập cho thấy tỷ lệ nam sinh đăng ký các khóa học chiếm phần đông

– **School:**

- Giải thích: Trường này chứa tên trường học của người dùng. Nó có thể là tên trường cấp 3, đại học hoặc tổ chức nơi người dùng học tập hay công tác.
- Cách sử dụng: Trường này hữu ích trong các báo cáo thống kê để hiểu rõ nguồn gốc học vấn của người dùng, hoặc dùng để cá nhân hóa nội dung liên quan đến học tập và đào tạo.

- Nhận xét:



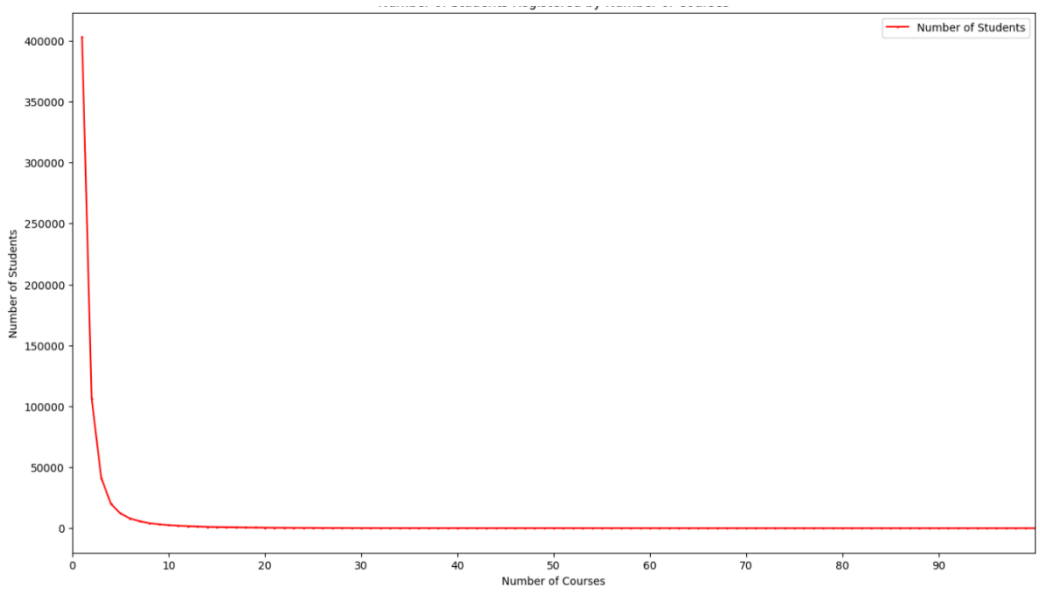
Danh sách các trường đại học có số lượng sinh viên đăng ký khóa học trực tuyến nhiều nhất

– **Year_of_brith**

- Giải thích: Năm sinh của người dùng, lưu dưới dạng số nguyên (ví dụ: 2001, 1995).
- Cách sử dụng: Dữ liệu này giúp tính tuổi của người dùng, hỗ trợ cho các phân tích nhân khẩu học, hoặc điều chỉnh nội dung và dịch vụ phù hợp với độ tuổi.

– **Course_order**

- Giải thích: Mảng này chứa các mã định danh (ID) của các khóa học mà người dùng đã đăng ký. Mỗi phần tử trong mảng là một số nguyên, đại diện cho ID của khóa học.
- Cách sử dụng: Dữ liệu này giúp theo dõi lịch sử học tập của người dùng, cho phép quản lý và đánh giá mức độ tham gia khóa học của họ.
- Nhận xét:



Bảng phân bố từng sinh viên đăng ký số lượng khóa học trên thấy phần đa sinh viên sẽ đăng ký từ 1->2 khóa học trực tuyến. Số lượng sinh viên đăng ký nhiều hơn 10 chiếm phần nhỏ trong bộ dữ liệu

– **enroll_time**

- Giải thích: Mảng này chứa thời điểm đăng ký các khóa học tương ứng với các ID khóa học trong course-order. Mỗi phần tử là một chuỗi ngày giờ để ghi nhận thời gian đăng ký.
- Cách sử dụng: Trường này quan trọng để theo dõi thời gian và thứ tự đăng ký các khóa học của người dùng. Nó giúp phân tích thói quen học tập của người dùng hoặc xác định các mốc thời gian cụ thể khi người dùng tham gia học.

2.2.2. Bảng user-problems

2.2.2.1. Mô tả:

Dữ liệu user-problems cung cấp chứa thông tin về các lần làm bài của người dùng trong một hệ thống, gồm có 2544061 dòng và 7 cột.

Trường dữ liệu	Ý nghĩa	Kiểu dữ liệu
log_id	Mã định danh duy nhất cho mỗi bản ghi trong dữ liệu, thường bao gồm ID của người dùng hoặc bài tập để giúp phân biệt các bản ghi. Trường này giúp truy xuất và tham chiếu đến các lần nộp riêng lẻ của từng người dùng đối với từng bài tập.	string
problem_id	Mã định danh của từng bài tập mà người dùng đã làm. Mỗi bài tập có một problem_id duy nhất, cho phép phân loại các lần nộp theo từng bài tập cụ thể. Trường này hữu ích trong việc phân tích mức độ khó của các bài tập và xu hướng nộp bài theo bài tập	string
user_id	Mã định danh duy nhất cho mỗi người dùng. Trường này giúp xác định người nào đã thực hiện từng lần nộp bài, từ đó hỗ trợ việc theo dõi hoạt động của người dùng, phân tích hiệu suất của từng người, và xác định các mẫu hành vi trong việc làm bài tập.	string
is_correct	Cột boolean (có thể có giá trị 0 hoặc 1) cho biết kết quả của lần nộp bài có đúng hay không. Giá trị "1" cho biết bài làm đúng, trong khi "0" là bài làm sai. Trường này cho phép phân tích tỷ lệ chính xác của người dùng hoặc của từng bài tập.	int
attempts	Số lần thử của người dùng cho một bài tập cụ thể. Trường này giúp xác định số lần mà người dùng đã cần để giải đúng một bài tập, từ đó cung cấp dữ liệu	int

	về độ khó của bài tập và khả năng học tập của người dùng qua các lần thử.	
score	Điểm số người dùng nhận được cho lần nộp bài. Trường này có thể dao động từ giá trị âm cho đến một số dương nhất định, tuỳ vào thang điểm của hệ thống. score giúp đánh giá hiệu suất của người dùng dựa trên chất lượng nộp bài và có thể phản ánh mức độ thành công của mỗi lần thử.	int
submit_time	Thời gian mà người dùng nộp bài, thường được lưu dưới dạng timestamp. Trường này giúp theo dõi thời gian hoạt động của người dùng, phân tích thời gian hoàn thành bài tập, và xác định các khung giờ hoạt động cao điểm.	timestamp

2.2.2.2. Phân tích:

– **Log_id:**

- Dữ liệu cho thấy tổng số lượng log_id là 2,544,060, và tất cả các giá trị đều là duy nhất. Trường log_id này có thể được sử dụng như một định danh duy nhất cho từng bản ghi trong bộ dữ liệu. Điều này giúp đảm bảo không có bản ghi nào bị trùng lặp.
- Nhận xét:
  - Tránh trùng lặp: Vì mỗi log_id là duy nhất, không có bản ghi nào bị lặp lại, đảm bảo tính toàn vẹn của dữ liệu.
  - Theo dõi tương tác: Trường log_id có thể giúp theo dõi cụ thể từng lần tương tác của người dùng với các bài tập, từ đó hỗ trợ phân tích chi tiết hành vi của người dùng theo từng lần nộp

– **Problem_id**

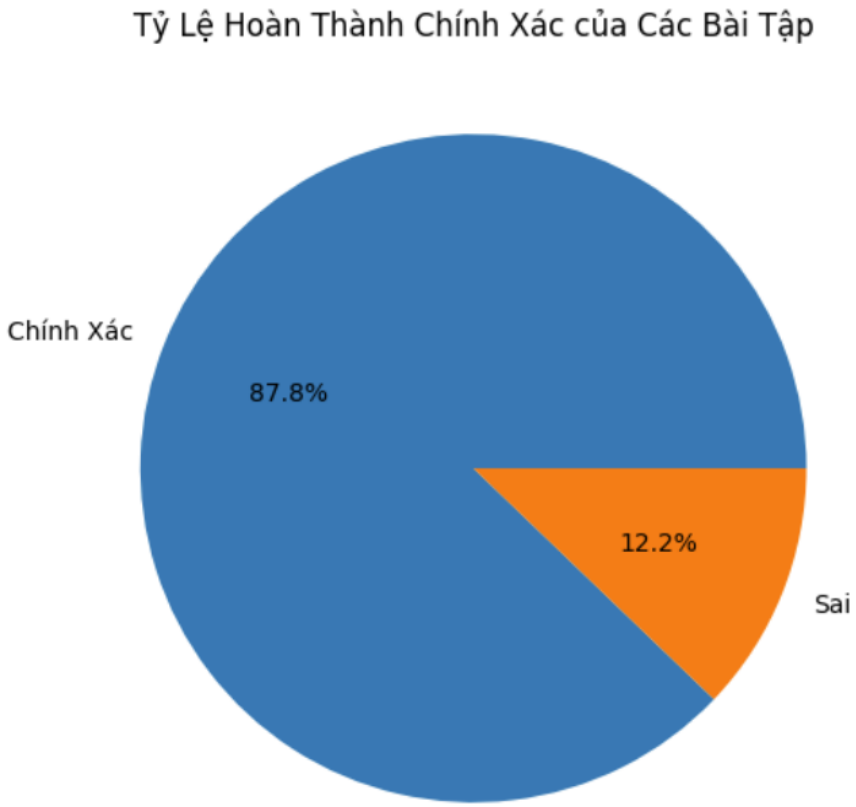
- Bộ dữ liệu này chứa 187,342 bài tập khác nhau được xác định bởi các problem_id khác nhau. Các bài tập này có số lượng lượt làm bài rất khác nhau, trong đó những bài tập phổ biến nhất có lượt làm bài lên đến hàng nghìn. Cụ thể, problem_id như Pm_3295868, Pm_3295869, và Pm_3295865 lần lượt có số lượt làm bài là 4,300, 4,299 và 4,296.
- Nhận xét:
  - Phổ biến và độ khó của bài tập: Các bài tập có lượt làm bài cao có thể là những bài dễ tiếp cận hoặc quan trọng, được đưa vào phần đầu của lộ trình học. Những bài này cũng có thể đóng vai trò nền tảng, yêu cầu người dùng phải vượt qua trước khi tiếp cận các nội dung khó hơn.
  - Khả năng lọc bài tập phổ biến: Các bài tập phổ biến cũng có thể được xem xét để tìm hiểu sâu hơn về các yếu tố khiến người dùng gặp khó khăn, giúp cải thiện nội dung học tập.

– **User_id**

- Thông tin về người dùng:
  - Số lượng người dùng duy nhất: Bộ dữ liệu có 24,128 người dùng khác nhau, đại diện cho một cộng đồng học viên hoặc người làm bài phong phú.
  - Số lần nộp bài trung bình mỗi người dùng: Trung bình, mỗi người dùng đã nộp khoảng 105 lần, cho thấy mức độ tương tác thường xuyên với hệ thống. Con số này có thể được ảnh hưởng bởi nhiều yếu tố, chẳng hạn như độ khó của các bài tập, độ hấp dẫn của nền tảng học tập, hoặc thậm chí là các quy định về số lần làm bài.
  - Phân tích hành vi người dùng: Các người dùng hoạt động nhiều nhất là U_10385081, U_10866892, U_10872553, U_10869570, và U_10868324, với số lần nộp bài từ 1125 đến 1523 lần.
- Nhận xét:
  - Tần suất nộp bài cao: Số lần nộp bài trung bình lớn có thể cho thấy rằng người dùng đang luyện tập nhiều lần, có thể để nâng cao kết quả học tập hoặc khắc phục các lỗi làm bài.

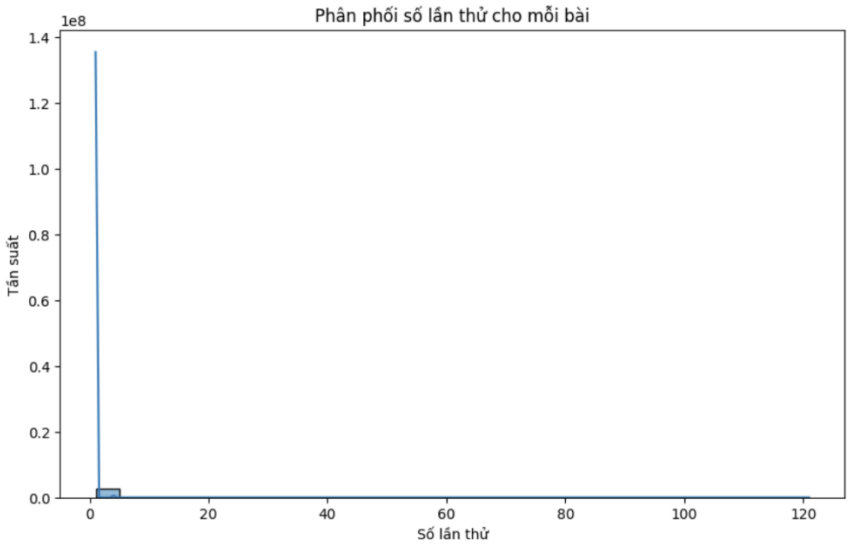
- Những người dùng hoạt động nhiều có thể là những người học tích cực hoặc có thể họ gặp khó khăn với một số bài tập nhất định và phải thử nhiều lần.

– **Is_correct**



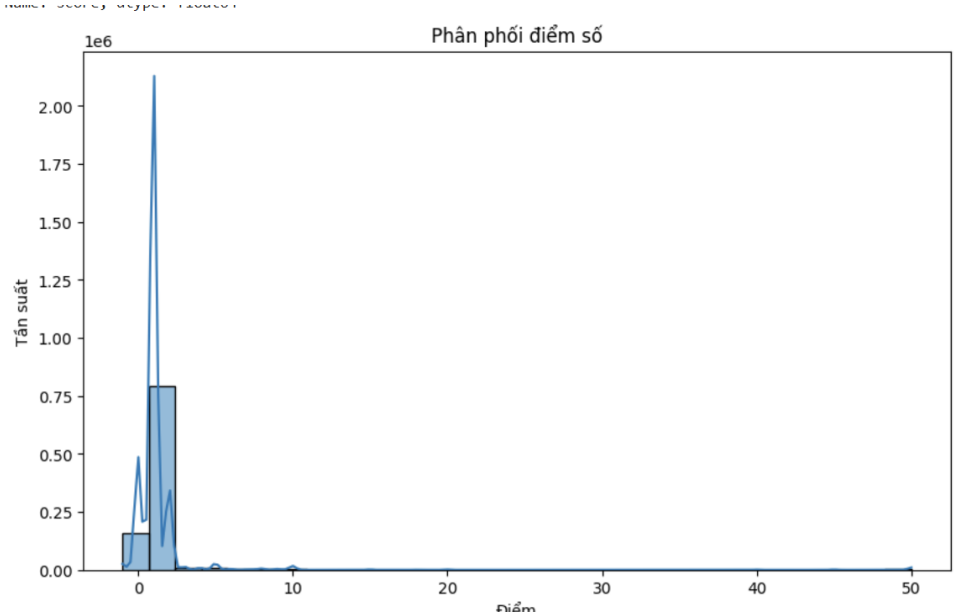
- Tỷ lệ trung bình đạt yêu cầu: Trung bình, trường is_correct cho thấy khoảng 87.85% các bài nộp được làm đúng (mean = 0.8785).
- Phân phối: Vì giá trị trung vị (50%) và giá trị phần tư thứ nhất và thứ ba (25% và 75%) đều là 1, điều này cho thấy phần lớn các bài nộp đều đạt kết quả chính xác.
- Nhận xét: Hầu hết người dùng có thể làm đúng các bài tập, cho thấy các bài tập có thể phù hợp với trình độ người học hoặc có thể không quá khó. Tuy nhiên, vẫn có một tỷ lệ không nhỏ các bài nộp không đạt yêu cầu, có thể là do các bài tập khó hơn hoặc do người học cần nhiều lần thử để thành công.

– **Attempts:**



- Số lượng tổng cộng: Bộ dữ liệu chứa hơn 2.5 triệu lần nộp bài (2,544,060).
- Số lần nộp trung bình: Trung bình mỗi bài tập được nộp khoảng 1.08 lần.
- Phân vị:
  - 25%, 50%, và 75%: Từ phân vị 25 đến 75, số lần nộp duy trì ở mức 1 lần, cho thấy phần lớn người dùng chỉ nộp bài một lần cho mỗi bài tập.
  - Phân phối bất thường: Giá trị lớn nhất (max) là 121 lần nộp cho một bài duy nhất, cho thấy có một số ít trường hợp mà người dùng cần nộp nhiều lần để đạt được kết quả.
- Nhận xét:
  - Khả năng tiếp cận kiến thức: Với phần lớn các bài tập chỉ yêu cầu một lần nộp, điều này có thể cho thấy các bài tập tương đối dễ tiếp thu và có thể hoàn thành trong lần thử đầu tiên.
  - Trường hợp ngoại lệ: Những bài tập có số lần nộp cao có thể là các bài khó hoặc người dùng đã gặp khó khăn trong việc hoàn thành đúng yêu cầu. Các trường hợp này cần được đánh giá thêm để xem có cần điều chỉnh độ khó hoặc cung cấp thêm hướng dẫn cho bài tập.

– **Score**

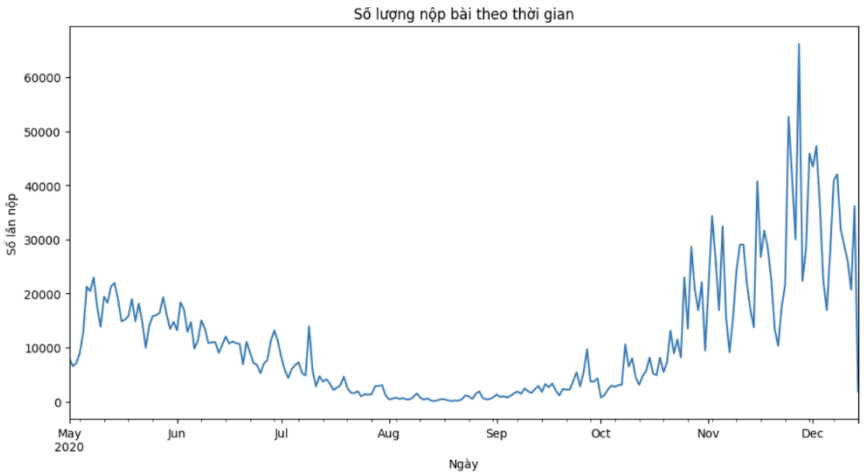


- Dữ liệu điểm số từ các bài nộp cho thấy các đặc điểm nổi bật:
  - Tổng số lần nộp có điểm số: Có 985,320 bản ghi có chứa điểm số, chiếm khoảng 38.7% tổng số lần nộp (2,544,060).
  - Điểm trung bình: Điểm trung bình là 1.32, khá gần với điểm cơ bản 1.0, cho thấy hầu hết các bài nộp đạt điểm ở mức này.
  - Phân vị: 25%, 50%, và 75%: Điểm ở ba phân vị này đều là 1.0, điều này có thể phản ánh một số cơ chế chấm điểm cơ bản hoặc mức điểm tối thiểu đạt yêu cầu cho bài tập.
  - Điểm âm và điểm cao nhất :
    - Điểm thấp nhất: -1.0, điều này có thể đại diện cho một hình phạt hoặc điểm cho các bài nộp không hợp lệ.
    - Điểm cao nhất: 50.0, một giá trị ngoại lệ, có thể phản ánh một loại điểm thưởng hoặc phần thưởng cho những bài làm đặc biệt tốt hoặc bài thi có độ khó cao.

- Nhận xét:

- Phân phối điểm số: Điểm số tập trung quanh giá trị 1.0 cho thấy người dùng thường đạt được mức điểm cơ bản. Tuy nhiên, một số ít người có thể đạt điểm cao hơn đáng kể.
- Điểm âm: Có một lượng nhỏ bài nộp bị trừ điểm hoặc đạt điểm thấp, cần phân tích chi tiết để hiểu rõ lý do, chẳng hạn như xem xét quy tắc chấm điểm hoặc lỗi người dùng.

– **submit_time**



- Biểu đồ cho thấy sự thay đổi trong số lượng lần nộp bài theo thời gian từ tháng 5 đến tháng 12 năm 2020.
- Nhận xét:
  - Sự tăng mạnh về số lần nộp bài: Đồ thị cho thấy số lần nộp bài có xu hướng tăng dần từ tháng 10 đến tháng 12. Đặc biệt, vào các ngày cuối tháng 11 và tháng 12, có những đỉnh điểm với số lần nộp bài lên đến hơn 60,000. Điều này có thể là do thời điểm cuối kỳ, khi người dùng (có thể là sinh viên) phải hoàn thành các bài tập hoặc chuẩn bị cho các kỳ thi.
  - Sự giảm trong mùa hè: Trong khoảng tháng 6 đến tháng 9, số lượng nộp bài giảm đi đáng kể. Đây có thể là giai đoạn nghỉ hè, dẫn đến ít hoạt động học tập và nộp bài hơn.
  - Sự gia tăng trở lại vào tháng 10: Từ tháng 10, số lần nộp bài bắt đầu tăng mạnh, có thể do các hoạt động học tập được tiếp tục sau kỳ nghỉ hè.

2.2.3. User-video:

2.2.3.1. Mô tả:

Bảng User_video lưu trữ thông tin chi tiết về các đoạn video mà người dùng đã xem trong hệ thống, bao gồm mã định danh của người dùng, ID của video, thời gian bắt đầu và kết thúc của từng đoạn xem, tốc độ phát, và thời điểm xem theo múi giờ địa phương. Các trường dữ liệu trong bảng này giúp xác định từng video và từng đoạn video mà người dùng đã tương tác, theo dõi thói quen xem, cũng như phân tích hành vi người dùng dựa trên thời lượng và tốc độ xem của họ.

Bộ dữ liệu gồm 113024 dòng và 7 cột và các trường được hiển thị trong bảng dưới đây:

Tên trường	Kiểu dữ liệu	Mô tả
seq	list	Danh sách chứa các video và các đoạn phát mà người dùng đã xem.
video_id	string	Mã định danh duy nhất của mỗi video, dùng để phân biệt và truy xuất thông tin cụ thể của video.
segment	list	Danh sách các đoạn phát mà người dùng đã xem trong một video cụ thể.
start_point	float	Thời điểm bắt đầu của đoạn phát trong video, tính bằng giây từ đầu video.
end_point	float	Thời điểm kết thúc của đoạn phát trong video, tính bằng giây từ đầu video.
speed	float	Tốc độ phát của đoạn video trong đoạn đó. Giá trị 1.0 là tốc độ bình thường, 1.25, 1.5, 2.0,... là tốc độ nhanh hơn.
local_start_time	int	Thời điểm thực tế khi người dùng bắt đầu phát đoạn video, tính bằng Unix timestamp

		(số giây từ 00:00:00 UTC ngày 1 tháng 1 năm 1970).
user_id	string	Mã định danh duy nhất cho người dùng, giúp theo dõi lịch sử xem video của từng người.

2.2.3.2. Phân tích:

- **user_id:**

- Tổng số lượng user_id: Mỗi user_id là duy nhất và xác định một người dùng riêng biệt trong hệ thống.
- Ý nghĩa: user_id giúp phân biệt người dùng và theo dõi hành vi xem video của từng cá nhân. Việc này quan trọng để cá nhân hóa nội dung, theo dõi quá trình học tập, và đánh giá mức độ tương tác của mỗi người dùng.

- **video_id:**

- Tổng số lượng video_id: Có nhiều video khác nhau trong hệ thống, mỗi video có một video_id duy nhất.
- Phân tích số lượng đoạn xem trong từng video_id:
  - Số lượng đoạn xem trong mỗi video có thể được nhóm và đếm để hiểu tần suất người dùng quay lại xem các đoạn trong một video nhất định.
  - Ví dụ:
    - V_1395633: 4 đoạn xem
    - V_1395635: 1 đoạn xem
    - V_6210799: 4 đoạn xem
- Ý nghĩa: video_id cho phép xác định các video cụ thể mà người dùng xem và giúp phân tích độ phổ biến của từng video. Những video có nhiều lượt xem hoặc được xem nhiều đoạn có thể là những video mang lại giá trị cao hoặc có nội dung khó hiểu, cần xem lại.

- **Segment:**

- Thống kê các trường trong segment:
  - start_point và end_point: Dữ liệu này cho biết vị trí mà người dùng bắt đầu và kết thúc mỗi lần xem, từ đó có thể tính toán độ dài của các đoạn xem. Những đoạn video thường xuyên được tua đi tua lại hoặc bỏ qua có thể cần cải thiện về nội dung.
  - speed: Tốc độ phát lại video cũng cung cấp thông tin về mức độ hấp dẫn hoặc khó hiểu của nội dung. Các trường hợp phát nhanh hơn (như 1.25 hoặc 1.5) thường cho thấy nội dung dễ hiểu hoặc quen thuộc, trong khi tốc độ bình thường có thể chỉ ra nội dung phức tạp hơn.
  - Thống kê tốc độ phát lại:
    - Tốc độ 1.0 (bình thường): Xác định nội dung được xem kỹ hoặc khó hiểu.
    - Tốc độ 1.25 - 1.5: Người xem thường đẩy nhanh những phần dễ hiểu hoặc kém hấp dẫn.
  - local_start_time: Giúp xác định thời điểm người dùng xem video. Các video được xem vào khung giờ nhất định (như tối hoặc cuối tuần) có thể là nội dung liên quan đến học tập hoặc giải trí.
- Ý nghĩa:
  - Phân tích hành vi xem video: segment cho phép phân tích các đoạn nào được xem nhiều nhất, thời lượng trung bình mỗi đoạn, và thói quen tua nhanh video của người dùng. Điều này giúp tối ưu nội dung hoặc cải thiện trải nghiệm học tập.
  - Tương tác và động lực học tập: Nếu người dùng xem đi xem lại nhiều lần hoặc có xu hướng tua nhanh, nội dung video có thể cần điều chỉnh để phù hợp hơn với nhu cầu học tập.

⇨ Nhận xét:

- Độ Đa Dạng Và Khả Năng Hiểu Video:
  - Các video chứa nhiều đoạn xem cho thấy người dùng có thể gặp khó khăn hoặc nội dung phức tạp. Ngược lại, nếu các đoạn được xem ít và chủ yếu ở tốc độ nhanh, điều này có thể chỉ ra nội dung đơn giản hoặc không thu hút.

- Mức độ phức tạp của video: Video có nhiều đoạn xem ngắn có thể cần được điều chỉnh độ dài hoặc cấu trúc lại nội dung, tạo điều kiện thuận lợi cho người dùng tiếp thu.

- Động Lực và Tỷ Lệ Hoàn Thành Khóa Học:
  - Động lực học tập: Video được xem liên tục hoặc nhiều lần có thể là tài liệu cần thiết, có sức hấp dẫn, hoặc khó hiểu. Khi video dễ hiểu, người dùng có xu hướng xem nhanh và tỷ lệ hoàn thành có thể cao hơn.
  - Phân bổ video hợp lý: Nếu các video có độ khó hoặc độ dài khác nhau nhưng được sắp xếp hợp lý, người dùng sẽ có động lực hơn trong quá trình học tập. Ngược lại, nếu nội dung quá dài hoặc khó, người dùng có thể cảm thấy áp lực và bỏ qua các phần khác.
- Kết Luận:
  - Cấu trúc và tính đa dạng của video: Video có thời lượng và nội dung phong phú, dễ tiếp cận giúp cải thiện động lực học tập. Nếu video có độ dài hợp lý, nội dung hấp dẫn và được phân đoạn phù hợp, người dùng có thể dễ dàng tiếp thu và hoàn thành khóa học. Ngược lại, nếu nội dung quá phức tạp hoặc đơn điệu, người dùng sẽ có xu hướng xem lướt qua, tua nhanh, hoặc bỏ dở.
  - Ứng dụng trong cải tiến nội dung: Dựa vào hành vi xem video, nhà quản lý khóa học có thể điều chỉnh nội dung video sao cho phù hợp hơn, cải thiện tỷ lệ hoàn thành khóa học và trải nghiệm học tập của người dùng.

2.2.4. Problem:

2.2.4.1. Mô tả:

Các nội dung cụ thể của bài tập bao gồm trong khóa học. Lưu ý rằng mỗi nhóm bài tập (bài tập) tương ứng với nhiều câu hỏi (vấn đề). Các trường được hiển thị trong bảng dưới đây.

Bộ dữ liệu gồm 1048576 dòng, 12 cột.

Trường dữ liệu	Ý nghĩa	Kiểu dữ liệu
id	Mã định danh duy nhất cho mỗi câu hỏi, bắt đầu bằng Pm_	string
exercise_id	Mã định danh duy nhất của loại bài tập, bắt đầu bằng Ex_	string
language	Mô tả ngôn ngữ của câu hỏi, Chinese/ English	string
title	Tiêu đề của bài tập	string
content	Mô tả nội dung câu hỏi	string
option	Các phương án lựa chọn	string
answer	Câu trả lời của câu hỏi	string
score	Điểm số của câu hỏi	float
type	Loại câu hỏi bằng số	int
typetext	Loại câu hỏi bằng chữ	string
location	Vị trí chương/ chuyên đề của câu hỏi	float
context_id	Mảng, leaf_id liên quan đến vấn đề	string

2.2.4.2. Phân tích:

– **id:**

Tổng số lượng id là 1048576, và tất cả các giá trị đều là duy nhất. Trường id này có thể được sử dụng như một định danh duy nhất cho từng câu hỏi trong bộ dữ liệu. Điều này giúp đảm bảo không có câu hỏi nào bị trùng lặp.

– excersie_id:

- Excersise sẽ được chia thành nhiều nhóm khác nhau do đó cần phải có id (mã định danh) để dễ dàng phân biệt chúng.
- Số lượng bài tập trong từng nhóm exercise_id:
- exercise_id

Ex_100000	5
Ex_100009	8
Ex_100013	5
Ex_100018	4
Ex_1001259	20
..	
Ex_998829	7
Ex_998836	5
Ex_998837	6
Ex_998838	7
Ex_99996	4

– **Typetext**:

- Dựa vào cột typetext mình sẽ xem xét câu hỏi đó thuộc dạng câu hỏi nào. Và đây là số liệu thống kế được Single Choice Question: 58.3%, Multiple Choice Question: 20.9%, Fill in the Blanks: 15.1%, True/False Question: 4.0%, Subjective Question: 1.7%, Voting Question: 0.0%
- Number of Problem Types:
  - Single Choice Question: 583456
  - Multiple Choice Question: 209245
  - Fill in the Blanks: 151287
  - True/False Question: 40277
  - Subjective Question: 16607
  - Voting Question: 355

– **Nhận xét:**

- Tính Đa Dạng Của Bài Tập (Typetext):
  - Ảnh hưởng đến kỹ năng cần thiết: Việc có nhiều loại bài tập khác nhau như Single Choice Question, Multiple Choice Question, Fill in the Blanks, True/False, và Subjective Question tạo ra một môi trường học tập phong phú, yêu cầu học sinh vận dụng nhiều kỹ năng khác nhau. Để hoàn thành khóa học, học sinh phải có khả năng trả lời không chỉ những câu hỏi có tính khách quan mà còn cả các bài tự luận, phức tạp hơn.
  - Khả năng nắm bắt kiến thức: Các dạng câu hỏi trắc nghiệm (như Single Choice và Multiple Choice) thường kiểm tra kiến thức cơ bản, trong khi dạng tự luận giúp đánh giá khả năng hiểu sâu và phân tích vấn đề. Nếu phần lớn bài tập là trắc nghiệm, học sinh có thể hoàn thành khóa học với kiến thức nông. Ngược lại, với nhiều câu hỏi tự luận, học sinh có thể cảm thấy khó khăn hơn nhưng lại có cái nhìn sâu hơn về nội dung học.
- Độ Phức Tạp và Phân Nhóm Bài Tập (exercise_id):
  - Mức độ thử thách theo nhóm: Các bài tập có thể được tổ chức theo nhóm exercise_id, có nghĩa là một nhóm bài tập có thể đi theo một chủ đề hoặc nội dung cụ thể. Nếu một nhóm có các câu hỏi phức tạp, nó có thể ảnh hưởng đến tốc độ hoàn thành của học sinh.
  - Phân bổ bài tập theo nhóm: Nếu bài tập trong một nhóm đều có độ khó cao hoặc yêu cầu kiến thức sâu, học sinh có thể cần nhiều thời gian hơn để hoàn thành, dẫn đến tỷ lệ hoàn thành khóa học thấp hơn. Ngược lại, nếu bài tập trong nhóm được phân bổ hợp lý từ dễ đến khó, học sinh có thể cảm thấy động lực hơn và có khả năng hoàn thành khóa học cao hơn.
- Mức Độ Tương Quan Giữa Các Nhóm Bài Tập:
  - Nếu các bài tập trong từng exercise_id có liên quan đến nhau về nội dung, học sinh sẽ dễ dàng kết nối các kiến thức lại với nhau, giúp học nhanh và củng cố kiến thức theo từng phần. Ngược lại, nếu các bài tập thiếu sự liên kết, học sinh

có thể gặp khó khăn khi hoàn thành khóa học, do họ phải học nhiều phần kiến thức rời rạc mà không thấy mối liên hệ giữa chúng.

- Tỷ Lệ Các Loại Câu Hỏi Và Động Lực Học Tập:
  - Khi phần lớn câu hỏi là dạng trắc nghiệm, học sinh có thể cảm thấy ít áp lực hơn, từ đó tỷ lệ hoàn thành có thể cao hơn. Tuy nhiên, nếu khóa học chủ yếu là các câu hỏi tự luận hay phức tạp, điều này có thể khiến học sinh nản lòng và giảm tỷ lệ hoàn thành.
  - Đồng thời, sự cân bằng giữa các loại câu hỏi sẽ giúp giữ sự hứng thú của học sinh, khiến họ ít cảm thấy nhàm chán hoặc quá tải. Tỷ lệ hoàn thành có thể cao hơn nếu học sinh được trải nghiệm một khóa học có cấu trúc bài tập đa dạng và được thử thách một cách hợp lý.

⇨ **Kết luận:** Cấu trúc của các bài tập và tính đa dạng của chúng có thể ảnh hưởng đến tỷ lệ hoàn thành khóa học. Một khóa học với các bài tập đa dạng, được phân nhóm hợp lý và có tính thử thách sẽ giúp học sinh học tập hiệu quả hơn và có thể dẫn đến tỷ lệ hoàn thành cao hơn. Ngược lại, nếu bài tập quá khó hoặc thiếu sự liên kết, học sinh có thể mất động lực và dễ dàng bỏ cuộc.

CHƯƠNG 3: PHƯƠNG PHÁP ĐỀ XUẤT

3.1. Các bước thực hiện tổng quan từ input đến output của bài toán:

Bước 1: Chuẩn Bị Bộ Dữ Liệu

- **Mục tiêu**: Tập hợp các bảng dữ liệu cần thiết từ bộ dữ liệu MOOCCubeX.
- **Input**: bộ dữ liệu

Bước 2: Tiền Xử Lý Dữ Liệu

- **Mục tiêu**: Làm sạch dữ liệu để đảm bảo tính chính xác và độ tin cậy.
- **Hoạt động**:
    - Loại bỏ các cột không cần thiết (ví dụ: name, course_order, seq).
    - Tính toán các đặc trưng mới cần thiết, chẳng hạn như watch_ratio (tỷ lệ video đã xem so với yêu cầu).
    - Gán nhãn completion_status dựa trên điều kiện tỷ lệ xem (thường là 0 hoặc 1).

Bước 3: Chia dữ liệu và cân bằng dữ liệu

- **Mục tiêu:** Đảm bảo có đủ dữ liệu huấn luyện và kiểm tra để đánh giá mô hình một cách khách quan. Việc chia dữ liệu thành các tập huấn luyện và kiểm tra giúp đánh giá khả năng tổng quát của mô hình. Đảm bảo rằng các lớp trong dữ liệu phân loại có sự phân bố đồng đều. SMOTE giúp tạo mẫu giả cho các lớp ít đại diện hơn, tránh tình trạng mô hình chỉ học được các đặc trưng của lớp phổ biến hơn (overfitting cho lớp chiếm ưu thế).
- **Hoạt động:**
    - **Chia dữ liệu:** Dữ liệu được chia thành 2 tập: tập huấn luyện (train) và tập kiểm tra (test) với tỷ lệ 70%/30% hoặc 80%/20%.
    - **Cân bằng dữ liệu:** Áp dụng kỹ thuật SMOTE (Synthetic Minority Over-sampling Technique) để tạo ra các mẫu giả cho lớp ít xuất hiện, giúp cân bằng các lớp trong bài toán phân loại.

Bước 4: Chuẩn hóa dữ liệu

- **Mục tiêu**: Đảm bảo tính đồng nhất và tăng hiệu quả huấn luyện.
- **Hoạt động**: Sử dụng các kỹ thuật chuẩn hóa như StandardScaler để chuẩn hóa dữ liệu huấn luyện và kiểm tra. Việc này giúp các thuật toán học máy hoạt động hiệu quả hơn, đặc biệt là đối với các thuật toán như Logistic Regression hoặc Neural Networks.

Bước 5: Xây dựng mô hình :

- **Mục tiêu**: Lựa chọn thuật toán phù hợp .
- **Hoạt động**:
    - Logistic Regression: Sử dụng mô hình phân loại tuyến tính với các tham số như penalty, C, solver để tối ưu mô hình.
    - Neural Network: Xây dựng mạng nơ-ron sâu (Deep Neural Network) với các lớp ẩn và đầu ra sử dụng hàm kích hoạt như ReLU (cho các lớp ẩn) và Sigmoid (cho lớp đầu ra).
    - Random Forest: Xây dựng mô hình với các cây quyết định, tối ưu hóa các tham số như số lượng cây (n_estimators), độ sâu của cây (max_depth), và số mẫu tối thiểu tại mỗi lá cây (min_samples_leaf).

Bước 6: Huấn luyện mô hình

- **Mục tiêu**:Tối ưu hóa mô hình và học các đặc trưng quan trọng.
- **Hoạt động**: Sử dụng dữ liệu huấn luyện đã được chuẩn hóa và cân bằng (nếu có SMOTE) để huấn luyện các mô hình. Quá trình này sẽ bao gồm việc điều chỉnh các tham số để tối ưu hóa hiệu suất của mô hình.

Bước 7: Dự đoán và đánh giá mô hình

- **Mục tiêu**: Đánh giá khả năng dự đoán và đo lường hiệu quả mô hình.
- **Hoạt động**:
    - **Dự đoán**: Sử dụng mô hình đã huấn luyện để dự đoán nhãn của tập kiểm tra (test)Thiết kế kiến trúc dữ liệu lớn để dễ dàng truy cập và xử lý.

- **Đánh giá mô hình:** Đánh giá mô hình bằng cách sử dụng các chỉ số như độ chính xác (accuracy), precision, recall, F1-score và classification report.

Bước 8: Chuyển đổi xác suất (nếu cần) :

- **Mục tiêu :** Nếu mô hình dự đoán xác suất (ví dụ: Logistic Regression và Neural Network), chuyển đổi xác suất thành nhãn 0 hoặc 1 để có kết quả rõ ràng (hoàn thành hoặc không hoàn thành). Việc chọn ngưỡng (thường là 0.5) giúp điều chỉnh các kết quả phân loại theo yêu cầu cụ thể của bài toán
- **Hoạt động :** Xác suất dự đoán: Một số mô hình (như Logistic Regression, Neural Network, hoặc Random Forest) sẽ trả về xác suất dự đoán thay vì nhãn trực tiếp. Ta có thể chuyển đổi xác suất này thành nhãn (0 hoặc 1) bằng cách chọn ngưỡng (thường là 0.5).

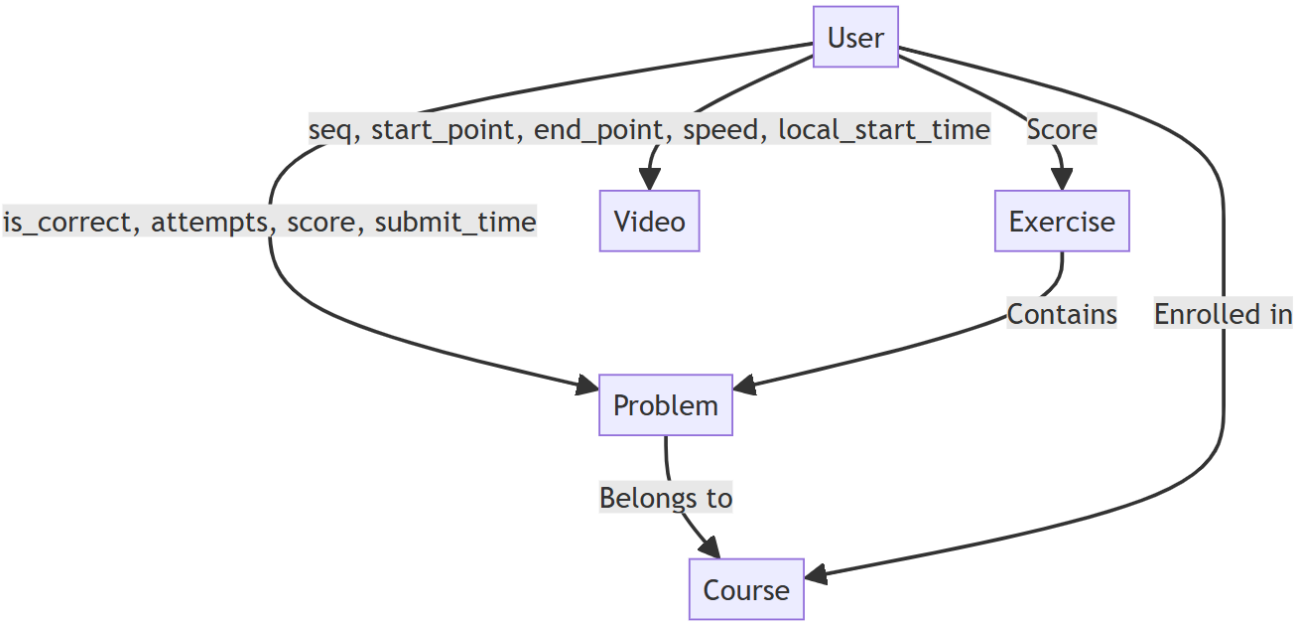
Bước 9 : Trực quan hóa kết quả

- **Mục tiêu :** Trực quan hóa hiệu quả mô hình.
- **Hoạt động:**
  - Trực quan hóa các kết quả dự đoán, chẳng hạn như độ chính xác của mô hình, phân phối của các lớp dự đoán, hoặc độ quan trọng của các đặc trưng trong mô hình.
  - Có thể sử dụng biểu đồ phân loại (confusion matrix), đồ thị ROC/AUC, hoặc biểu đồ tròn để thể hiện kết quả của mô hình.

Bước 10 : Điều chỉnh và tối ưu mô hình :

- **Mục tiêu :** Cải thiện mô hình, Giảm overfitting và underfitting.
- **Hoạt Động :**
  - **Tối ưu mô hình**: Dựa trên kết quả đánh giá, điều chỉnh các tham số của mô hình (hyperparameters) hoặc thử các mô hình khác nhau để cải thiện kết quả.
  - **Cross-validation**: Có thể sử dụng cross-validation để đảm bảo mô hình không bị overfitting và có hiệu quả tốt trên dữ liệu chưa thấy

3.2. Ý tưởng xây dựng đồ thị mạng:



Mô tả:

Mô hình đồ thị này sẽ bao gồm các **nút** (nodes) đại diện cho các thực thể như người dùng, bài tập, video, khóa học và các **cạnh** (edges) đại diện cho các mối quan hệ giữa chúng.

- **Nút (Nodes):**
  - **User**: Đại diện cho người dùng trong hệ thống.
  - **Problem**: Các vấn đề liên quan giữa người dùng và khóa học.
  - **Exercise**: Đại diện cho bài tập (nhóm câu hỏi).
  - **Course**: Đại diện cho khóa học, nơi các bài tập và câu hỏi được tổ chức.
  - **Video**: Đại diện cho các video mà người dùng đã xem.
- **Cạnh (Edges):**
  - **User → Problem**: Mối quan hệ giữa người dùng và một số vấn đề liên quan đến bài tập mà họ đã làm (thông qua bảng User-Problem).
    - is_correct, attempts, score, submit_time: Các thuộc tính giúp phân tích kết quả làm bài của người dùng.

- **User → Exercise**: Mối quan hệ giữa người dùng và bài tập mà họ đã tham gia. Cung cấp thông tin về bài tập mà người dùng đã hoàn thành và điểm số đạt được.
- **Exercise→Problem**: Mối quan hệ giữa câu hỏi và bài tập. Mỗi bài tập bao gồm một hoặc nhiều câu hỏi.
- **Problem → Course**: Mối quan hệ giữa câu hỏi và khóa học mà câu hỏi thuộc về. Cho phép chúng ta biết câu hỏi thuộc khóa học nào.
- **User → Video**: Mối quan hệ giữa người dùng và video mà họ đã xem.
  - seq, start_point, end_point, speed, local_start_time: Các thuộc tính giúp phân tích thói quen xem video của người dùng.
- **User → Course**: Mối quan hệ giữa người dùng và khóa học mà họ đã tham gia.

3.3. Các chức năng của mô hình mạng

- Phân tích hành vi người dùng:
  - Mối quan hệ giữa người dùng và các câu hỏi giúp phân tích hiệu suất làm bài tập của họ (số lần thử, điểm số, tỷ lệ đúng sai).
  - Mối quan hệ giữa người dùng và video giúp theo dõi thói quen xem video, thời gian xem, và tốc độ phát video.
- Phân tích khóa học và bài tập:
  - Phân tích các bài tập và câu hỏi trong các khóa học để đánh giá mức độ khó của chúng.
  - Từ đó, tối ưu hóa khóa học và bài tập sao cho phù hợp với người học.
- Cá nhân hóa trải nghiệm học tập:
  - Gợi ý video hoặc câu hỏi bổ sung dựa trên hành vi học tập của người dùng.
  - Dự đoán kết quả của người dùng trong các bài tập hoặc khóa học tiếp theo dựa trên dữ liệu lịch sử.

3.4. Mô hình huấn luyện

- **Logistic Regression:** Mô hình đơn giản và dễ giải thích, phù hợp với bài toán phân loại nhị phân.
- **Decision Tree:** Mô hình dễ trực quan hóa và hiểu rõ quy trình ra quyết định.

- **Random Forest:** Mô hình mạnh hơn Decision Tree, sử dụng nhiều cây quyết định để cải thiện độ chính xác.
- **Support Vector Machine (SVM):** Tạo ra một siêu phẳng để phân loại dữ liệu.
- **Neural Networks:** Mô hình phức tạp hơn, có khả năng học các mối quan hệ phi tuyến tính giữa các đặc trưng trong dữ liệu.
- Sử dụng tập huấn luyện để đào tạo mô hình, tức là mô hình sẽ học từ dữ liệu đã biết nhãn (hoàn thành hoặc không hoàn thành khóa học).

CHƯƠNG 4: THỰC NGHIỆM

4.1 Tiền xử lý dữ liệu:

STT	Thao tác xử lý	Cách hoạt động	Đối tượng áp dung
1	Loại bỏ các ký tự đặc biệt	Gồm các ký tự dấu chấm câu, dấu ngoặc, ký hiệu đặc biệt, các icon. Những ký tự này không cần thiết cho việc xử lý dữ liệu cũng như chạy mô hình đồ thị mạng	user, course
2	Loại bỏ các dữ liệu có giá trị NaN	Các giá trị này không thể xác định và không thể xử lý, dữ liệu nhiều nên sẽ loại bỏ hàng dữ liệu đó	user-problem
3	Loại bỏ cột dữ liệu	Những cột dữ liệu không cần thiết cho việc xử lý và chạy mô hình đồ thị mạng	user, course

4.2 Trực quan hóa dữ liệu đầu vào:

4.2.1 Mục Đích Của Trực Quan Hóa Dữ Liệu:

Trực quan hóa dữ liệu là một công cụ quan trọng giúp bạn hiểu rõ hơn về các mối quan hệ giữa các biến trong dữ liệu. Trong bối cảnh của bài toán này, mục tiêu là:

- Khám Phá Mối Quan Hệ: Xác định các mối quan hệ giữa các input (như thông tin học sinh, bình luận, lượt giải bài tập, và thông tin khóa học) và kết quả mong muốn (mức độ hài lòng của học viên).
- Phát Hiện Xu Hướng: Nhận diện các xu hướng có thể ảnh hưởng đến mức độ hài lòng của học viên.
- Hỗ Trợ Ra Quyết Định: Cung cấp cái nhìn tổng quan để hỗ trợ các quyết định trong việc cải thiện chất lượng khóa học

4.3 Các Phương Pháp Trực Quan Hóa:

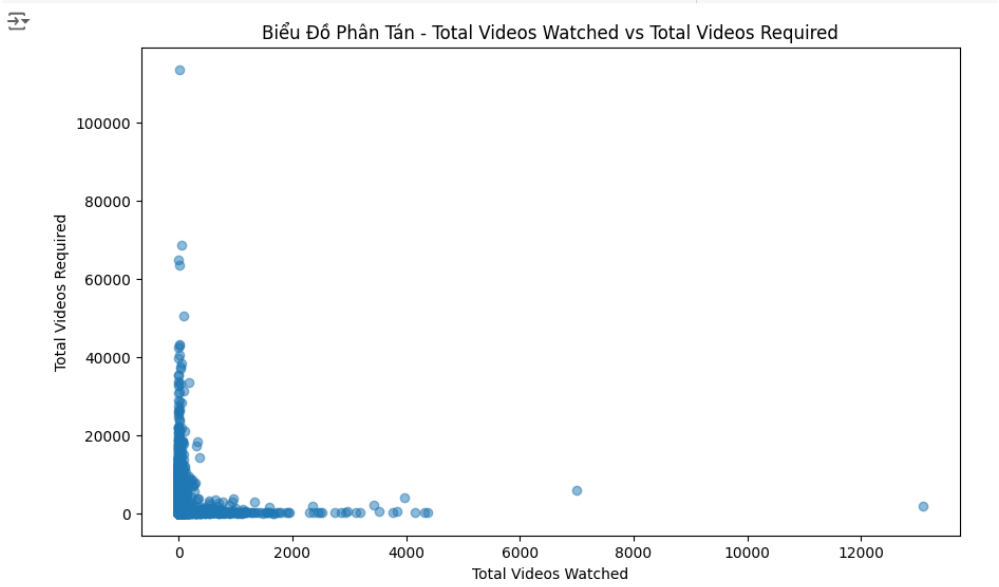
4.3.1 Biểu Đồ Phân Tán (Scatter Plot):

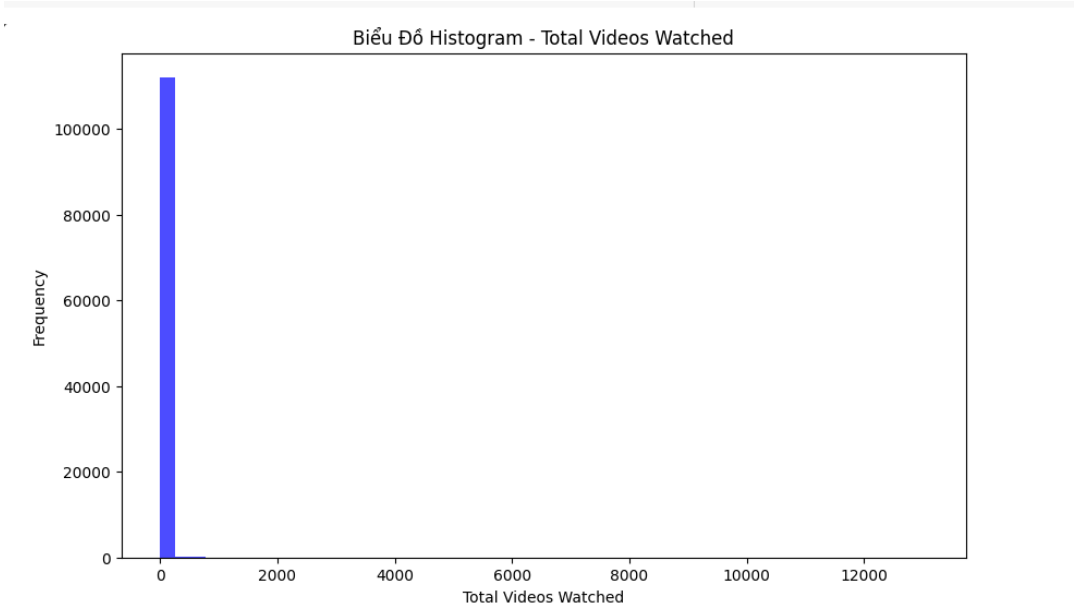
- Mô tả: Biểu đồ phân tán cho phép bạn xem mối quan hệ giữa hai biến số. Mỗi điểm trên biểu đồ đại diện cho một cặp giá trị của hai biến.
- Ứng dụng: sử dụng biểu đồ phân tán để thể hiện mối quan hệ giữa độ khó của khóa học và tỷ lệ hoàn thành. Điều này giúp xác định xem có mối liên hệ nào giữa độ khó và khả năng hoàn thành khóa học hay không.

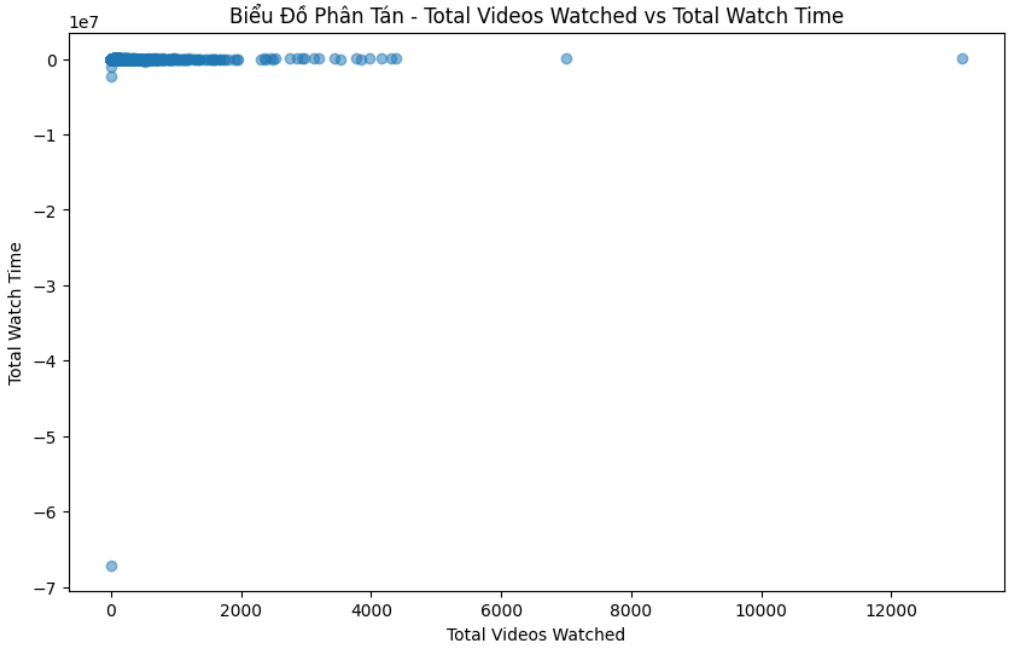
4.3.2 Biểu Đồ Histogram:

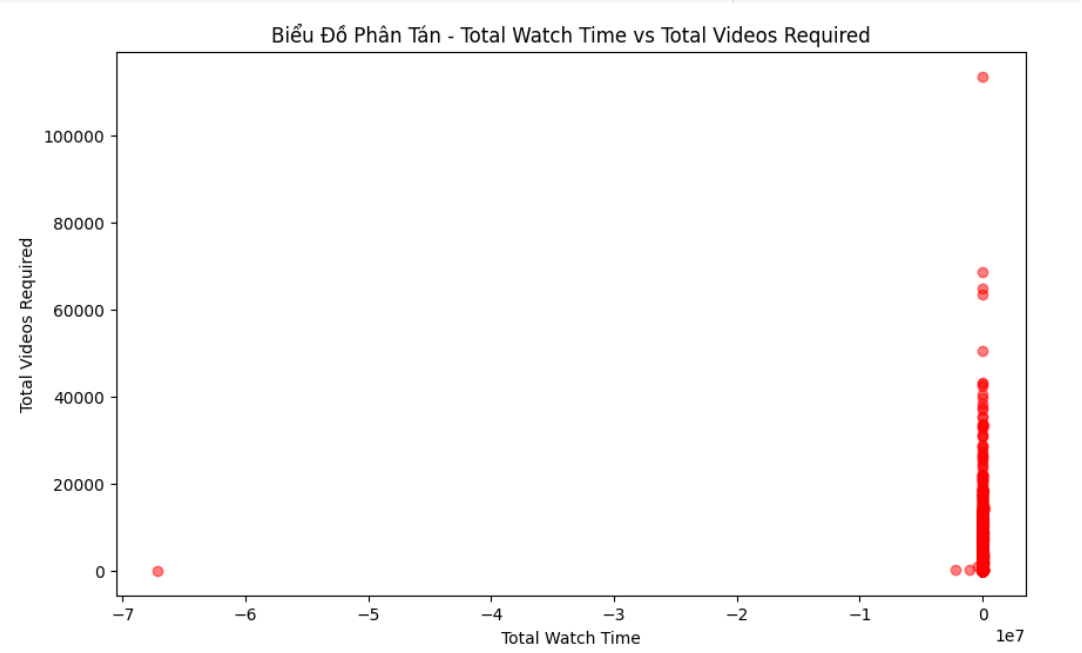
- Mô tả: Biểu đồ histogram thể hiện phân phối của một biến số bằng cách chia dữ liệu thành các khoảng (bins) và đếm số lượng điểm dữ liệu trong mỗi khoảng.
- Ứng dụng: Sử dụng histogram để phân tích phân phối độ khó của các khóa học hoặc phân phối mức độ hài lòng của học viên. Điều này giúp hiểu rõ hơn về cách mà các khóa học được đánh giá.

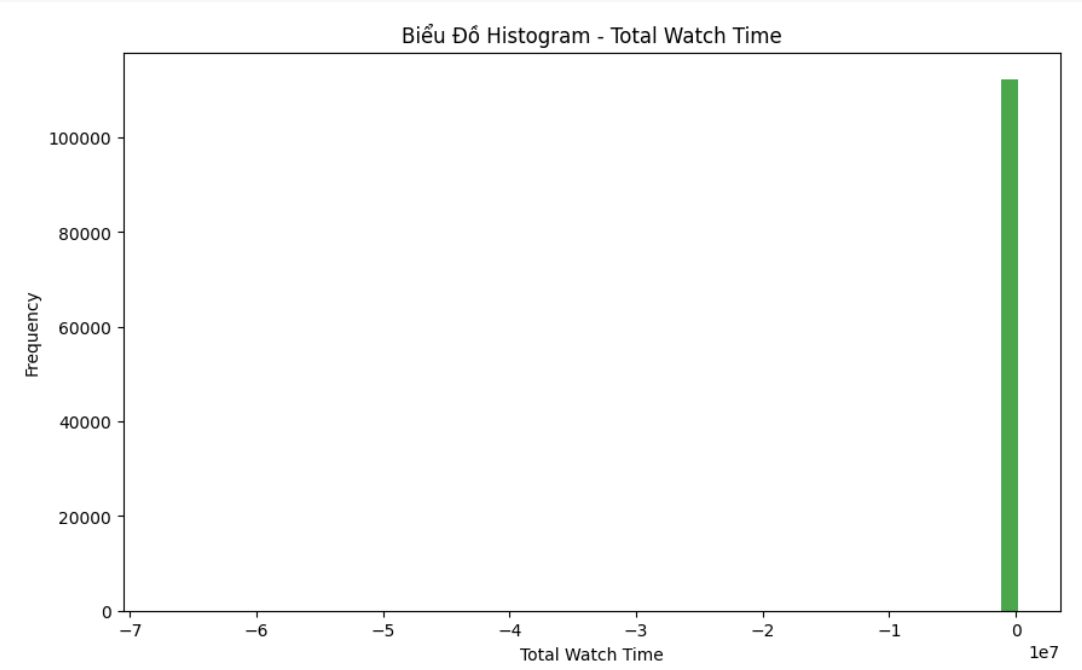
4.3.3 Biểu đồ trực quan hóa:

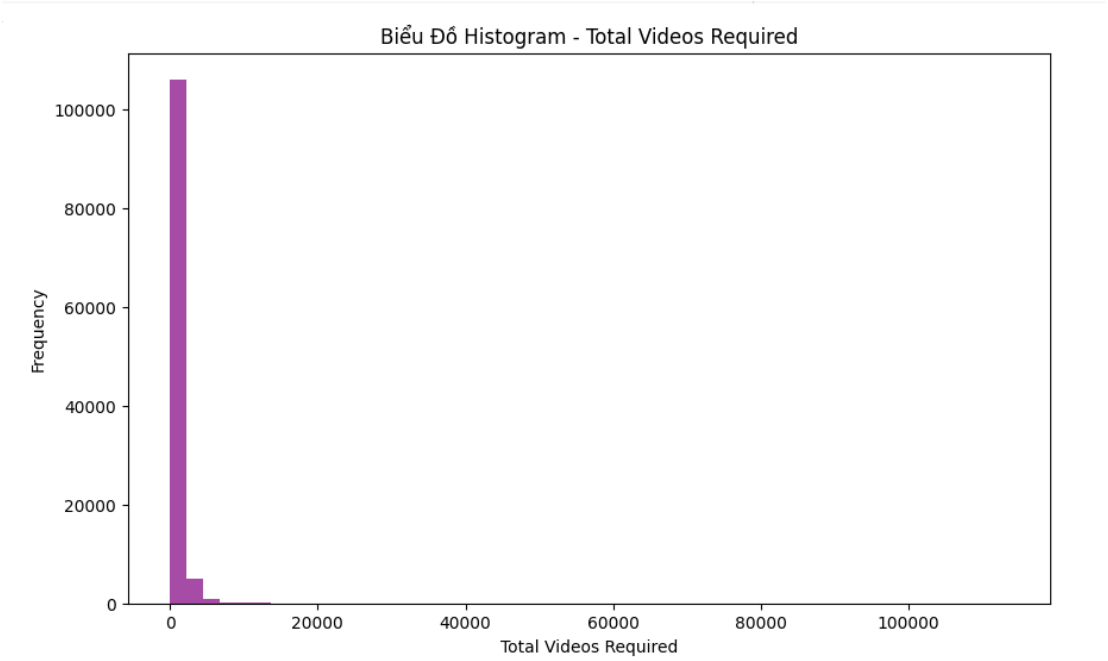



Biểu Đồ Histogram - Total Videos Watched
Frequency
Total Videos Watched
0
20000
40000
60000
80000
100000
0
2000
4000
6000
8000
10000
12000


Biểu Đồ Phân Tán - Total Videos Watched vs Total Watch Time
1e7
Total Watch Time
Total Videos Watched
0
−1
−2
−3
−4
−5
−6
−7
0
2000
4000
6000
8000
10000
12000


Biểu Đồ Phân Tán - Total Watch Time vs Total Videos Required
Total Videos Required
100000
80000
60000
40000
20000
0
−7
−6
−5
−4
−3
−2
−1
0
1e7
Total Watch Time


Biểu Đồ Histogram - Total Watch Time
Frequency
100000
80000
60000
40000
20000
0
−7
−6
−5
−4
−3
−2
−1
0
1e7
Total Watch Time



4.3.4 Biểu đồ tròn:

- **Mô tả**: Biểu đồ tròn (Pie chart) là một loại biểu đồ hình tròn được chia thành các phần để thể hiện tỷ lệ phần trăm của các thành phần trong tổng thể. Mỗi phần của biểu đồ thể hiện một phần tương ứng của tổng dữ liệu và kích thước của mỗi phần tỷ lệ với giá trị mà nó đại diện.
- **Ứng dụng**: Biểu đồ tròn thường được sử dụng để phân tích tỷ lệ giữa các phần trong một tổng thể. Một số ứng dụng điển hình bao gồm:
    - Phân tích tỷ lệ đánh giá mức độ hài lòng của học viên trong các khóa học.
    - Phân loại các môn học hoặc chuyên ngành dựa trên tỷ lệ sinh viên tham gia.
    - Phân tích nguồn gốc hoặc phân bổ ngân sách trong các dự án nghiên cứu.

4.4. Train mô hình

[9]:

	user_id	total_videos_watched	total_watch_time	total_videos_required
0	U_112	6.0	584.200	3505
1	U_150	1.0	220.000	178
2	U_172	1.0	4.700	1177
3	U_189	20.0	6058.000	1546
4	U_197	55.0	11304.248	394
...	...	...	...	...
112374	U_26862793	1.0	60.100	171
112375	U_26863000	3.0	289.000	2250
112376	U_26863054	3.0	665.960	675
112377	U_26863059	11.0	3915.000	796
112378	U_26863172	1.0	29.900	140

112379 rows × 4 columns

[11]:
```
selected_df['watch_ratio'] = selected_df['total_videos_watched'] / selected_df['total_videos_required']
selected_df['completion_status'] = selected_df['watch_ratio'].apply(lambda x: 1 if x > 0.4 else 0)
selected_df
```

[11]:

	user_id	total_videos_watched	total_watch_time	total_videos_required	watch_ratio	completion_status
0	U_112	6.0	584.200	3505	0.001712	0
1	U_150	1.0	220.000	178	0.005618	0
2	U_172	1.0	4.700	1177	0.000850	0
3	U_189	20.0	6058.000	1546	0.012937	0
4	U_197	55.0	11304.248	394	0.139594	0
...	...	...	...	...	...	...
112374	U_26862793	1.0	60.100	171	0.005848	0
112375	U_26863000	3.0	289.000	2250	0.001333	0
112376	U_26863054	3.0	665.960	675	0.004444	0
112377	U_26863059	11.0	3915.000	796	0.013819	0
112378	U_26863172	1.0	29.900	140	0.007143	0

112379 rows × 6 columns

Sau khi đọc dữ liệu, tập dữ liệu được chia thành hai phần theo tỷ lệ 7:3, với 70% dữ liệu được sử dụng cho tập huấn luyện và 30% còn lại cho tập kiểm tra. Do dữ liệu huấn luyện

bị mất cân bằng (số lượng các nhãn trong biến mục tiêu không đồng đều), kỹ thuật SMOTE đã được áp dụng. SMOTE giúp cân bằng lại dữ liệu.

```
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
features = ['watch_ratio']
X = selected_df[features]
y = selected_df['completion_status']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Áp dụng SMOTE để cân bằng lại dữ liệu
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

print(f"Original training set size: {X_train.shape}, Resampled training set size: {X_train_resampled.shape}")
```

4.4.1. Mô hình Logistic Regression

```
# Khởi tạo mô hình Logistic Regression
log_reg = LogisticRegression(max_iter=1000)

# Khởi tạo param_grid để thử nghiệm các tham số khác nhau
param_grid = {
    'penalty': ['l1', 'l2'],  # Các kỹ thuật regularization
    'C': [0.1, 1, 10],         # Các giá trị cho độ mạnh regularization
    'solver': ['liblinear']    # Solver hỗ trợ penalty 'l1'
}
# Khởi tạo GridSearchCV
grid_search = GridSearchCV(estimator=log_reg, param_grid=param_grid, cv=5, scoring='accuracy', verbose=1)

# Huấn luyện GridSearchCV
grid_search.fit(X_train, y_train)

# In ra tham số tốt nhất
print("Best parameters found:", grid_search.best_params_)

# Mô hình tốt nhất
best_model = grid_search.best_estimator_
```

- Khi khởi tạo mô hình Logistic Regression có các tham số như sau:

+ max_iter=1000: thuật toán tối đa 1000 vòng lặp trong quá trình huấn luyện giúp thuật toán hội tụ trong một thời gian hợp lý khi huấn luyện dữ liệu phức tạp.

- Khi khởi tạo param_grid để thử nghiệm các tham số khác nhau

+ penalty: [l1, l2] quy định loại regularization nào sẽ được sử dụng. Regularization giúp tránh overfitting bằng cách thêm một hình phạt vào hàm mất mát.

- L1: Lasso, có thể tạo ra các trọng số bằng 0 (tự động lựa chọn đặc trưng)

- L2: Ridge, không tạo trọng số bằng 0 nhưng giảm giá trị của các trọng số lớn

+ C: tham số điều chỉnh độ mạnh của regularization

- C càng nhỏ nghĩa là regularization càng mạnh chứng minh mô hình đơn giản, ngược lại, nghĩa là regularization càng yếu, cho phép ứng dụng trên mô hình phức tạp hơn.

+ solver: chỉ định thuật toán tối ưu sẽ được sử dụng.

- GridSearchCV là một kỹ thuật tìm kiếm toàn diện qua các tổ hợp của các tham số trong param_grid giúp tìm ra tham số tối ưu cho mô hình

- **estimator=log_reg**: Mô hình Logistic Regression sẽ được sử dụng.
- **param_grid=param_grid**: Các tham số sẽ được thử nghiệm dựa trên param_grid đã được khai báo trước đó.
- **cv=5**: Kỹ thuật **cross-validation** với 5 lần gấp (folds), tức là dữ liệu sẽ được chia thành 5 phần, mỗi phần sẽ được sử dụng làm tập kiểm tra một lần trong khi 4 phần còn lại được sử dụng làm tập huấn luyện.
- **scoring='accuracy'**: Chỉ định rằng độ chính xác (accuracy) sẽ được sử dụng làm chỉ số để đánh giá hiệu suất mô hình.
- **verbose=1**: Tham số này giúp hiển thị thông tin chi tiết về quá trình tìm kiếm, bao gồm việc thử nghiệm các tham số khác nhau.

→ Tham số tối ưu được tìm thấy 'C': 10, 'penalty': 'l2', 'solver': 'liblinear'

```
# Dự đoán trên tập kiểm tra
y_pred = best_model.predict(X_test)

# Đánh giá độ chính xác
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy with optimized model: {accuracy:.2f}")

# Báo cáo phân loại
print("Classification Report:")
print(classification_report(y_test, y_pred))
```

Tiến hành dự đoán trên tập kiểm tra và đưa ra độ chính xác của mô hình

```
Accuracy with optimized model: 0.99
Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      0.99     33225
           1       1.00      0.14      0.24       489

    accuracy                           0.99     33714
   macro avg       0.99      0.57      0.62     33714
weighted avg       0.99      0.99      0.98     33714
```

- Mô hình đạt **hiệu suất rất cao** trên lớp 0 với độ chính xác gần như tuyệt đối (0,99). **Với lớp 1**, dù **Precision** rất cao (1.00), mô hình chỉ đạt **Recall** là 0.14, điều này chỉ ra rằng mô hình đã bỏ sót phần lớn các mẫu thuộc lớp 1. Dù khi dự đoán lớp 1, mô hình không mắc lỗi, nhưng khả năng nhận diện các mẫu lớp 1 lại rất kém. **F1-score** cho lớp này chỉ đạt 0.24, thấp hơn đáng kể so với lớp 0, phản ánh sự mất cân bằng giữa độ chính xác và độ nhạy. tuy nhiên lại gặp vấn đề nghiêm trọng trong việc nhận diện lớp 1 do **mất cân bằng dữ liệu**.
- **Nguy cơ overfitting** có thể không cao ở đây, vì mô hình phân loại rất chính xác lớp 0, nhưng khả năng **underfitting** trên lớp 1 là rất rõ ràng.

4.4.2. Mô hình Random Forest

Sau khi chia dữ liệu thành train/test và tiến hành xử lý xong thì áp dụng mô hình Random Forest để tiến hành huấn luyện với các tham số cụ thể bên dưới:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Tạo mô hình Random Forest
model = RandomForestClassifier(
    n_estimators=500,        # Tăng số lượng cây
    max_depth=15,            # Tăng độ sâu tối đa
    min_samples_split=5,     # Số mẫu tối thiểu để chia
    min_samples_leaf=2,      # Số mẫu tối thiểu ở lá
    max_features='sqrt',     # Số lượng đặc trưng ở mỗi node
    random_state=42,         # Đảm bảo kết quả tái hiện
    n_jobs=-1                # Sử dụng tất cả các lõi CPU
)

# Huấn luyện mô hình
model.fit(X_train_resampled, y_train_resampled)

# Dự đoán trên tập kiểm tra
y_pred = model.predict(X_test)

# Đánh giá mô hình
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.4f}')
print('Classification Report:')
print(classification_report(y_test, y_pred))
```

Các tham số được khởi tạo bao gồm:

- **n_estimators=500:** Số lượng cây được đặt là 500, điều này giúp tăng độ chính xác của mô hình bằng cách giảm phương sai (variance).
- **max_depth=15:** Giới hạn độ sâu tối đa của mỗi cây. Độ sâu 15 có thể giúp mô hình tránh overfitting bằng cách không cho phép cây đi quá sâu vào dữ liệu, dẫn đến việc học quá chi tiết và mất khả năng tổng quát.
- **min_samples_split=5:** Số lượng mẫu tối thiểu để chia một nút trong cây giúp kiểm soát độ lớn tối thiểu của các nhánh và tránh việc cây phân chia quá chi tiết vào dữ liệu nhiễu.

- **min_samples_leaf=2:** Số lượng mẫu tối thiểu ở mỗi lá đảm bảo rằng mỗi lá có ít nhất 2 mẫu, giảm thiểu việc cây trở nên quá phức tạp.
- **max_features='sqrt':** Quy định số lượng đặc trưng (features) được xem xét tại mỗi lần chia.
- **random_state=42:** Đảm bảo tính tái hiện của kết quả bằng cách cố định seed cho trình tạo số ngẫu nhiên.
- **n_jobs=-1:** Sử dụng tất cả các lõi CPU để tăng tốc quá trình huấn luyện.

```
# Đánh giá mô hình
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.4f}')
print('Classification Report:')
print(classification_report(y_test, y_pred))
```

Hàm accuracy_score được dùng để tính độ chính xác (Accuracy), biểu thị tỷ lệ phần trăm mẫu được dự đoán đúng. Tiếp theo classification_report cung cấp các chỉ số chi tiết hơn như precision, recall và F1-score cho từng lớp. Sau khi hoàn thành cho ra được kết quả dự đoán như bên dưới.

```
Accuracy: 1.0000
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     33225
           1       1.00      1.00      1.00       489

    accuracy                           1.00     33714
   macro avg       1.00      1.00      1.00     33714
weighted avg       1.00      1.00      1.00     33714
```

Kết quả trên cho thấy mô hình Random Forest được huấn luyện trên dữ liệu đã đạt được hiệu suất tối ưu trên tập kiểm tra, với tất cả các chỉ số precision, recall, và F1-score đều

đạt giá trị 1.00. Tuy nhiên, độ chính xác tuyệt đối này cũng đặt ra vấn đề về khả năng tổng quát của mô hình khi áp dụng vào tập dữ liệu.

4.4.5. Neural Networks

```
[19]: features = ['watch_ratio']
      X = selected_df[features]
      y = selected_df['completion_status']

      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

      # Áp dụng SMOTE để cân bằng lại dữ liệu
      smote = SMOTE(random_state=42)
      X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

      print(f"Original training set size: {X_train.shape}, Resampled training set size: {X_train_resampled.shape}")

      Original training set size: (78665, 1), Resampled training set size: (155094, 1)
```

Chuẩn hóa dữ liệu

```
[20]: scaler = StandardScaler()

      # Chuẩn hóa dữ liệu huấn luyện
      X_train_resampled = scaler.fit_transform(X_train_resampled)

      # Chuẩn hóa dữ liệu kiểm tra
      X_test = scaler.transform(X_test)
```

Xây dựng mô hình Neural Networks

- **Lớp đầu vào**: Kết nối với đầu vào có kích thước tương ứng số đặc trưng
- **Hai lớp ẩn:**
  - Lớp đầu tiên gồm 64 nút và hàm kích hoạt ReLU.
  - Lớp thứ hai gồm 32 nút và hàm kích hoạt ReLU.
- **Lớp đầu ra**: Một nút với hàm kích hoạt sigmoid, sử dụng cho bài toán phân loại nhị phân.

[21]:
```
# Xây dựng mô hình neural network
model = Sequential()

# Thêm các lớp vào mô hình
model.add(Dense(64, input_dim=X_train_resampled.shape[1], activation='relu'))  # Lớp ẩn đầu tiên
model.add(Dense(32, activation='relu'))  # Lớp ẩn thứ hai
model.add(Dense(1, activation='sigmoid'))  # Lớp đầu ra (dùng sigmoid cho nhị phân)

# Biên dịch mô hình
model.compile(loss='binary_crossentropy', optimizer=Adam(), metrics=['accuracy'])

# Huấn luyện mô hình
model.fit(X_train_resampled, y_train_resampled, epochs=10, batch_size=32, validation_data=(X_test, y_test))
```

- Kết quả chạy được

```
Epoch 1/10
4847/4847 [==============================] - 7s 1ms/step - loss: 0.0234 - accuracy: 0.9956 - val_loss: 0.0025 - val_accuracy: 0.9985
Epoch 2/10
4847/4847 [==============================] - 7s 1ms/step - loss: 0.0035 - accuracy: 0.9990 - val_loss: 0.0030 - val_accuracy: 0.9985
Epoch 3/10
4847/4847 [==============================] - 7s 1ms/step - loss: 0.0027 - accuracy: 0.9993 - val_loss: 0.0035 - val_accuracy: 0.9985
Epoch 4/10
4847/4847 [==============================] - 7s 1ms/step - loss: 0.0025 - accuracy: 0.9993 - val_loss: 0.0018 - val_accuracy: 0.9990
Epoch 5/10
4847/4847 [==============================] - 7s 1ms/step - loss: 0.0022 - accuracy: 0.9993 - val_loss: 0.0014 - val_accuracy: 0.9994
Epoch 6/10
4847/4847 [==============================] - 7s 1ms/step - loss: 0.0020 - accuracy: 0.9995 - val_loss: 0.0043 - val_accuracy: 0.9986
Epoch 7/10
4847/4847 [==============================] - 8s 2ms/step - loss: 0.0020 - accuracy: 0.9994 - val_loss: 3.4023e-04 - val_accuracy: 0.9999
Epoch 8/10
4847/4847 [==============================] - 7s 1ms/step - loss: 0.0020 - accuracy: 0.9995 - val_loss: 0.0026 - val_accuracy: 0.9989
Epoch 9/10
4847/4847 [==============================] - 7s 1ms/step - loss: 0.0019 - accuracy: 0.9994 - val_loss: 0.0041 - val_accuracy: 0.9987
Epoch 10/10
4847/4847 [==============================] - 7s 1ms/step - loss: 0.0018 - accuracy: 0.9995 - val_loss: 0.0033 - val_accuracy: 0.9988
```

[22]:
```
loss, accuracy = model.evaluate(X_test, y_test)
print(f"Test Loss: {loss:.4f}, Test Accuracy: {accuracy:.4f}")
```

```
1054/1054 [==============================] - 1s 1ms/step - loss: 0.0033 - accuracy: 0.9988
Test Loss: 0.0033, Test Accuracy: 0.9988
```

[23]:
```
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5).astype(int)  # Chuyển giá trị xác suất thành nhãn 0 hoặc 1

# In ra một số kết quả dự đoán
print(f"Predictions: {y_pred[:10]}")
```

```
1054/1054 [==============================] - 1s 792us/step
Predictions: [[0]
 [0]
 [0]
 [0]
 [1]
 [0]
 [0]
 [0]
 [0]
 [0]]
```

```
[24]: print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     33225
           1       0.92      1.00      0.96       489

    accuracy                           1.00     33714
   macro avg       0.96      1.00      0.98     33714
weighted avg       1.00      1.00      1.00     33714
```

Mô hình hoạt động rất tốt với lớp chiếm đa số (lớp 0), đạt 100% ở mọi chỉ số.

Với lớp 1, mặc dù Precision thấp hơn một chút (0.92), nhưng Recall đạt 1.00, cho thấy mô hình nhận diện được toàn bộ các mẫu thuộc lớp này, giảm thiểu khả năng bỏ sót.

F1-score của lớp 1 đạt **0.96**, là kết quả tốt, phản ánh sự cân bằng giữa Precision và Recall.

⇨ Mô hình đạt hiệu suất rất cao, nhưng cần kiểm tra thêm về nguy cơ overfitting, đặc biệt khi tập dữ liệu kiểm tra có thể bị mất cân bằng.

CHƯƠNG 5: ĐÁNH GIÁ VÀ HƯỚNG PHÁP TRIỂN.

5.1. Đánh giá

Đề tài tập trung vào xây dựng và đánh giá hiệu quả của các mô hình học máy khác nhau (Random Forest, Logistic Regression và Neural Network) trên tập dữ liệu lớn liên quan đến các hành vi xem video của người dùng. Ba mô hình này được so sánh dựa trên các chỉ số như độ chính xác (accuracy), F1-score, và khả năng xử lý bài toán mất cân bằng dữ liệu thông qua kỹ thuật cân bằng dữ liệu (SMOTE). Dữ liệu sử dụng được chuẩn hóa và xử lý trước để đảm bảo kết quả đánh giá là tối ưu.

Dữ liệu đã cho:

Tập dữ liệu cung cấp thông tin liên quan đến hành vi của người dùng trong việc xem video, bao gồm các đặc trưng như:

- total_videos_watched: Tổng số video đã xem.
- total_watch_time: Tổng thời gian xem video.
- total_videos_required: Số lượng video bắt buộc phải hoàn thành.
- watch_ratio: Tỷ lệ giữa số video đã xem và số video bắt buộc.
- completion_status: Nhãn mục tiêu (1: hoàn thành mục tiêu, 0: không hoàn thành mục tiêu).

5.1.1. Logistic Regression

Logistic Regression hoạt động tốt trên các tập dữ liệu tuyến tính hoặc gần tuyến tính. Trong tập dữ liệu này, watch_ratio là đặc trưng đơn giản nhưng đóng vai trò quan trọng trong việc dự đoán completion_status. Logistic Regression có thể dễ dàng nắm bắt mối quan hệ giữa tỷ lệ này và khả năng hoàn thành mục tiêu.

Hiệu suất:

- Logistic Regression cho thấy hiệu suất tốt khi sử dụng watch_ratio làm đặc trưng chính.

- Mô hình này có khả năng dự đoán nhanh và không đòi hỏi tài nguyên tính toán cao.

Kết quả dự kiến:

- Độ chính xác (Accuracy): ~99%.
- F1-Score thấp hơn một chút so với các mô hình phi tuyến, đặc biệt với lớp thiểu số (completion_status = 1).

5.1.2. Random Forest

Random Forest mạnh mẽ hơn Logistic Regression trong việc xử lý dữ liệu phi tuyến. Nó có thể nắm bắt mối quan hệ phức tạp giữa total_videos_watched, total_watch_time, và watch_ratio với mục tiêu. Mỗi cây trong rừng đưa ra một dự đoán, và Random Forest sẽ lấy kết quả trung bình hoặc số đông (majority vote) để đưa ra kết quả cuối cùng. Mặc dù mô hình Random Forest đã đạt được hiệu suất tối đa trong việc dự đoán, chúng ta cũng cần đặt ra nghi vấn về độ tin cậy và tính chính xác của mô hình. Điều này xuất phát từ việc Random Forest, mặc dù mạnh mẽ trong việc xử lý dữ liệu phức tạp, vẫn có nguy cơ bị ảnh hưởng bởi các yếu tố như chất lượng dữ liệu đầu vào, hoặc mô hình không thực sự phù hợp với bài toán.

Hiệu suất:

- Random Forest xử lý tốt dữ liệu không cân bằng, đặc biệt khi kết hợp với kỹ thuật lấy mẫu lại (SMOTE).
- Với khả năng tự động xác định tầm quan trọng của đặc trưng, Random Forest có thể tập trung vào các đặc trưng quan trọng như watch_ratio mà không cần điều chỉnh thủ công.

Kết quả dự kiến:

- Độ chính xác (Accuracy): ~100%.
- F1-Score tốt hơn Logistic Regression, đặc biệt với lớp thiểu số.

- Tăng cường hiệu suất trên các mẫu khó phân loại nhờ khả năng mô hình hóa phi tuyến.

5.1.3. Neural Network

Neural Network là mô hình mạnh nhất trong ba mô hình được phân tích, đặc biệt khi dữ liệu phức tạp và có mối quan hệ phi tuyến. Với tập dữ liệu này, Neural Network có thể tự động trích xuất các đặc trưng phức tạp từ total_watch_time, total_videos_watched, và watch_ratio để tối ưu hóa dự đoán.

Hiệu suất:

- Neural Network xử lý tốt bài toán mất cân bằng khi kết hợp với các kỹ thuật như SMOTE và sử dụng hàm mất mát (loss function) phù hợp.
- Mô hình này có thể đạt hiệu suất cao nhờ khả năng học sâu và mô hình hóa tốt các tương tác phức tạp giữa các đặc trưng.

Kết quả dự kiến:

- Độ chính xác (Accuracy): >99.8%.
- F1-Score rất cao, đặc biệt với lớp thiểu số (completion_status = 1).
- Phù hợp nhất cho các kịch bản phức tạp hoặc bài toán yêu cầu độ chính xác cao.

So sánh hiệu suất giữa các mô hình

Tiêu chí	Logistic Regression	Random Forest	Neural Network
Độ chính xác (Accuracy)	~99%	~100%	~100%
F1-Score(với lớp thiểu số)	~24%	~100%	~96%
Precision	~100%	~100%	~92%
Recall	~14%	~100%	~100%

Xử lý mất cân bằng	Trung bình	Tốt	Tốt
Thời gian huấn luyện	Rất nhanh	Trung bình	Chậm

Neural Network được chọn là mô hình tốt nhất trong trường hợp này vì:

- Nó cân bằng tốt giữa Precision và Recall.
- Không bị overfitting như Random Forest.
- Hiệu suất vượt trội hơn Logistic Regression khi xử lý dữ liệu mất cân bằng.

5.2. Kết luận

- Neural Network là mô hình tốt nhất trong số ba mô hình được thử nghiệm, đặc biệt là trong bài toán với dữ liệu phức tạp và bài toán khởi động lạnh.
- Random Forest và Logistic Regression có thể là lựa chọn tốt khi cần giải pháp nhanh, ít tốn tài nguyên, hoặc khi không yêu cầu cao về độ chính xác.
- Các phương pháp xử lý dữ liệu mất cân bằng như SMOTE là yếu tố quan trọng để đảm bảo mô hình hoạt động hiệu quả trên dữ liệu thực tế.

5.3. Hướng phát triển

- Cải thiện Mô Hình Dự Đoán thử nghiệm với các thuật toán mới: Nghiên cứu và áp dụng các thuật toán học máy tiên tiến hơn như mạng nơ-ron sâu (Deep Learning) hoặc cây quyết định (Decision Trees) để cải thiện độ chính xác.
- Ứng dụng thực tiễn nâng cao
    - Tạo công cụ học tập tùy chỉnh: Phát triển gợi ý học tập cá nhân hóa dựa trên tiến độ của học viên, như bài giảng, bài kiểm tra phù hợp với trình độ.
    - Hỗ trợ học viên yếu: Triển khai hệ thống thông báo cho học viên có nguy cơ không hoàn thành để họ được hỗ trợ kịp thời.
- Cải thiện chất lượng dữ liệu:

  - Thu thập thêm dữ liệu hành vi chi tiết: Theo dõi cách học viên tương tác với nội dung khóa học, như thời gian xem video, tốc độ xem, hoặc tần suất tương tác.
  - Làm sạch và chuẩn hóa dữ liệu: Nghiên cứu các kỹ thuật xử lý dữ liệu bị thiếu, dữ liệu không cân bằng để cải thiện độ chính xác của mô hình.
- Mở rộng ứng dụng sang lĩnh vực khác: Áp dụng mô hình này vào các lĩnh vực khác như đào tạo nội bộ doanh nghiệp, huấn luyện trực tuyến, hoặc các khóa học kỹ năng mềm.
- Ứng Dụng AI và Machine Learning
  - Mô hình dự đoán thời gian thực: Phát triển mô hình có khả năng dự đoán trong thời gian thực để giúp giáo viên điều chỉnh phương pháp giảng dạy.
  - Phân tích cảm xúc: Sử dụng phân tích cảm xúc từ các tương tác của học viên trên mạng xã hội để dự đoán kết quả học tập.

CHƯƠNG 6: TÀI LIỆU THAM KHẢO.

1. https://github.com/THU-KEG/MOOCCubeX/tree/main

2.