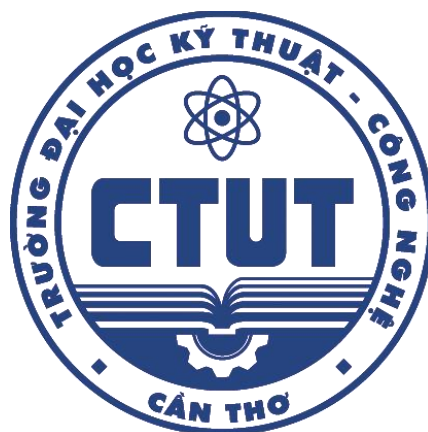


TRƯỜNG ĐẠI HỌC KỸ THUẬT – CÔNG NGHỆ CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN 3
XÂY DỰ MÔ HÌNH DỰ ĐOÁN KẾT QUẢ HỌC
TẬP CỦA SINH VIÊN

NGÀNH: KHOA HỌC DỮ LIỆU

GIẢNG VIÊN HƯỚNG DẪN: Lê Anh Nhã Uyên

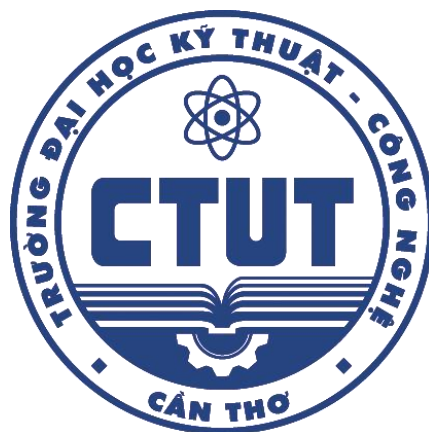
SINH VIÊN THỰC HIỆN:

Huỳnh Chí Phi Thuận

MSSV: KHDL2211038

Cần Thơ, tháng 8 năm 2025

TRƯỜNG ĐẠI HỌC KỸ THUẬT – CÔNG NGHỆ CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN 3
XÂY DỰNG MÔ HÌNH DỰ ĐOÁN KẾT QUẢ HỌC
TẬP CỦA SINH VIÊN

NGÀNH: KHOA HỌC DỮ LIỆU

GIẢNG VIÊN HƯỚNG DẪN: Lê Anh Nhã Uyên

SINH VIÊN THỰC HIỆN:

Huỳnh Chí Phi Thuận

MSSV: KHDL2211038

Cần Thơ, tháng 8 năm 2025

PHIẾU NHẬN XÉT GIẢNG VIÊN HƯỚNG DẪN

Tên đề tài: PHÂN TÍCH HIỆU XUẤT HỌC TẬP CỦA SINH VIÊN

Tên SV: Huỳnh Chí Phi Thuận **MSSV:** KHDL2211038

Họ và tên GVHD: Lê Anh Nhã Uyên

Nhận xét của giảng viên hướng dẫn:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Cần thơ, ngày.....tháng.....năm 2025

GIẢNG VIÊN HƯỚNG DẪN

Lê Anh Nhã Uyên

PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN

Giảng viên phản biện:

Nhận xét của giảng viên phản biện:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Cần Thơ, ngày.....tháng.....năm 2025

GIẢNG VIÊN PHẢN BIỆN

LỜI CẢM ƠN

xin gửi lời cảm ơn chân thành đến giảng viên Lê Anh Nhã Uyên Khoa Công nghệ thông tin đã trang bị cho những kiến thức, kỹ năng cần có để hoàn thành đề tài nghiên cứu này.

Tuy nhiên, trong quá trình nghiên cứu đề tài, do kiến thức chuyên ngành còn hạn chế nên vẫn còn nhiều thiếu sót khi tìm hiểu, đánh giá và trình bày về đề tài. Rất mong nhận được sự quan tâm, góp ý của giảng viên trong khoa để đề tài được đầy đủ và hoàn chỉnh hơn.

Xin chân thành cảm ơn.

TÓM TẮT ĐỒ ÁN

Đồ án này tập trung vào việc xây dựng hệ thống dự đoán hiệu suất học tập (điểm Grade) của sinh viên dựa trên nhiều yếu tố như điểm trung bình, tỷ lệ chuyên cần, điểm bài tập, bài kiểm tra, mức độ tham gia lớp, thời gian học hàng tuần, hoạt động ngoại khóa, áp lực học tập và thời gian ngủ.

Quy trình thực hiện gồm thu thập và làm sạch dữ liệu, phân tích đặc trưng, trích xuất các yếu tố quan trọng, sau đó huấn luyện nhiều mô hình Machine Learning như Random Forest, CatBoost, XGBoost để so sánh và chọn ra mô hình tối ưu nhất.

Ngoài ra, hệ thống được tích hợp giao diện trực quan bằng Streamlit, cho phép người dùng x biểu đồ, thống kê và dự đoán kết quả một cách dễ hiểu, sinh động.

Ứng dụng này hỗ trợ nhà trường và giảng viên hoặc chính sinh viên phát hiện sớm sinh viên có nguy cơ kết quả thấp hoặc bỏ học, từ đó đưa ra các biện pháp can thiệp, hỗ trợ cá nhân hóa, góp phần cải thiện chất lượng đào tạo, giảm tỷ lệ bỏ học và nâng cao trải nghiệm học tập của sinh viên.

MỤC LỤC

MỤC LỤC.....	i
DANH MỤC HÌNH.....	iv
CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI	1
1.1. Lý Do Chọn Đề Tài.....	1
1.2. Mục Tiêu Nghiên Cứu.....	1
1.3. Phương Pháp Nghiên Cứu.....	2
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT	3
2.1. Tìm Hiểu Về Máy Học.....	3
2.1.1. Máy học là gì ?.....	3
2.1.2. Lịch Sử Hình Thành Máy Học.....	4
2.1.3. Phân Loại Máy Học	6
2.1.4. Cách hoạt động của thuật toán trong máy học:.....	7
2.2. Thuật toán CatBoost.....	11
2.2.1. Các Lợi Ích Chính của CatBoost:	11
2.2.2. Cơ Chế Hoạt Động của CatBoost	11
2.2.3. Các Đặc Trưng Nổi Bật của CatBoost:	12
2.2.4. Tham Số và Siêu Tham Số (Hyperparameters) của CatBoost.....	13
2.2.5. Tinh chỉnh siêu tham số (Hyperparameter Tuning):	15
2.2.6. Ứng dụng của CatBoost:	15
2.2.7. Hạn chế của CatBoost:.....	16
2.2.8. So sánh CatBoost với các thuật toán Boosting khác:	17
2.2.9. Cải thiện mô hình:.....	17
CHƯƠNG 3: TIỀN XỬ LÝ VÀ TRỰC QUANG HÓA DỮ LIỆU SINH VIÊN	18
3.1. Dữ liệu sinh viên	18
3.2. Tiền xử lý dữ liệu	22
3.2.1. Kiểm tra và làm sạch dữ liệu	22
3.2.2. Kiểm tra giá trị duy nhất của các thuộc tính	23

3.3. Trục quang hóa dữ liệu.....	24
3.3.1. Phân tích phân bố điểm chữ của sinh viên.....	25
3.3.2. Phân tích các yếu tố ảnh hưởng đến học tập.....	26
3.3.3. Phân tích hoạt động ảnh hưởng đến tổng điểm.....	28
3.3.4. Phân tích phân bố điểm số theo khoa	29
3.3.5. Phân tích tổng quan với biểu đồ nhiệt	30
3.3.6. Kết luận chung qua quá trình phân tích	32
CHƯƠNG 4: XÂY DỰNG MÔ HÌNH CATBOOST DỰ ĐOÁN HIỆU	
XUẤT HỌC TẬP SINH VIÊN.....	34
4.1. Lý do chọn mô hình CatBoost	34
4.1.1. Khả năng xử lý tự động các tính năng phân loại	34
4.1.2. Giảm thiểu hiện tượng Overfitting.....	35
4.1.3. Hiệu suất cao và khả năng mở rộng.....	35
4.1.4. Xử lý giá trị bị thiếu (Missing Values).....	36
4.1.5. Chiến lược phân tách cây (Tree Splitting Strategy).....	36
4.1.6. So sánh hiệu suất thực tế.....	36
4.2. Mục tiêu mô hình	37
4.3. Chuẩn bị dữ liệu	37
4.4. Mô hình CatBoost chưa tinh chỉnh (Baseline).....	38
4.4.2. Ý nghĩa các tham số mặc định quan trọng.....	38
4.4.3. Phân tích kết quả	39
4.4.4. Nhận xét tổng quan	39
4.5. Tinh chỉnh siêu tham số bằng RandomizedSearchCV	40
4.5.1. Phương pháp	40
4.5.2. Các tham số tinh chỉnh.....	41
4.5.3. Kết quả mô hình tối ưu	42
CHƯƠNG 5: KẾT LUẬN	43
5.1. Kết quả đạt được đánh giá đạt được	43
5.2. Ứng Dụng và Tinh Chỉnh.....	44

5.2.1. Ứng dụng trong giáo dục (thực tế).....	44
5.2.2. Tinh chỉnh mô hình	45
5.3. Hướng Phát Triển	46
5.4. Hạn Chế.....	46
TÀI LIỆU KHAM KHẢO.....	47

DANH MỤC HÌNH

Hình 2.1: Học máy (Machine Learning)	3
Hình 2.2: Quá trình phát triển của máy học	5
Hình 2.3: Phân loại trong máy học	7
Hình 2.4: Quá trình hoạt động của máy học	10
Hình 3.1: điền giá trị bị thi	22
Hình 3.2: Kiểm tra các thuộc tính là duy nhất.....	23
Hình 3.3: Biểu đồ cột phân bố điểm	25
Hình 3.4: Mối quan hệ giữa các yếu tố học tập	26
Hình 3.5: Hoạt động ảnh hưởng đến tổng điểm	28
Hình 3.6: Phân tích phân bố điểm theo khoa	29
Hình 3.7: Biểu đồ nhiệt	30
Hình 4.1: Chuẩn bị dữ liệu	37
Hình 4.2: Mô hình CatBoos chưa tinh chỉnh tham số	38
Hình 4.3: Tinh chỉnh siêu tham số với RandomizedSearchCV	41
Hình 5.1: Giao diện streamlit giúp người dùng nhập thông tin	43
Hình 5.2: Kết quả đạt được.....	43

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

1.1. Lý Do Chọn Đề Tài

Hiệu suất kết quả học tập của sinh viên là một trong những yếu tố then chốt phản ánh chất lượng đào tạo của một cơ sở giáo dục. Trong bối cảnh giáo dục hiện đại, việc ứng dụng công nghệ và trí tuệ nhân tạo để hỗ trợ quản lý, giảng dạy và học tập ngày càng trở nên cần thiết. Tuy nhiên, nhiều sinh viên vẫn đang gặp khó khăn trong quá trình học tập mà nhà trường chưa thể phát hiện và hỗ trợ kịp thời.

Với sự phát triển mạnh mẽ của khoa học dữ liệu và học máy, việc khai thác dữ liệu học tập để xây dựng các mô hình dự đoán hiệu suất học tập có thể giúp nhà trường phát hiện sớm những sinh viên có nguy cơ tụt hậu hoặc bỏ học. Điều này giúp đưa ra các biện pháp hỗ trợ phù hợp, từ đó nâng cao chất lượng đào tạo và giảm tỷ lệ sinh viên bỏ học.

Đề tài này không chỉ có ý nghĩa thực tiễn cao trong quản lý giáo dục mà còn giúp nâng cao kiến thức chuyên môn về xử lý dữ liệu, học máy và xây dựng ứng dụng thực tế. Vì vậy, chọn thực hiện đề tài “Xây dựng mô hình dự đoán hiệu suất học tập của sinh viên” nhằm giải quyết một vấn đề thiết thực trong lĩnh vực giáo dục hiện nay.

1.2. Mục Tiêu Nghiên Cứu

Xây dựng hệ thống mô hình học máy có khả năng dự đoán hiệu suất học tập của sinh viên dựa trên các đặc trưng như điểm số, mức độ chuyên cần, áp lực học tập, thời gian ngủ, và các yếu tố liên quan khác.

Phân tích, xử lý và trực quan hóa dữ liệu sinh viên nhằm hiểu rõ mối liên hệ giữa các đặc trưng đầu vào với kết quả học tập.

So sánh và đánh giá hiệu quả giữa các mô hình như Random Forest, CatBoost, XGBoost,... để chọn ra mô hình tối ưu nhất về độ chính xác và hiệu năng.

Xây dựng giao diện ứng dụng dự đoán thân thiện, cho phép người dùng nhập dữ liệu đầu vào và nhận kết quả dự đoán ngay lập tức.

Hỗ trợ nhà trường và cố vấn học tập trong việc phát hiện sớm sinh viên có nguy cơ gặp khó khăn học tập để có biện pháp can thiệp kịp thời.

1.3. Phương Pháp Nghiên Cứu

Phương pháp thu thập dữ liệu: Thu thập bộ dữ liệu về sinh viên bao gồm điểm số, thời gian học, mức độ chuyên cần, áp lực học tập, thời gian ngủ, mức thu nhập của gia đình, mức độ căng thẳng v.v... từ các nguồn giả lập hoặc khảo sát thực tế.

Phương pháp tiền xử lý dữ liệu (Data Preprocessing): Thực hiện làm sạch dữ liệu (loại bỏ dữ liệu thiếu, nhiễu), chuẩn hóa dữ liệu, mã hóa các biến phân loại và chia tập dữ liệu thành tập huấn luyện và tập kiểm tra.

Phương pháp phân tích thống kê: Áp dụng các kỹ thuật thống kê mô tả và trực quan hóa dữ liệu để khám phá mối quan hệ giữa các biến đầu vào và kết quả học tập.

Phương pháp xây dựng mô hình học máy (Machine Learning): Sử dụng các thuật toán như Random Forest, CatBoost, và XGBoost để xây dựng mô hình dự đoán hiệu suất học tập của sinh viên.

Phương pháp đánh giá mô hình: Sử dụng các chỉ số như độ chính xác (accuracy), F1-score, confusion matrix... để đánh giá và so sánh hiệu quả các mô hình.

Phương pháp triển khai ứng dụng: Tích hợp mô hình dự đoán vào giao diện ứng dụng sử dụng thư viện Streamlit, giúp người dùng tương tác và sử dụng trực tiếp mô hình.

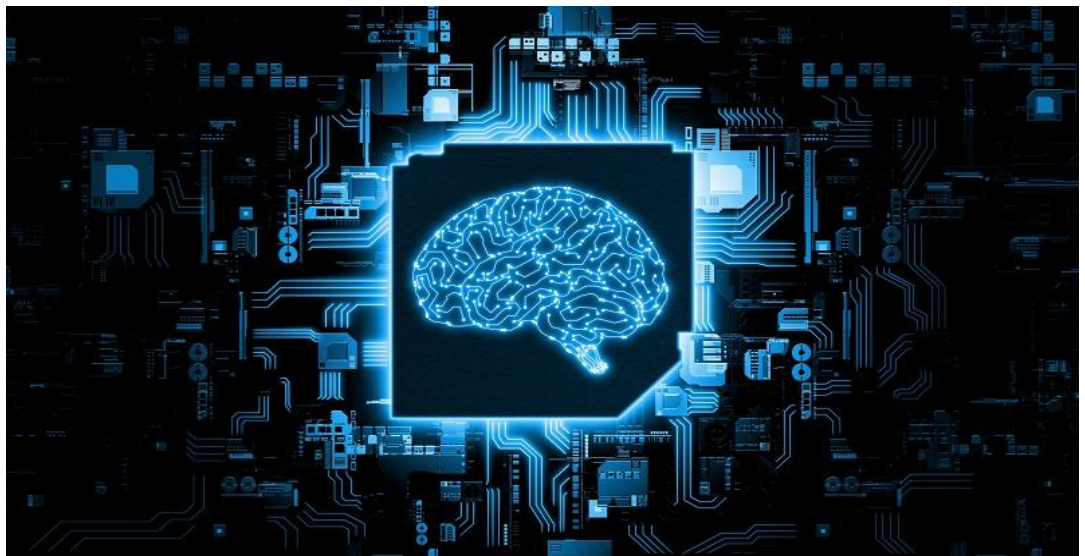
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Tìm Hiểu Về Máy Học

2.1.1. Máy học là gì ?

Máy học (machine learning) là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kỹ thuật cho phép các hệ thống “học” tự động từ dữ liệu để giải quyết những vấn đề cụ thể. Theo định nghĩa trong cuốn sách "Machine Learning" của Tom Mitchell (1997), máy học được mô tả như sau:

“A computer program is said to learn to perform a task T from experience E, if its performance at task T, as measured by a performance metric P, improves with experience E over time.”



Hình 2.1: Học máy (Machine Learning)

Máy học là một thành phần quan trọng của lĩnh vực khoa học dữ liệu đang phát triển. Thông qua việc sử dụng phương pháp thống kê, các thuật toán được đào tạo để phân loại hoặc dự đoán và khám phá những thông tin chi tiết trong các dự án khai thác dữ liệu. [1]

Những thông tin chi tiết này hỗ trợ, thúc đẩy việc đưa ra quyết định trong các ứng dụng, công cụ hỗ trợ doanh nghiệp, người dùng. Khi khối lượng dữ liệu

tiếp tục mở rộng và phát triển, khả năng dự đoán, phân tích chính xác của máy học sẽ tăng lên.

2.1.2. Lịch Sử Hình Thành Máy Học

Lịch sử hình thành và phát triển của máy học (machine learning) có thể được chia thành nhiều giai đoạn khác nhau:

1950s - 1960s: Khởi đầu

1950: Alan Turing đề xuất "Turing Test" trong bài viết "Computing Machinery and Intelligence," đặt nền móng cho việc nghiên cứu trí tuệ nhân tạo (AI).

1952: Arthur Samuel phát triển chương trình máy tính có khả năng chơi cờ vua và tự học từ kinh nghiệm.

1957: Frank Rosenblatt phát triển Perceptron, mô hình mạng neuron đơn giản, đánh dấu bước tiến đầu tiên trong nghiên cứu mạng neuron.

1970s - 1980s: Thời kỳ khó khăn

1970s: AI gặp phải "Mùa đông AI" khi những kỳ vọng quá cao không được đáp ứng, dẫn đến giảm tài trợ và nghiên cứu.

1980: John Hopfield và David Rumelhart tái phát triển mạng neuron bằng cách giới thiệu mô hình Hopfield và thuật toán học ngược.

1990s: Thời kỳ phục hưng

1990s: Máy học bắt đầu phát triển mạnh mẽ trở lại với sự gia tăng của dữ liệu và sức mạnh tính toán.

1995: Vladimir Vapnik và Alexey Chervonenkis phát triển Support Vector Machines (SVM), một phương pháp phân loại mạnh mẽ.

2000s: Phát triển nhanh chóng

2006: Geoffrey Hinton và các cộng sự giới thiệu thuật toán Deep Belief Networks, đặt nền móng cho sự phát triển của deep learning.

2010: Sự gia tăng của dữ liệu lớn (big data) và sự phát triển của GPU giúp deep learning và các phương pháp máy học khác phát triển mạnh mẽ.

2010s - Hiện tại: Kỷ nguyên AI, phát triển và ứng dụng

2012: AlexNet, một mô hình deep learning, chiến thắng cuộc thi ImageNet, chứng minh sức mạnh của mạng neuron sâu.

2014: Google DeepMind giới thiệu AlphaGo, một hệ thống AI có khả năng chơi cờ vây vượt qua người chơi chuyên nghiệp.

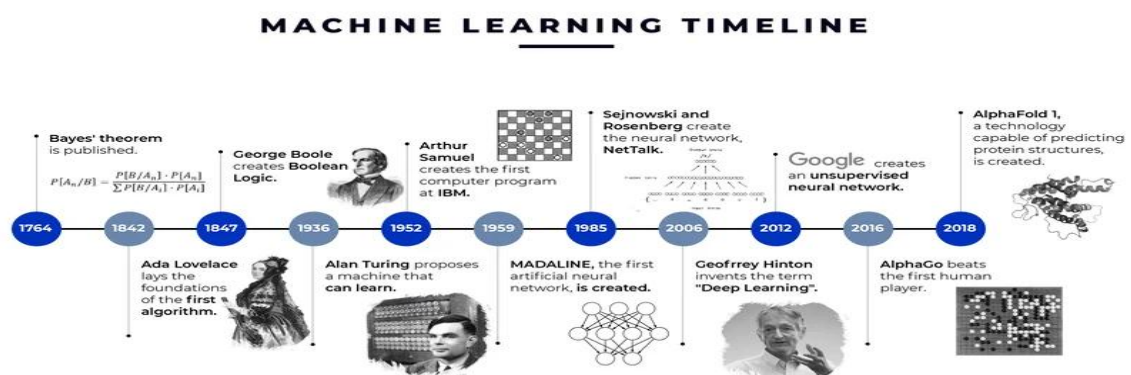
2016: AI được ứng dụng rộng rãi trong nhiều lĩnh vực như y tế, tài chính, và ô tô tự lái.

2017: Sự ra đời của Transformer (trong bài báo "Attention is All You Need" của Vaswani et al.), một mô hình mạng neuron mới đã cách mạng hóa nhiều ứng dụng trong xử lý ngôn ngữ tự nhiên (NLP).

2018: GPT-2 và sau đó là GPT-3 của OpenAI, mô hình ngôn ngữ lớn có khả năng tạo ra văn bản tự nhiên với độ phức tạp cao, đã gây ấn tượng mạnh.

2019: BERT (Bidirectional Encoder Representations from Transformers) của Google, một mô hình NLP mạnh mẽ, đã cải thiện nhiều nhiệm vụ xử lý ngôn ngữ tự nhiên.

2020: Sự phát triển của AI tự học không giám sát và học tăng cường, như AlphaFold của DeepMind, giải quyết vấn đề gấp cuộn protein, một trong những thách thức lớn nhất trong sinh học.



Hình 2.2: Quá trình phát triển của máy học

2.1.3. Phân Loại Máy Học

Học có giám sát (Supervised Learning):

Học có giám sát là một trong những phương pháp phổ biến nhất trong máy học, nơi mà mô hình học từ một tập dữ liệu đã được gán nhãn. Tập dữ liệu huấn luyện gồm các cặp đầu vào (input) và đầu ra (output) mong muốn. Mục tiêu của mô hình là học cách ánh xạ từ đầu vào đến đầu ra để có thể dự đoán chính xác đầu ra cho các dữ liệu mới chưa thấy trước đó. Trong quá trình học, mô hình điều chỉnh các thông số của nó dựa trên sự khác biệt giữa dự đoán và thực tế, nhằm giảm thiểu lỗi dự đoán.

Học không giám sát (Unsupervised Learning):

Học không giám sát là một kỹ thuật trong máy học mà không có nhãn đầu ra đi kèm với dữ liệu đầu vào. Thay vào đó, mục tiêu của mô hình là khám phá các cấu trúc, mẫu hoặc mối quan hệ ẩn trong dữ liệu. Điều này giúp hiểu rõ hơn về dữ liệu và có thể được sử dụng để phát hiện các nhóm (clusters), giảm chiều dữ liệu hoặc phát hiện các ngoại lệ (anomalies).

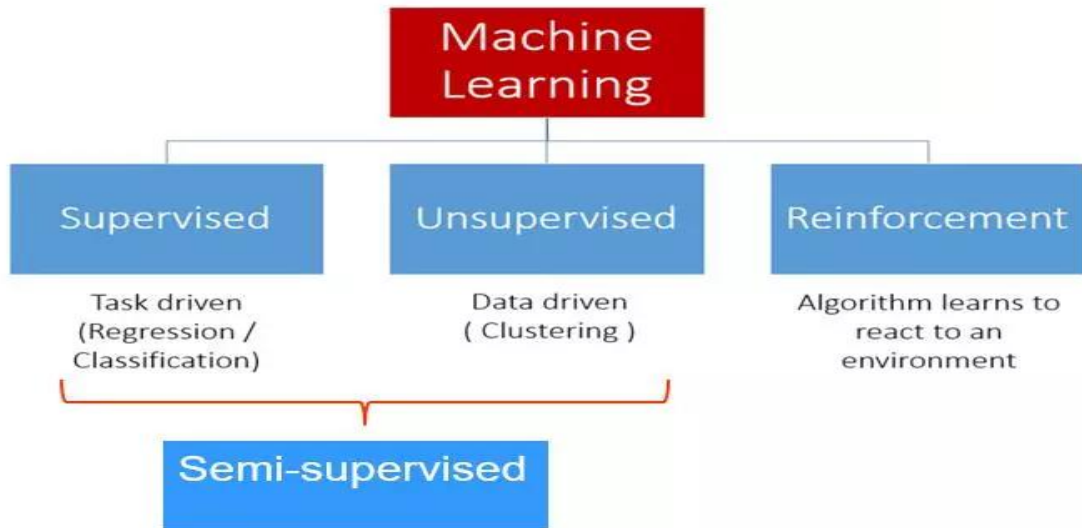
Học tăng cường (Reinforcent Learning):

Học tăng cường là một kỹ thuật trong máy học mà một tác nhân (agent) học cách tương tác với môi trường để tối đa hóa phần thưởng tích lũy theo thời gian. Môi trường đưa ra phản hồi dưới dạng phần thưởng hoặc hình phạt cho các hành động của tác nhân. Tác nhân sử dụng phản hồi này để điều chỉnh chính sách hành động của mình sao cho đạt được phần thưởng tối đa.

Học bán giám sát (Si-supervised Learning):

Học bán giám sát là một phương pháp kết hợp giữa học có giám sát và học không giám sát. Mô hình được đào tạo trên một tập dữ liệu có cả nhãn và không nhãn. Điều này rất hữu ích khi việc gán nhãn dữ liệu tốn kém hoặc khó khăn, nhưng có nhiều dữ liệu không nhãn sẵn có. Học bán giám sát sử dụng dữ liệu không nhãn để cải thiện hiệu suất của mô hình hơn so với chỉ sử dụng dữ liệu có nhãn.

Types of Machine Learning



Hình 2.3: Phân loại trong máy học

2.1.4. Cách hoạt động của thuật toán trong máy học:

a. Thu thập dữ liệu (Gathering Data / Data Collection)

Đây là bước đầu tiên và quan trọng nhất trong quy trình học máy. Trong bước này, cần thu thập dữ liệu từ các nguồn khác nhau mà mô hình của sẽ được huấn luyện. Dữ liệu có thể được thu thập từ nhiều nguồn như cơ sở dữ liệu, cảm biến, dịch vụ web, khảo sát, hoặc thậm chí từ các tệp tin đã có sẵn. Mục tiêu của bước này là đảm bảo có đủ dữ liệu chất lượng để huấn luyện mô hình.

b. Tiền xử lý dữ liệu (Data Preprocessing)

Tiền xử lý dữ liệu là một bước quan trọng để chuẩn bị dữ liệu cho mô hình học máy. Quá trình này bao gồm nhiều công việc khác nhau:

Trích xuất dữ liệu (Data Extraction): Lấy dữ liệu từ các nguồn thu thập và đưa vào định dạng dễ xử lý.

Làm sạch dữ liệu (Data Cleaning): Xóa bỏ hoặc sửa chữa các lỗi trong dữ liệu, chẳng hạn như dữ liệu bị thiếu, dữ liệu không chính xác, hoặc dữ liệu bị trùng lặp.

Chuyển đổi dữ liệu (Data Transformation): Chuyển đổi dữ liệu từ định dạng này sang định dạng khác để phù hợp với yêu cầu của mô hình, ví dụ như chuyển đổi dữ liệu văn bản thành số hoặc thay đổi định dạng thời gian.

Chuẩn hóa dữ liệu (Data Normalization): Đưa các thuộc tính dữ liệu về cùng một phạm vi giá trị để giúp mô hình học dễ dàng hơn. Ví dụ, chuẩn hóa các giá trị về khoảng từ 0 đến 1.

Trích xuất đặc trưng (Feature Extraction): Xác định và chọn lựa các đặc trưng (features) quan trọng từ dữ liệu gốc mà mô hình sẽ sử dụng. Đây là bước quan trọng để cải thiện hiệu suất của mô hình bằng cách loại bỏ các đặc trưng không cần thiết hoặc không liên quan.

c. Phân tích dữ liệu (Data Analysis)

Trong bước này, thực hiện các phân tích sơ bộ để hiểu rõ hơn về dữ liệu của mình. Điều này có thể bao gồm việc khám phá các mối quan hệ giữa các biến, phân phối của dữ liệu, và các mẫu hoặc xu hướng có thể có trong dữ liệu. Phân tích dữ liệu giúp xác định các vấn đề tiềm ẩn và hướng dẫn việc chọn lựa các thuật toán và kỹ thuật phù hợp cho việc xây dựng mô hình.

d. Xây dựng mô hình máy học (Model Building)

Sau khi dữ liệu đã được chuẩn bị và phân tích, tiến hành xây dựng mô hình học máy. Trong bước này, chọn lựa và thiết kế các thuật toán học máy phù hợp với bài toán của mình. Có nhiều loại mô hình khác nhau như hồi quy tuyến tính, cây quyết định, mạng nơ-ron, và máy vector hỗ trợ (SVM). Mô hình được xây dựng dựa trên các đặc trưng đã chọn và cấu hình các tham số của mô hình.

e. Huấn luyện mô hình (Model Training)

Huấn luyện mô hình là bước quan trọng để mô hình học máy học từ dữ liệu. Trong bước này, dữ liệu huấn luyện được sử dụng để điều chỉnh các tham số của mô hình sao cho mô hình có thể dự đoán chính xác các kết quả. sẽ chia dữ liệu thành hai tập: tập huấn luyện và tập kiểm tra. Tập huấn luyện được dùng để cập nhật các tham số của mô hình, trong khi tập kiểm tra dùng để đánh giá hiệu suất của mô hình trong quá trình huấn luyện.

f. Đánh giá mô hình (Model Evaluation)

Sau khi mô hình đã được huấn luyện, cần đánh giá hiệu suất của nó để đảm bảo rằng nó hoạt động tốt và có thể dự đoán chính xác trên dữ liệu mới. Đánh giá mô hình thường được thực hiện bằng cách sử dụng các chỉ số hiệu suất như độ chính xác, độ nhạy, độ đặc hiệu. cũng có thể sử dụng các kỹ thuật như kiểm tra chéo để đảm bảo rằng mô hình không bị overfit với dữ liệu huấn luyện và có thể hoạt động tốt với dữ liệu chưa thấy.

g. Các chỉ số đánh giá chính

Accuracy (Độ chính xác): Tỷ lệ dự đoán đúng trên tổng số mẫu. Đây là chỉ số phổ biến nhưng không phản ánh toàn diện nếu dữ liệu mất cân bằng (ví dụ: đa số sinh viên đều có kết quả tốt).

Precision (Độ chính xác theo lớp): Tỷ lệ dự đoán đúng trong số những mẫu mà mô hình dự đoán là "sinh viên kém".

Recall (Độ nhạy/Sensitivity): Tỷ lệ phát hiện đúng những sinh viên thực sự kém trong tất cả những sinh viên kém thực tế. Chỉ số này đặc biệt quan trọng nếu mục tiêu là phát hiện sớm để can thiệp.

F1-score: Trung bình điều hòa giữa Precision và Recall, giúp cân bằng hai yếu tố trên.

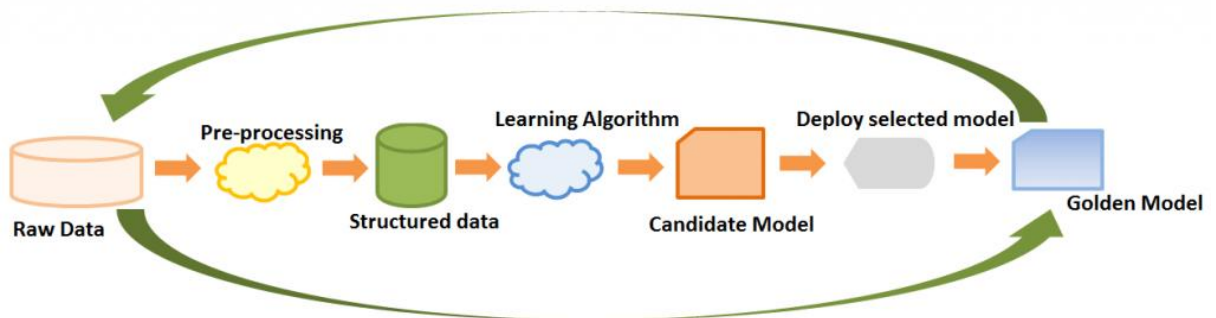
Kỹ thuật kiểm tra chéo (Cross-validation)

Để đảm bảo rằng mô hình không chỉ hoạt động tốt trên dữ liệu huấn luyện mà còn tổng quát hóa tốt trên dữ liệu mới, nhóm thực hiện k-fold cross-validation. Kỹ thuật này giúp chia tập dữ liệu thành nhiều phần và hoán đổi luân phiên giữa các phần để huấn luyện và kiểm tra mô hình, từ đó giảm thiểu rủi ro của kết quả đánh giá bị lệch do chia tập dữ liệu không đồng đều.

So sánh mô hình và lựa chọn mô hình tối ưu

Sau khi đánh giá nhiều mô hình như Random Forest, XGBoost, CatBoost,... trên cùng một bộ tiêu chí, nhóm tiến hành so sánh hiệu suất giữa các mô hình. Việc lựa chọn mô hình cuối cùng không chỉ dựa trên chỉ số chính xác cao nhất, mà còn cân nhắc đến tốc độ xử lý, khả năng diễn giải (interpretability), và khả năng mở rộng cho các ứng dụng thực tế như tích hợp vào giao diện web.

Trực quan hóa kết quả: Cuối cùng, nhóm sử dụng các biểu đồ như biểu đồ cột (bar chart), biểu đồ đường (line chart), và confusion matrix dạng heatmap để trình bày trực quan hiệu suất mô hình. Điều này giúp dễ dàng thuyết minh kết quả với người dùng không chuyên và nâng cao tính minh bạch trong mô hình dự đoán.



Hình 2.4: Quá trình hoạt động của máy học

2.2. Thuật toán CatBoost

CatBoost là một bộ công cụ mã nguồn mở được phát triển bởi Yandex, rất phổ biến cho kỹ thuật **gradient boosting trên cây quyết định**. Nó có thể được áp dụng để giải quyết nhiều vấn đề học máy khác nhau, bao gồm phân loại (classification), hồi quy (regression), xếp hạng (ranking), và nhiều hơn nữa. [1]

2.2.1. Các Lợi Ích Chính của CatBoost:

Xử lý tự động các đặc trưng phân loại (categorical features): CatBoost có khả năng xử lý các đặc trưng phân loại một cách tự động mà không cần mã hóa hay tiền xử lý. Điều này giúp tránh các vấn đề như ma trận thưa thớt (sparse matrix) hoặc hiện tượng quá khớp (overfitting) thường gặp khi sử dụng các kỹ thuật mã hóa truyền thống như One-Hot Encoding.

Giảm thiểu quá khớp: Nó sử dụng một lược đồ gradient boosting mới lạ và các kỹ thuật điều chuẩn (regularization) để giảm quá khớp. CatBoost sử dụng các kỹ thuật thông minh như boosting có thứ tự (ordered boosting), kết hợp đặc trưng ngẫu nhiên (random feature combinations), và các phương pháp boosting mạnh mẽ để giúp mô hình hoạt động tốt ngay cả trên dữ liệu mới, chưa từng thấy.

Hiệu suất cao và khả năng mở rộng: CatBoost đạt được hiệu suất cao và khả năng mở rộng nhờ các triển khai hiệu quả cho CPU và GPU. Nó hỗ trợ huấn luyện tăng tốc bằng GPU, giúp đẩy nhanh quá trình xây dựng mô hình, đặc biệt với các bộ dữ liệu lớn. Thư viện này cũng tận dụng các kỹ thuật xử lý song song để sử dụng nhiều lõi CPU trong quá trình huấn luyện, làm cho quá trình này hiệu quả và có khả năng mở rộng.

2.2.2. Cơ Chế Hoạt Động của CatBoost

CatBoost hoạt động dựa trên kỹ thuật **gradient boosting**. Quy trình này xây dựng các cây quyết định một cách tuần tự để giảm thiểu sai số và cải thiện dự đoán.

Xây dựng cây quyết định đầu tiên: Một cây quyết định được xây dựng và đánh giá mức độ sai số trong các dự đoán của nó.

Sửa lỗi từ cây trước: Cây tiếp theo được tạo ra để sửa các lỗi mà cây trước đó đã mắc phải.

Quy trình lặp lại: Quá trình này tiếp tục lặp đi lặp lại, với mỗi cây mới tập trung vào việc cải thiện dự đoán của mô hình bằng cách giảm các sai số trước đó, cho đến khi đạt đến số lần lặp đã định trước. Kết quả là một tập hợp các cây quyết định hoạt động cùng nhau để đưa ra các dự đoán chính xác.

CatBoost đặc biệt phù hợp với các bộ dữ liệu lớn có nhiều đặc trưng độc lập. Không giống như các thuật toán gradient boosting khác, CatBoost được thiết kế đặc biệt để xử lý cả các đặc trưng phân loại và số một cách liền mạch mà không yêu cầu mã hóa đặc trưng thủ công. Nó cũng sử dụng thuật toán **Symmetric Weighted Quantile Sketch (SWQS)** để xử lý các giá trị thiếu (missing values), giảm quá khớp và cải thiện hiệu suất mô hình.

2.2.3. Các Đặc Trưng Nổi Bật của CatBoost:

Xử lý đặc trưng phân loại: CatBoost tự động chuyển đổi các đặc trưng phân loại thành đặc trưng số một cách hiệu quả, sử dụng các chiến lược mã hóa mục tiêu (target encoding) và mã hóa một nóng (one-hot encoding) nội bộ.

Xử lý giá trị thiếu: CatBoost có thể xử lý các giá trị thiếu trong dữ liệu đầu vào mà không cần phương pháp điền khuyết (imputation). Thuật toán SWQS giúp xử lý dữ liệu thiếu hiệu quả, giảm quá khớp và cải thiện hiệu suất.

Huấn luyện và phân tích mô hình:

- Hỗ trợ huấn luyện tăng tốc GPU, giúp tăng tốc quá trình xây dựng mô hình, đặc biệt với bộ dữ liệu lớn.
- Sử dụng các kỹ thuật xử lý song song để tận dụng nhiều lõi CPU trong quá trình huấn luyện, làm cho quá trình này hiệu quả và có khả năng mở rộng.

Các chỉ số CatBoost (Metrics): CatBoost sử dụng các chỉ số để kiểm tra hiệu suất của mô hình, bao gồm độ chính xác (accuracy), độ chuẩn xác

(precision), độ thu hồi (recall), F1-score, ROC-AUC cho phân loại, và RMSE cho hồi quy.

2.2.4. Tham Số và Siêu Tham Số (Hyperparameters) của CatBoost

Việc hiểu và tinh chỉnh các tham số và siêu tham số là rất quan trọng để tối ưu hóa hiệu suất của mô hình CatBoost.

- **Tham số CatBoost:** Đây là các cài đặt nội bộ của mô hình mà nó học được trong quá trình huấn luyện. Một số tham số quan trọng:
- **iterations:** Chỉ định số lần lặp (số cây) được sử dụng trong quá trình huấn luyện.
- **learning_rate:** Kiểm soát kích thước bước ở mỗi lần lặp khi di chuyển về phía cực tiểu của hàm mất mát.
- **depth:** Xác định độ sâu tối đa của các cây quyết định riêng lẻ trong tập hợp.
- **l2_leaf_reg:** Thuật ngữ điều chuẩn ngăn chặn quá khớp bằng cách phạt các giá trị tham số lớn.
- **cat_features:** Một mảng các chỉ số cho biết đặc trưng nào là phân loại.
- **loss_function:** Chỉ định hàm mất mát cần tối ưu hóa trong quá trình huấn luyện. Ví dụ: 'RMSE' cho hồi quy, 'Logloss' hoặc 'MultiClass' cho phân loại.
- **Siêu tham số CatBoost (Hyperparameters):** [2]

Đây là các tham số mà phải cung cấp trước khi huấn luyện mô hình. Chúng kiểm soát các biến liên quan đến huấn luyện. Việc lựa chọn siêu tham số phù hợp là rất cần thiết để mô hình hoạt động tốt. CatBoost cung cấp một giao diện linh hoạt để cấu hình và tinh chỉnh siêu tham số, được chia thành nhiều loại:

- **Siêu tham số chung (Common hyperparameters):** Áp dụng cho bất kỳ vấn đề học máy nào, như hàm mất mát, tốc độ học, hoặc hạt giống ngẫu nhiên.
- **Siêu tham số Bootstrap (Bootstrap hyperparameters):** [3] cho mỗi cây, như loại bootstrap hoặc tỷ lệ lấy mẫu con (subsampling rate).

- **Siêu tham số cấu trúc cây (Tree structure hyperparameters):** Kiểm soát hình dạng và kích thước của mỗi cây, như độ sâu, số lá, hoặc số mẫu tối thiểu trong một lá.
- **Siêu tham số tầm quan trọng của đặc trưng (Feature importance hyperparameters):** Kiểm soát cách các đặc trưng được chọn và phân tách cho mỗi cây.
- **Siêu tham số điều chuẩn (Regularization hyperparameters):** Kiểm soát mức độ phức tạp của mô hình bị phạt, như điều chuẩn L2 hoặc phương pháp ước tính lá.
- **Siêu tham số phát hiện quá khớp (Overfitting detector hyperparameters):** Kiểm soát cách dừng huấn luyện khi xảy ra quá khớp, như chỉ số đánh giá (eval metric) hoặc tùy chọn sử dụng mô hình tốt nhất. Một số siêu tham số chung được sử dụng để tinh chỉnh:
- **Learning rate (tốc độ học):** Giảm bước gradient. Giá trị càng nhỏ thì quá trình huấn luyện càng lâu, nhưng cần ít lần lặp hơn.
- **Tree Depth (độ sâu của cây):** Xác định độ sâu tối đa của mỗi cây quyết định trong tập hợp. Cây sâu hơn có thể nắm bắt các mẫu phức tạp hơn, nhưng có thể quá khớp nếu giá trị quá cao.
- **Bagging tperature:** Điều chỉnh mức độ ngẫu nhiên của các mẫu được chọn để huấn luyện. Giá trị cao hơn (> 1) làm mẫu trở nên xác định hơn, trong khi giá trị nhỏ hơn (ví dụ: 1) làm mẫu trở nên ngẫu nhiên, giúp cải thiện khả năng tổng quát hóa.
- **Border count:** Kiểm soát số lượng phân tách tối đa cho các đặc trưng số, ảnh hưởng đến độ phức tạp của mô hình và hiệu quả huấn luyện.
- **L2 regularization (Điều chuẩn L2):** Thêm một thuật ngữ phạt vào hàm mất mát trong quá trình huấn luyện để ngăn chặn các giá trị trọng số cao và khuyến khích một mô hình đơn giản hơn, giúp ngăn ngừa quá khớp. Giá trị cao hơn áp đặt điều chuẩn mạnh hơn, được kiểm soát bởi siêu tham số "reg_lambda".

2.2.5. Tinh chỉnh siêu tham số (Hyperparameter Tuning):

Là quá trình chọn bộ siêu tham số tối ưu cho một vấn đề và bộ dữ liệu cụ thể. Các bước chung để tinh chỉnh siêu tham số bao gồm:

Xác định không gian tìm kiếm: Một phạm vi hoặc danh sách các giá trị có thể có cho mỗi siêu tham số muốn tinh chỉnh.

Xác định hàm mục tiêu: Một hàm đánh giá mức độ hoạt động của mô hình trên tập xác thực với một bộ siêu tham số nhất định.

Xác định chiến lược tìm kiếm: Một phương pháp quyết định cách khám phá không gian tìm kiếm và tìm ra bộ siêu tham số tối ưu (ví dụ: grid search, random search, Bayesian optimization, Optuna).

Chạy tìm kiếm: Thực hiện chiến lược tìm kiếm và thu thập kết quả.

Phân tích kết quả: So sánh và trực quan hóa hiệu suất của các bộ siêu tham số khác nhau và chọn bộ tốt nhất.

2.2.6. Ứng dụng của CatBoost:

CatBoost có thể được sử dụng trong nhiều nhiệm vụ học máy:

a. Phân loại (Classification Tasks):

- Phân tích cảm xúc (Sentiment analysis)
- Phát hiện thư rác (ail spam detection)
- Dự đoán ung thư vú (Breast cancer prediction)

b. Hồi quy (Regression Tasks):

- Dự đoán giá nhà (House price prediction)
- Dự đoán tiêu thụ nhiên liệu (Fuel consumption prediction)
- Dự đoán thị trường chứng khoán (Stock market prediction)

2.2.7. Hạn chế của CatBoost:

Mặc dù CatBoost là một trong những thuật toán học máy tiên tiến và được đánh giá cao trong các bài toán về phân loại và hồi quy, nhưng giống như bất kỳ công cụ nào khác, nó cũng tồn tại một số hạn chế nhất định trong quá trình sử dụng:

Tiêu thụ bộ nhớ: Có thể yêu cầu tài nguyên bộ nhớ đáng kể, đặc biệt đối với các bộ dữ liệu lớn.

Tiêu thụ bộ nhớ: Mô hình có thể sử dụng lượng lớn bộ nhớ, đặc biệt khi xử lý tập dữ liệu lớn với nhiều đặc trưng phân loại.

Thời gian huấn luyện: Quá trình huấn luyện thường chậm hơn các thuật toán boosting khác như LightGBM do sử dụng cơ chế boosting có thứ tự (ordered boosting).

Tinh chỉnh siêu tham số: Việc lựa chọn và tinh chỉnh các siêu tham số hiệu quả đòi hỏi nhiều kinh nghiệm, kiến thức chuyên môn và thời gian thử nghiệm với nhiều tổ hợp khác nhau để đạt được hiệu suất tối ưu.

Hạn chế phân tán: Mặc dù có hỗ trợ GPU, CatBoost vẫn còn hạn chế trong việc huấn luyện phân tán trên nhiều máy, gây khó khăn khi áp dụng ở quy mô rất lớn.

Cộng đồng và tài liệu: So với XGBoost và LightGBM, CatBoost có cộng đồng người dùng nhỏ hơn và tài liệu hỗ trợ vẫn chưa thực sự phong phú, gây trở ngại cho người mới bắt đầu.

2.2.8. So sánh CatBoost với các thuật toán Boosting khác:

	CatBoost	LightGBM	XGboost
Xử lý đặc trưng phân loại (Categorical Features)	Xử lý đặc trưng phân loại tự động , không cần tiền xử lý	Hỗ trợ one-hot encoding và xử lý trực tiếp đặc trưng phân loại	Bắt buộc tiền xử lý đặc trưng phân loại
Chiến lược chia nhánh cây (Tree Splitting Strategy)	Chia đối xứng (Symmetric)	Chia theo lá (Leaf-wise)	Chia theo độ sâu (Depth-wise)
Khả năng diễn giải mô hình (Interpretability)	Quan trọng đặc trưng (Feature importances), SHAP values	Quan trọng đặc trưng, biểu đồ tần suất giá trị chia (split value histograms)	Quan trọng đặc trưng, sơ đồ cây (tree plots)
Tốc độ và hiệu suất (Speed and Efficiency)	Tối ưu cho tốc độ và bộ nhớ	Hiệu quả cho tập dữ liệu lớn	Khả năng mở rộng và nhanh

Bảng so sánh cho thấy CatBoost [2], đặc biệt là phiên bản đã tinh chỉnh, thường có hiệu suất cạnh tranh hoặc tốt hơn so với LightGBM và XGBoost trên một số bộ dữ liệu, với lỗi thấp hơn.

2.2.9. Cải thiện mô hình:

Để nâng cao hơn nữa mô hình của , hãy cân nhắc:

Kỹ thuật đặc trưng (Feature engineering): Tạo ra các đặc trưng thông tin từ dữ liệu thô.

Khám phá các chỉ số đánh giá khác: Đặc biệt quan trọng nếu làm việc với dữ liệu không cân bằng.

Các kỹ thuật điều chuẩn: Để ngăn chặn quá khớp.

CHƯƠNG 3: TIỀN XỬ LÝ VÀ TRỰC QUANG HÓA DỮ LIỆU SINH VIÊN

3.1. Dữ liệu sinh viên

Dữ liệu được sử dụng trong đề án đến từ nguồn mở trên nền tảng Kaggle, tại địa chỉ:

<https://www.kaggle.com/datasets/mahmoudelhaly/students-grading-dataset>

Tập dữ liệu chứa thông tin chi tiết của từng sinh viên bao gồm yếu tố học tập, xã hội và cá nhân – có ảnh hưởng trực tiếp hoặc gián tiếp đến kết quả học tập. Tổng cộng có **23 thuộc tính** (columns), trong đó thuộc tính mục tiêu (target) chính là “**Grade**”. Dưới đây là mô tả từng thuộc tính và nhận định ảnh hưởng của nó trong quá trình học máy:

Thuộc tính	Giải thích chi tiết
Student_ID	Mã định danh duy nhất cho mỗi sinh viên. Được dùng để theo dõi và phân biệt các bản ghi trong tập dữ liệu, không mang thông tin học thuật.
First_Name, Last_Name	Họ và tên sinh viên. Dữ liệu nhận diện cá nhân, không phản ánh yếu tố học tập. Có thể được ẩn danh nếu cần bảo mật.
ail	Địa chỉ liên lạc của sinh viên. Tương tự như tên, chỉ mang tính định danh hoặc quản lý, không ảnh hưởng trực tiếp đến kết quả học tập.
Gender	Giới tính của sinh viên (Nam, Nữ hoặc Khác). Có thể liên quan đến

	các khác biệt về phương pháp học tập, môi trường xã hội hoặc các yếu tố tâm lý.
Age	Tuổi của sinh viên. Tuổi tác có thể ảnh hưởng đến sự trưởng thành, khả năng tiếp thu kiến thức và phương pháp học tập.
Department	Ngành học của sinh viên (ví dụ: Khoa học máy tính, Kinh doanh,...). Ngành học khác nhau có mức độ khó, yêu cầu kỹ năng và cách đánh giá khác nhau.
Attendance (%)	Tỷ lệ chuyên cần trên lớp. Chuyên cần cao thường phản ánh mức độ cam kết và khả năng nắm bắt bài giảng tốt hơn, từ đó ảnh hưởng đến điểm số.
Midterm_Score	Điểm thi giữa kỳ (trên thang 100). Là chỉ số quan trọng thể hiện kết quả học tập trong giai đoạn đầu học kỳ.
Final_Score	Điểm thi cuối kỳ (trên 100). Đại diện cho thành tích học tập toàn bộ học kỳ, thường có trọng số cao trong tổng điểm.
Assignments_Avg	Trung bình điểm bài tập. Cho thấy mức độ chăm chỉ, đều đặn và khả

	năng áp dụng kiến thức trong quá trình học.
Quizzes_Avg	Trung bình điểm các bài kiểm tra ngắn. Giúp phản ánh sự hiểu bài ngay tại lớp và khả năng duy trì kiến thức trong thời gian ngắn.
Participation_Score	Mức độ tham gia hoạt động lớp học (trên thang 10). Thể hiện sự chủ động học tập và thái độ tích cực trong môi trường lớp học.
Projects_Score	Điểm đánh giá bài tập lớn/đồ án. Là chỉ số quan trọng thể hiện kỹ năng áp dụng thực tế, làm việc nhóm, giải quyết vấn đề.
Total_Score	Tổng điểm tích lũy – thường là trung bình có trọng số từ các yếu tố như bài tập, kiểm tra, thi cuối kỳ... Là chỉ số tổng hợp để xác định học lực.
Grade	Xếp loại học lực bằng chữ (A, B, C, D, F). Là biến mục tiêu cuối cùng trong các mô hình dự đoán hiệu suất học tập.
Study_Hours_per_Week	Số giờ tự học trung bình mỗi tuần. Là một trong những yếu tố then chốt thể hiện tính kỷ luật và mức độ đầu tư học tập.
Extracurricular_Activities	Tham gia các hoạt động ngoại khóa (Có/Không). Ảnh hưởng gián

	tiếp đến sự phát triển kỹ năng mềm và khả năng quản lý thời gian.
Internet_Access_at_Home	Có Internet tại nhà (Có/Không). Là yếu tố hỗ trợ học tập, đặc biệt quan trọng trong học online hoặc tìm kiếm tài liệu.
Parent_Education_Level	Trình độ học vấn cao nhất của phụ huynh (Không học, THPT, Cử nhân, Thạc sĩ, Tiến sĩ). Có thể ảnh hưởng đến môi trường học tập, định hướng và hỗ trợ từ gia đình.
Family_Income_Level	Mức thu nhập gia đình (Thấp, Trung bình, Cao). Phản ánh điều kiện kinh tế và khả năng tiếp cận nguồn tài nguyên học tập (sách vở, Internet, lớp thêm,...).
Stress_Level (1-10)	Mức độ căng thẳng tự đánh giá (1 là thấp, 10 là cao). Mức stress cao thường ảnh hưởng tiêu cực đến khả năng tiếp thu và kết quả học tập.
Sleep_Hours_per_Night	Số giờ ngủ trung bình mỗi đêm. Giấc ngủ đủ giúp cải thiện trí nhớ, khả năng tập trung và sức khỏe học tập tổng thể.

3.2. Tiền xử lý dữ liệu

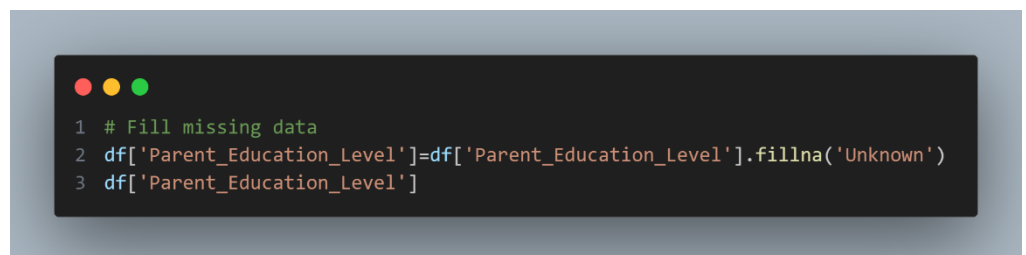
Tiền xử lý dữ liệu là bước quan trọng trước khi xây dựng mô hình học máy, nhằm đảm bảo dữ liệu đầu vào có chất lượng tốt, giảm nhiễu và phù hợp với yêu cầu của thuật toán. Trong nghiên cứu này, quá trình tiền xử lý dữ liệu được thực hiện theo các bước sau:

3.2.1. Kiểm tra và làm sạch dữ liệu

Trong quá trình khảo sát tập dữ liệu, nhóm tiến hành kiểm tra số lượng giá trị không khuyết (non-null) của từng cột bằng phương thức `df.info()`. Kết quả cho thấy đa số các cột đều có **5.000 bản ghi đầy đủ**, ngoại trừ cột **Parent_Education_Level** chỉ có **3.975 giá trị**, tương ứng với **1.025 giá trị bị thiếu**.

Để đảm bảo dữ liệu đầu vào không gây ảnh hưởng đến quá trình huấn luyện mô hình, nhóm tiến hành các bước làm sạch dữ liệu như sau:

- **Xử lý giá trị thiếu:**
 - Với cột **Parent_Education_Level**, các giá trị bị thiếu được thay thế bằng giá trị "Unknown". Điều này giúp duy trì số lượng bản ghi ban đầu, đồng thời vẫn phân biệt được những trường hợp không có thông tin về trình độ học vấn của phụ huynh.



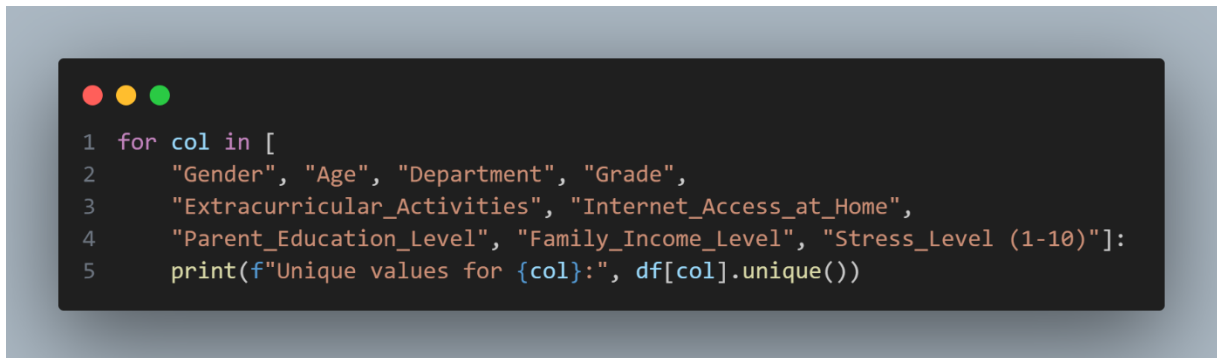
```
1 # Fill missing data
2 df['Parent_Education_Level']=df['Parent_Education_Level'].fillna('Unknown')
3 df['Parent_Education_Level']
```

Hình 3.1: điền giá trị bị thiếu

- Các cột định lượng khác không xuất hiện giá trị khuyết nên không cần xử lý bổ sung.

3.2.2. Kiểm tra giá trị duy nhất của các thuộc tính

Sau khi xử lý dữ liệu thiếu, nhóm tiến hành kiểm tra các giá trị duy nhất trong từng cột phân loại để đảm bảo tính hợp lệ và thống nhất. Kết quả cho thấy:



```
1 for col in [
2     "Gender", "Age", "Department", "Grade",
3     "Extracurricular_Activities", "Internet_Access_at_Home",
4     "Parent_Education_Level", "Family_Income_Level", "Stress_Level (1-10)"]:
5     print(f"Unique values for {col}:", df[col].unique())
```

Hình 3.2: Kiểm tra các thuộc tính là duy nhất

- **Gender:** Chỉ có 2 giá trị hợp lệ "Male" và "Fale", không có dữ liệu nhiều.
- **Age:** Các giá trị trong khoảng từ 18 đến 24 tuổi, phù hợp với độ tuổi sinh viên đại học.
- **Department:** Bao gồm 4 khoa "Mathatics", "Business", "Engineering", "CS", không xuất hiện khoa ngoài danh sách.
- **Grade:** Có 5 mức xếp loại "A", "B", "C", "D", "F", đúng theo thang điểm chữ chuẩn.
- **Extracurricular_Activities:** Chỉ gồm "Yes" và "No".
- **Internet_Access_at_Home:** Chỉ gồm "Yes" và "No".
- **Parent_Education_Level:** Có các mức "Master's", "High School", "Bachelor's", "PhD" và giá trị "Unknown" (được gán cho dữ liệu thiếu).
- **Family_Income_Level:** Gồm "Low", "Medium", "High".
- **Stress_Level (1-10):** Giá trị nguyên từ 1 đến 10, phản ánh mức độ căng thẳng.

Kết quả trên cho thấy dữ liệu phân loại đã được chuẩn hóa tốt, không cần thêm bước làm sạch nâng cao ngoài xử lý giá trị thiếu.

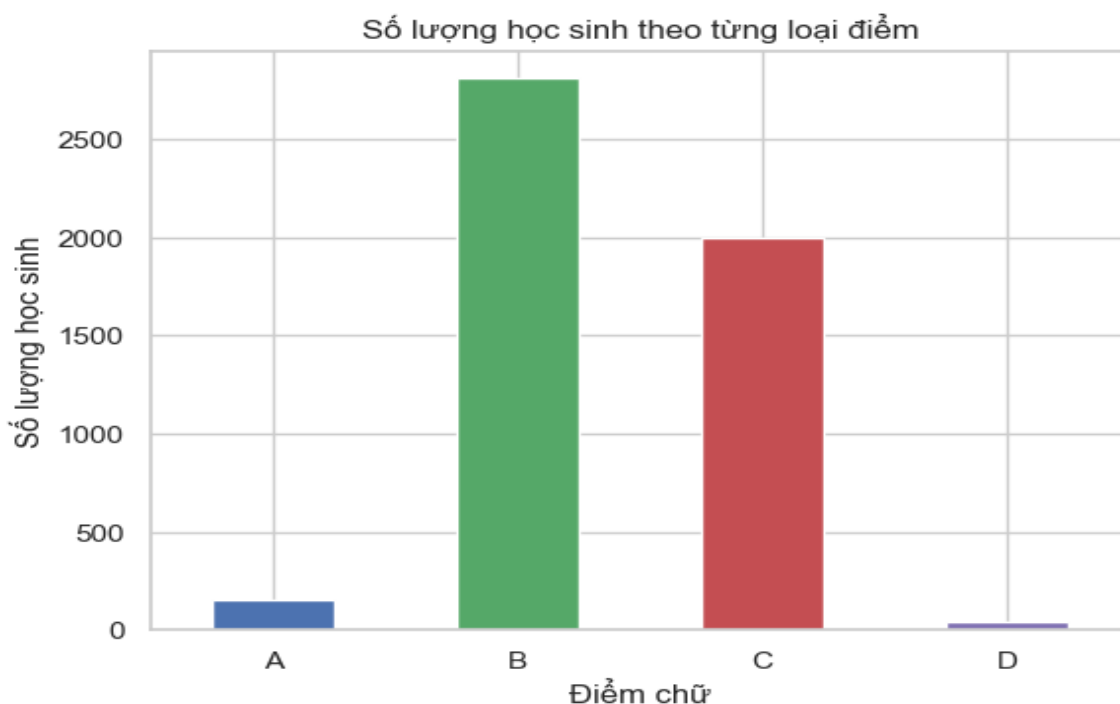
3.3. Trục quang hóa dữ liệu

Phân tích dữ liệu sinh viên là quá trình thu thập, xử lý và khai thác thông tin từ các dữ liệu liên quan đến học tập, hành vi, và đặc điểm cá nhân của sinh viên. Ý nghĩa của việc này bao gồm:

- Giúp nhà trường, giáo viên hiểu rõ hơn về tình hình học tập, các yếu tố ảnh hưởng đến kết quả học tập của sinh viên.
- Phát hiện sớm các sinh viên có nguy cơ học lực yếu để có biện pháp hỗ trợ kịp thời.
- Đánh giá hiệu quả của các chương trình đào tạo, hoạt động ngoại khóa, và các chính sách giáo dục.
- Hỗ trợ cá nhân hóa việc học, xây dựng lộ trình phù hợp cho từng sinh viên.
- Là cơ sở để dự báo, ra quyết định quản lý, cải tiến chất lượng đào tạo.

Tóm lại, phân tích dữ liệu sinh viên giúp tối ưu hóa quá trình giáo dục, nâng cao chất lượng và hiệu quả đào tạo.

3.3.1. Phân tích phân bố điểm chữ của sinh viên



Hình 3.3: Biểu đồ cột phân bố điểm

- **Grade B** chiếm tỷ lệ lớn nhất: **2,808 sinh viên** ($\approx 56,16\%$).
- **Grade C** đứng thứ hai: **1,997 sinh viên** ($\approx 39,94\%$).
- **Grade A** rất ít: **153 sinh viên** ($\approx 3,06\%$).
- **Grade D** gần như hiếm: **42 sinh viên** ($\approx 0,84\%$).

a. Nhận xét

Phân bố này cho thấy phần lớn sinh viên đạt **mức trung bình – khá** (B và C), trong khi số sinh viên xuất sắc (A) hoặc yếu (D) rất nhỏ.

Khoảng chênh lệch lớn giữa nhóm **B** và các nhóm còn lại có thể phản ánh:

- **Đề thi, bài tập hoặc tiêu chí chấm điểm** được thiết kế để đa số sinh viên đạt mức khá.
- **Hiệu suất học tập** của phần lớn sinh viên tương đối đồng đều, không có nhiều cá nhân nổi trội hoặc yếu kém.

Nhóm **A** và **D** đều là thiểu số, cho thấy độ phân tán kết quả thấp, dữ liệu tương đối “tập trung” quanh trung bình.

b. Ý nghĩa đối với mô hình học máy

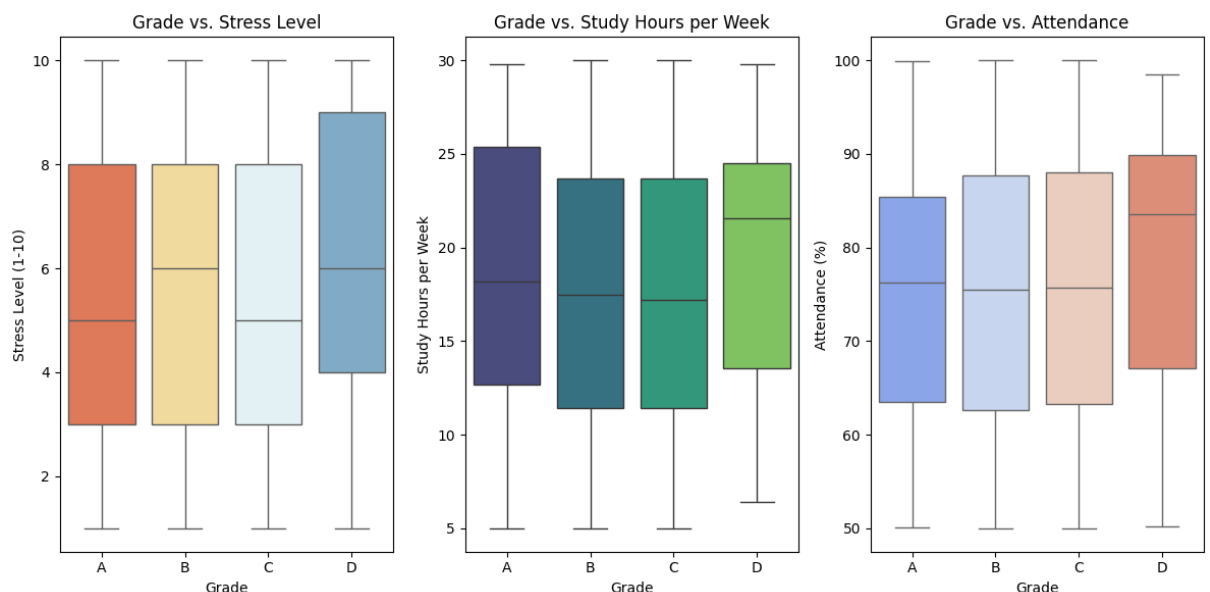
Vấn đề mất cân bằng lớp (class imbalance):

- Lớp B và C chiếm gần 96% dữ liệu → mô hình dễ **thiên lệch** dự đoán vào hai nhóm này, khó nhận diện chính xác A và D.
- Các lớp A và D cần **kỹ thuật xử lý imbalance** (oversampling, SMOTE, hoặc trọng số lớp trong CatBoost).

Khó phân loại nhóm hiếm:

- Vì dữ liệu nhóm A và D quá ít, mô hình cần được huấn luyện với cơ chế penalize lỗi nhiều hơn ở nhóm hiếm.
- Có thể kết hợp **feature engineering** để tìm đặc trưng giúp tách bạch A/D khỏi B/C.

3.3.2. Phân tích các yếu tố ảnh hưởng đến học tập



Hình 3.4: Mối quan hệ giữa các yếu tố học tập

a. Grade vs. Stress Level

Điểm A, B, C: Trung vị stress dao động từ ~5 đến 6, phân bố khá rộng từ 1 đến 10, Không có chênh lệch lớn về mức độ stress giữa các nhóm điểm này.

Điểm D: Trung vị stress cao hơn (~6), cho thấy nhóm học sinh điểm thấp có xu hướng chịu áp lực nhiều hơn. Có thể stress là yếu tố góp phần làm giảm hiệu suất học tập.

Ý nghĩa: Stress không hoàn toàn tỷ lệ nghịch với điểm số, nhưng nhóm điểm D đáng chú ý vì áp lực cao hơn.

b. Grade vs. Study Hours per Week

Điểm A: Trung vị số giờ học cao nhất (~18 giờ/tuần). Khoảng phân bố lệch về phía nhiều giờ học, cho thấy nhóm này thường đầu tư thời gian học đáng kể.

Điểm B và C: Trung vị thấp hơn (khoảng 17 giờ cho C, 17–18 giờ cho B). Có sự phân tán cao, tức là trong nhóm này vẫn có học sinh học rất nhiều hoặc rất ít.

Điểm D: Đáng ngạc nhiên là trung vị giờ học (~21–22 giờ/tuần) cao hơn cả nhóm B và C. Điều này có thể cho thấy học sinh điểm D dù học nhiều nhưng phương pháp học chưa hiệu quả.

Ý nghĩa: Thời gian học không phải yếu tố duy nhất quyết định điểm số chất lượng và phương pháp học cũng quan trọng.

c. Grade vs. Attendance (%)

Điểm A: Trung vị điểm danh khoảng 76–77%, không quá cao.

Điểm B và C: Trung vị tương tự nhau (~75%), phân bố rộng từ 50% đến 100%.

Điểm D: Trung vị cao nhất (~83–85%), điều này khá bất thường — nhóm điểm thấp lại đi học đều hơn. Có thể nguyên nhân là phương pháp học chưa phù hợp, hoặc chỉ tham gia lớp học không đủ để đạt điểm cao.

Ý nghĩa: Tỷ lệ điểm danh cao không đồng nghĩa với kết quả học tập cao, cho thấy cần kết hợp nhiều yếu tố khác như phương pháp học và kỹ năng làm bài.

3.3.3. Phân tích hoạt động ảnh hưởng đến tổng điểm

a. Extracurricular Activities vs. Total Score (Hoạt động ngoại khóa)

- **Yes** và **No** có trung vị điểm gần như tương đương (~71–72).
- Phân bố điểm ở cả hai nhóm khá rộng (từ ~50 đến >90).
- Nhóm “Yes” có một số điểm cao nổi bật, nhưng khác biệt tổng thể không lớn.

Nhận xét: Tham gia hoạt động ngoại khóa không tạo ra sự khác biệt rõ rệt về điểm số trung bình; có thể ảnh hưởng gián tiếp qua kỹ năng mềm hoặc tinh thần học tập.

b. Internet Access vs. Total Score (Truy cập Internet tại nhà)

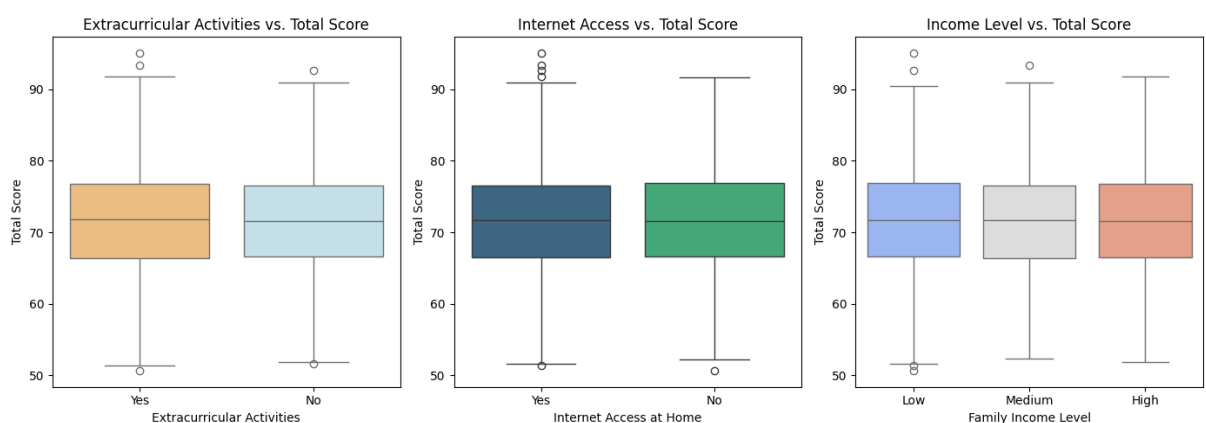
- **Yes** và **No** có trung vị điểm tương tự (~71–72).
- Nhóm “Yes” có nhiều giá trị ngoại lai ở mức điểm cao hơn.
- Khoảng phân bố điểm rộng ở cả hai nhóm.

Nhận xét: Có Internet ở nhà không đảm bảo điểm số cao hơn, nhưng có thể giúp một số học sinh đạt thành tích vượt trội.

c. Income Level vs. Total Score (Mức thu nhập gia đình)

- **Low, Medium, High** đều có trung vị gần nhau (~71–72).
- Phân bố điểm rất tương tự, không có nhóm nào vượt trội rõ rệt.
- Cả ba nhóm đều có điểm cao và thấp.

Nhận xét: Mức thu nhập gia đình không cho thấy tác động rõ rệt đến điểm số tổng thể; yếu tố cá nhân và môi trường học tập có thể quan trọng hơn.



Hình 3.5: Hoạt động ảnh hưởng đến tổng điểm

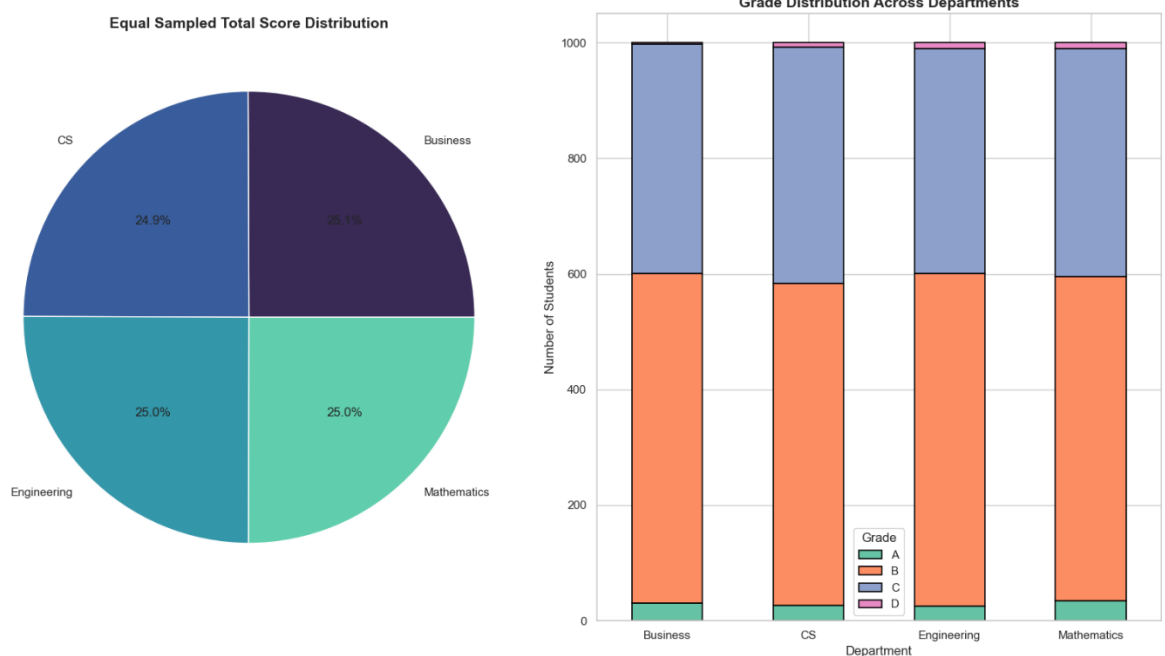
3.3.4. Phân tích phân bố điểm số theo khoa

a. Equal Sampled Total Score Distribution (Phân bố mẫu đều theo khoa)

- Tỷ lệ sinh viên giữa các khoa **Business**, **CS**, **Engineering**, và **Mathematics** gần như bằng nhau (~25%).
- Điều này cho thấy dữ liệu đã được **lấy mẫu cân bằng** nhằm tránh thiên lệch khi so sánh thành tích học tập giữa các khoa.

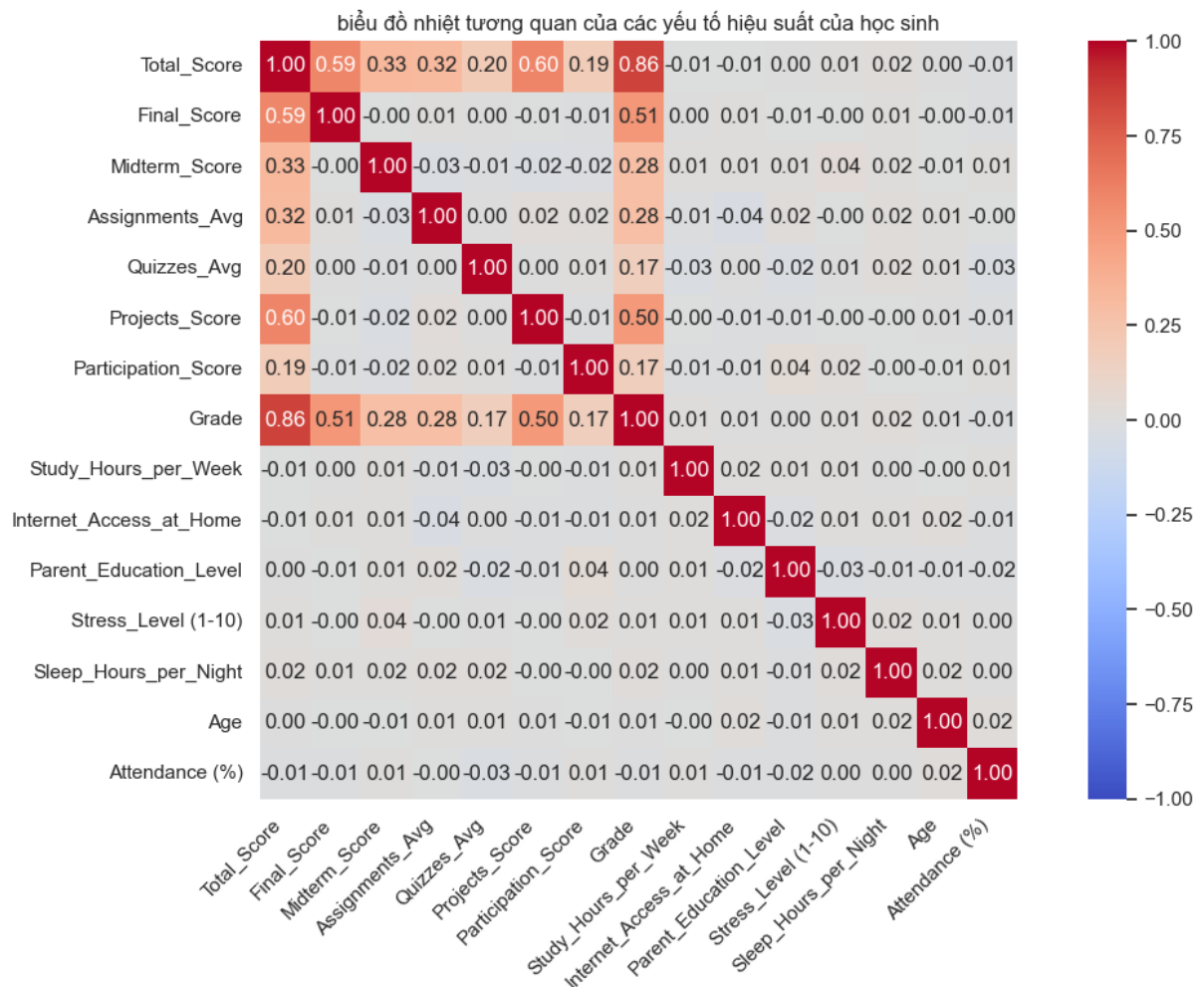
b. Grade Distribution Across Departments (Phân bố loại điểm theo khoa)

- Ở cả 4 khoa, điểm **B** chiếm tỷ lệ áp đảo (~58–60%), tiếp theo là điểm **C** (~38–40%).
- Điểm **A** chiếm tỷ lệ rất thấp (~2–3%) ở tất cả các khoa.
- Điểm **D** xuất hiện với tỷ lệ cực nhỏ (<1%), gần như không đáng kể.
- Không có sự khác biệt rõ rệt về phân bố điểm giữa các khoa — cấu trúc điểm gần như giống hệt nhau.



Hình 3.6: Phân tích phân bố điểm theo khoa

3.3.5. Phân tích tổng quan với biểu đồ nhiệt



Hình 3.7: Biểu đồ nhiệt

Biểu đồ nhiệt này thể hiện **hệ số tương quan Pearson** giữa các yếu tố trong bộ dữ liệu và cho thấy mức độ liên hệ tuyến tính giữa chúng. Dựa vào số liệu trong biểu đồ, có thể rút ra các ý chính:

a. Các yếu tố liên quan chặt chẽ với nhau

- Total_Score có tương quan mạnh với:
 - Grade (0.86) → Tổng điểm ảnh hưởng trực tiếp tới xếp loại.
 - Projects_Score (0.60) → Điểm dự án góp phần lớn vào tổng điểm.
 - Final_Score (0.59) → Điểm cuối kỳ là thành phần quan trọng của tổng điểm.
- Final_Score tương quan vừa với Grade (0.51).

- Projects_Score tương quan vừa với Grade (0.50).

b. Các yếu tố gần như không liên quan

- Study_Hours_per_Week, Internet_Access_at_Home, Parent_Education_Level, Stress_Level, Sleep_Hours_per_Night, Age, và Attendance (%) có hệ số tương quan rất thấp (gần 0) với hầu hết các yếu tố điểm số → nghĩa là trong dữ liệu này, các yếu tố này không cho thấy mối liên hệ tuyến tính rõ ràng với hiệu suất học tập.
- Midterm_Score và Total_Score (0.33) → Điểm giữa kỳ có ảnh hưởng nhưng không mạnh như điểm cuối kỳ hoặc điểm dự án.
- Assignments_Avg và Total_Score (0.32) → Trung bình bài tập cũng góp phần nhưng không lớn.

c. Ý nghĩa tổng quan

- Các yếu tố liên quan trực tiếp đến **điểm đánh giá (Final, Midterm, Projects, Assignments)** mới có ảnh hưởng đáng kể đến tổng điểm và xếp loại.
- Các yếu tố về **điều kiện học tập, thời gian học, hoặc yếu tố cá nhân** trong bộ dữ liệu này không thể hiện tương quan mạnh với kết quả học tập.

d. Điểm bất cập

Trong phần tới sẽ huấn luyện mô hình dựa trên các thuộc tính như Study_Hours_per_Week, Internet_Access_at_Home, Parent_Education_Level, Stress_Level, Sleep_Hours_per_Night, Age, và Attendance (%)... Tuy nhiên, theo biểu đồ nhiệt về hệ số tương quan, các yếu tố này đều có **độ tương quan rất thấp** với các chỉ số kết quả học tập (Total_Score, Grade, Final_Score, ...). Điều này đồng nghĩa rằng về mặt thống kê, chúng không thể hiện mối liên hệ tuyến tính mạnh mẽ với hiệu suất học tập.

Nguyên nhân không sử dụng các thuộc tính có độ tương quan cao như Final_Score, Midterm_Score, Projects_Score, hay Assignments_Avg là vì:

- Các thuộc tính này **trực tiếp phản ánh kết quả học tập** (ví dụ: điểm giữa kỳ, cuối kỳ, điểm dự án).
- Nếu dùng chúng để huấn luyện mô hình, kết quả dự đoán sẽ gần như **chỉ là phép tính cộng/trung bình** từ các điểm thành phần, chứ không còn là quá trình “dự đoán” thực sự.
- Mục tiêu ở đây là **dự đoán hiệu suất học tập trước khi có điểm thi hoặc điểm dự án**, nên cần dựa vào các yếu tố gián tiếp như thói quen học tập, điều kiện học tập, yếu tố cá nhân, v.v.

Do đó, việc huấn luyện mô hình trên các thuộc tính có tương quan thấp tuy khó khăn hơn, nhưng sẽ giúp mô hình phản ánh đúng năng lực dự đoán từ các dữ liệu đầu vào chưa chứa sẵn thông tin kết quả.

3.3.6. Kết luận chung qua quá trình phân tích

a. Kết luận chi tiết

Kết quả phân tích cho thấy yếu tố quyết định mạnh mẽ nhất đến thành tích học tập của sinh viên là **điểm các bài kiểm tra đánh giá**, đặc biệt:

- **Final Exam** có mức ảnh hưởng lớn nhất, thể hiện qua hệ số tương quan cao.
- **Midterm, dự án và bài tập** cũng góp phần quan trọng nhưng mức độ thấp hơn một chút so với Final Exam.

Ngược lại, các yếu tố như **thời gian học mỗi tuần, mức độ căng thẳng, điểm danh**, hay **tham gia hoạt động ngoại khóa** cho thấy mối liên hệ yếu hoặc không đáng kể với điểm số. Điều này gợi ý rằng, **chỉ số điểm số được hình thành chủ yếu từ hiệu quả làm bài thi, không chỉ từ thời gian hoặc tần suất tham gia lớp học**.

b. Một số phát hiện đáng chú ý khác:

- **Trình độ học vấn của cha mẹ** có xu hướng tác động tích cực: Cha mẹ có trình độ học vấn cao (Thạc sĩ, Tiến sĩ) thường có con đạt điểm cao hơn.

- **Thời gian học hợp lý** (tối đa khoảng 30 giờ/tuần) thường đi kèm kết quả tốt; tuy nhiên, học quá nhiều không đảm bảo điểm cao hơn.
- **Mức độ căng thẳng vừa phải** (4–6/10) có thể mang tính thúc đẩy, nhưng căng thẳng quá cao hoặc quá thấp đều không tối ưu cho kết quả học tập.
- **Truy cập internet tại nhà** mang lại lợi thế nhỏ, nhưng không tạo khác biệt quá lớn.
- **Hoạt động ngoại khóa** có ảnh hưởng gián tiếp: Tham gia vừa phải giúp cải thiện kỹ năng mềm và điểm số, nhưng tham gia quá nhiều dễ gây quá tải và giảm thành tích học tập.

c. Gợi ý các phương pháp cải thiện hiệu suất học tập

1. **Tập trung cải thiện kỹ năng làm bài thi và dự án**, vì đây là yếu tố then chốt quyết định điểm số.
2. **Ưu tiên chất lượng học** hơn là thời lượng; học quá nhiều giờ chưa chắc hiệu quả.
3. **Đầu tư hạ tầng công nghệ và tài nguyên học tập** để đảm bảo sinh viên có đủ điều kiện học tập.
4. **Quản lý căng thẳng hợp lý**, tránh để áp lực học tập vượt quá khả năng chịu đựng.
5. **Khuyến khích hoạt động ngoại khóa** ở mức cân bằng để vừa phát triển kỹ năng mềm vừa duy trì kết quả học tập.

CHƯƠNG 4: XÂY DỰNG MÔ HÌNH CATBOOST DỰ ĐOÁN HIỆU XUẤT HỌC TẬP SINH VIÊN

4.1. Lý do chọn mô hình CatBoost

Để lựa chọn mô hình CatBoost thay vì các mô hình khác như XGBoost hay LightGBM trong một nghiên cứu dự đoán hiệu suất học tập sinh viên, có một số lý do chính dựa trên các tính năng và ưu điểm nổi bật của CatBoost:

	CatBoost	LightGBM	XGboost
Xử lý đặc trưng phân loại (Categorical Features)	Xử lý đặc trưng phân loại tự động , không cần tiền xử lý	Hỗ trợ one-hot encoding và xử lý trực tiếp đặc trưng phân loại	Bắt buộc tiền xử lý đặc trưng phân loại
Chiến lược chia nhánh cây (Tree Splitting Strategy)	Chia đối xứng (Symmetric)	Chia theo lá (Leaf-wise)	Chia theo độ sâu (Depth-wise)
Khả năng diễn giải mô hình (Interpretability)	Quan trọng đặc trưng (Feature importances), SHAP values	Quan trọng đặc trưng, biểu đồ tần suất giá trị chia (split value histograms)	Quan trọng đặc trưng, sơ đồ cây (tree plots)
Tốc độ và hiệu suất (Speed and Efficiency)	Tối ưu cho tốc độ và bộ nhớ	Hiệu quả cho tập dữ liệu lớn	Khả năng mở rộng và nhanh

4.1.1. Khả năng xử lý tự động các tính năng phân loại

CatBoost là một bộ công cụ tăng cường gradient mã nguồn mở phổ biến trên cây quyết định, được tạo bởi Yandex. Một trong những ưu điểm đáng kể nhất của nó là khả năng **tự động xử lý các tính năng phân loại mà không cần mã**

hóa hay tiền xử lý. Nó mã hóa các tính năng phân loại bằng cách sử dụng các chiến lược mã hóa mục tiêu (target encoding) và one-hot encoding nội bộ.

Điều này mang lại lợi thế lớn so với **XGBoost** và **LightGBM**, vốn thường yêu cầu người dùng phải thực hiện tiền xử lý thủ công (như one-hot encoding) cho các tính năng phân loại. One-Hot Encoding có thể dẫn đến ma trận thưa và gây ra hiện tượng overfitting.

4.1.2. Giảm thiểu hiện tượng Overfitting

CatBoost được thiết kế để giảm thiểu overfitting bằng cách sử dụng một sơ đồ tăng cường gradient mới lạ và các kỹ thuật điều chuẩn (regularization techniques).

Cụ thể, nó sử dụng các kỹ thuật thông minh như **tăng cường theo thứ tự (ordered boosting)**, kết hợp tính năng ngẫu nhiên (random feature combinations), và các phương pháp tăng cường mạnh mẽ để giúp mô hình hoạt động tốt ngay cả trên dữ liệu mới, chưa từng thấy. Kỹ thuật này giúp giải quyết vấn đề thiên vị dự đoán (prediction shift) thường gặp trong các thuật toán tăng cường gradient khác khi xử lý dữ liệu phân loại.

4.1.3. Hiệu suất cao và khả năng mở rộng

CatBoost có thể đạt được hiệu suất cao và khả năng mở rộng thông qua các triển khai hiệu quả cho CPU và GPU.

Nó hỗ trợ đào tạo tăng tốc bằng GPU, giúp tăng tốc quá trình xây dựng mô hình, đặc biệt là khi làm việc với các bộ dữ liệu lớn.

Ngoài ra, CatBoost sử dụng các kỹ thuật xử lý song song để tận dụng nhiều lõi CPU trong quá trình đào tạo, giúp quá trình này hiệu quả và có khả năng mở rộng cho các vấn đề phức tạp và tác vụ học máy quy mô lớn.

4.1.4. Xử lý giá trị bị thiếu (*Missing Values*)

Không giống như nhiều mô hình khác, CatBoost có khả năng xử lý các giá trị bị thiếu trong dữ liệu đầu vào mà **không yêu cầu điền giá trị (imputation)**.

Thuật toán Symmetric Weighted Quantile Sketch (SWQS) của CatBoost xử lý hiệu quả dữ liệu bị thiếu bằng cách giảm overfitting và cải thiện hiệu suất mô hình.

4.1.5. Chiến lược phân tách cây (*Tree Splitting Strategy*)

CatBoost sử dụng chiến lược phân tách cây **đối xứng (Symmetric)**. Điều này có nghĩa là các cây quyết định trong CatBoost được xây dựng theo cách cân bằng hơn, giúp duy trì cấu trúc cây ổn định.

Ngược lại, LightGBM sử dụng chiến lược theo lá (Leaf-wise), và XGBoost sử dụng chiến lược theo độ sâu (Depth-wise). Chiến lược Leaf-wise của LightGBM có thể xây dựng các cây rất sâu và không cân bằng, đôi khi dẫn đến overfitting nếu không được kiểm soát tốt.

4.1.6. So sánh hiệu suất thực tế

Theo kết quả so sánh trên một số bộ dữ liệu, **CatBoost đã tinh chỉnh (Tuned CatBoost)** thường cho thấy hiệu suất tương đương hoặc tốt hơn so với LightGBM và XGBoost (cả mặc định và đã tinh chỉnh). Ví dụ, trên bộ dữ liệu "Adult" và "Amazon", Tuned CatBoost đạt lỗi thấp hơn đáng kể so với các đối thủ. Điều này cho thấy CatBoost có tiềm năng cung cấp độ chính xác cao ngay cả khi không cần tinh chỉnh quá nhiều.

4.2. Mục tiêu mô hình

Mục tiêu của chương này là phát triển mô hình dự đoán kết quả học tập của học sinh dựa trên các yếu tố đầu vào như thông tin nhân khẩu học, điểm số các kì tước, điều kiện gia đình và thói quen học tập... Mô hình được lựa chọn là **CatBoostClassifier** – một thuật toán boosting trên cây quyết định (Gradient Boosting Decision Trees) do Yandex phát triển, nổi bật ở khả năng:

- Xử lý trực tiếp biến phân loại (categorical features) không cần one-hot encoding.
- Giảm hiện tượng overfitting thông qua điều chuẩn (l2_leaf_reg).
- Tự động chọn cách chia dữ liệu (oblivious decision trees).
- Hiệu quả cao với dữ liệu có nhiều loại biến khác nhau.

4.3. Chuẩn bị dữ liệu

Dữ liệu được lấy từ file **students_grading_dataset_clean.csv** với các bước tiền xử lý sau:

```
1 # Đọc dữ liệu
2 df = pd.read_csv(r'D:\MINI_Project\DoAn3\predict_train_XGBoost\students_grading_dataset_clean.csv')
3
4 # Loại bỏ các cột không cần thiết
5 df.drop(['Student_ID', 'First_Name', 'Last_Name', 'Email', 'Midterm_Score',
6         'Final_Score', 'Projects_Score', 'Total_Score'],
7         axis=1, inplace=True)
8
9 # Tách X, y
10 X = df.drop('Grade', axis=1)
11 y = df['Grade']
12
13 # Xác định các cột phân loại
14 cat_features = X.select_dtypes(include='object').columns.tolist()
15 #print("Categorical features:", cat_features)
16
17 # Chia dữ liệu
18 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
19
```

Hình 4.1: Chuẩn bị dữ liệu

Loại bỏ cột không cần thiết

- Student_ID, First_Name, Last_Name, ail: thông tin định danh cá nhân, không liên quan đến dự đoán.

Tách biến đầu vào và biến mục tiêu

- **X**: tập đặc trưng đầu vào (features).
- **y**: biến mục tiêu (Grade) – hạng của học sinh (A, B, C, D).

Xác định biến phân loại

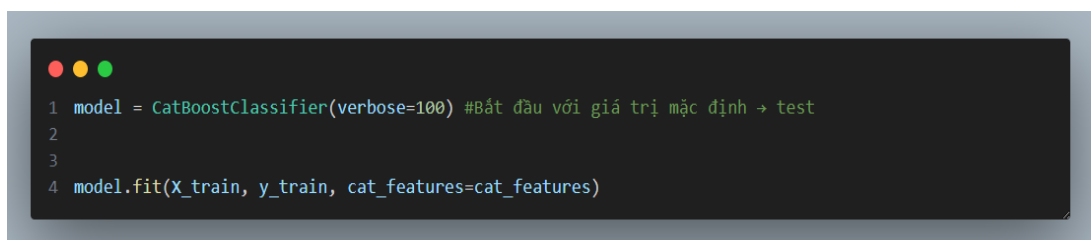
- Dùng `select_dtypes(include='object')` để lấy danh sách cột dạng chuỗi.
- Các biến này được truyền vào `cat_features` của CatBoost để mô hình tự mã hóa tối ưu.

Chia tập dữ liệu

- `train_test_split` với tỷ lệ 80% huấn luyện – 20% kiểm tra, `random_state=42` để tái lập kết quả.

4.4. Mô hình CatBoost chưa tinh chỉnh (Baseline)

Phiên bản baseline của mô hình được huấn luyện với cấu hình mặc định:



```
1 model = CatBoostClassifier(verbose=100) #Bắt đầu với giá trị mặc định + test
2
3
4 model.fit(X_train, y_train, cat_features=cat_features)
```

Hình 4.2: Mô hình CatBoos chưa tinh chỉnh tham số

4.4.2. Ý nghĩa các tham số **mặc định** quan trọng

- `iterations=1000`: Số vòng lặp (cây) huấn luyện. Nhiều vòng hơn giúp mô hình học sâu hơn nhưng dễ overfit.
- `learning_rate=0.03`: Tốc độ học, càng nhỏ thì học càng chậm nhưng ổn định hơn.
- `depth=6`: Độ sâu tối đa của mỗi cây. Độ sâu lớn giúp học mối quan hệ phức tạp hơn nhưng tăng nguy cơ overfitting.

- `l2_leaf_reg=3.0`: Hệ số điều chuẩn L2, càng lớn thì mô hình càng bị phạt khi cây quá phức tạp.
- `loss_function='MultiClass'`: Hàm mất mát cho phân loại nhiều lớp.

4.4.3. Phân tích kết quả

Hiệu suất tổng quan

- Mô hình đạt **accuracy 95%** và **F1-weighted 0.95**, chứng tỏ khả năng dự đoán tổng thể rất tốt.
- Điều này chủ yếu nhờ các lớp lớn (1 và 2) được dự đoán chính xác và cân bằng giữa precision và recall.

Kết quả theo từng lớp

- **Lớp 0**: Precision cao (93%), nhưng recall chỉ 81% → mô hình bỏ sót một số mẫu.
- **Lớp 1 và lớp 2**: F1 gần 0.95–0.96, hiệu suất ổn định nhờ dữ liệu nhiều.
- **Lớp 3**: Kém nhất, F1 = 0.57 do chỉ có 8 mẫu, khó học được đặc trưng.

Vấn đề nổi bật

- Dữ liệu **mất cân bằng** nghiêm trọng, đặc biệt là lớp 3 rất ít mẫu.
- Macro F1 chỉ 0.84, thấp hơn weighted F1 do các lớp nhỏ bị dự đoán kém.

Hướng cải thiện

- **Tăng dữ liệu** cho các lớp nhỏ, nhất là lớp 3.
- Áp dụng **class_weight** hoặc kỹ thuật **oversampling** (SMOTE).
- Tinh chỉnh tham số như `depth`, `l2_leaf_reg`, `bagging_temperature`.
- Kiểm tra **feature importance** để loại bỏ đặc trưng ít đóng góp.

4.4.4. Nhận xét tổng quan

Mô hình CatBoost đang hoạt động rất tốt trên bộ dữ liệu này, tuy nhiên để cải thiện khả năng nhận diện lớp A và D, có thể cần **thu thập thêm dữ liệu** cho nhóm này hoặc **tối ưu tham số** hướng tới các lớp thiểu số (ví dụ dùng `class_weight` hoặc `oversampling`).

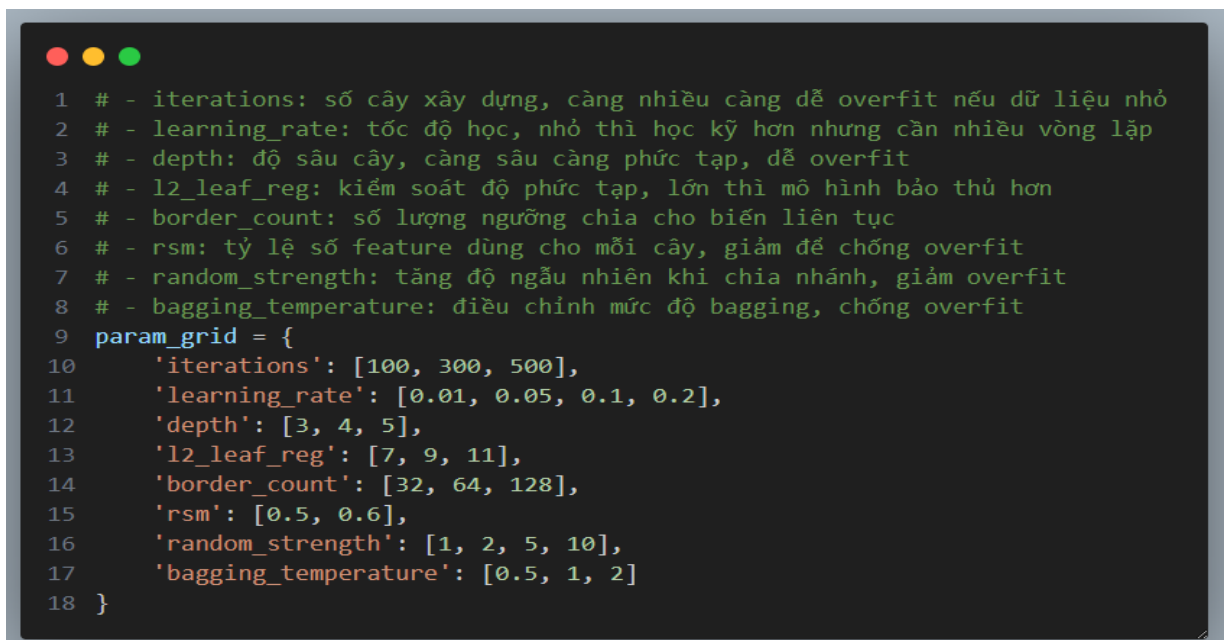
4.5. Tinh chỉnh siêu tham số bằng RandomizedSearchCV

Khi chạy mô hình mặc với các tham số mặc định, kết quả cho thấy được dự đoán khá tốt. Nhưng để tối ưu hơn, tiến hành **tinh chỉnh siêu tham số có hệ thống** bằng phương pháp **RandomizedSearchCV** nhằm tìm tổ hợp tham số tối ưu. [3]

4.5.1. Phương pháp

- **Lựa chọn RandomizedSearchCV thay cho GridSearchCV:**
 - Với **GridSearchCV**, tất cả các tổ hợp tham số sẽ được thử, dẫn đến thời gian huấn luyện tăng mạnh, đặc biệt khi số tham số cần điều chỉnh nhiều.
 - **RandomizedSearchCV** chỉ thử một số tổ hợp ngẫu nhiên (trong nghiên cứu này đặt $n_iter = 20$), giúp giảm đáng kể thời gian huấn luyện nhưng vẫn có khả năng tìm được bộ tham số tối ưu.
 - Cách tiếp cận này đặc biệt phù hợp với **CatBoost**, vốn có nhiều siêu tham số như `depth`, `l2_leaf_reg`, `border_count`,...
- **Tiêu chí đánh giá – f1_weighted:**
 - Không sử dụng **accuracy** vì độ chính xác có thể gây sai lệch khi dữ liệu bị **mất cân bằng nhãn** (lớp nhiều chiếm ưu thế).
 - **f1_weighted** tính trung bình F1-score có trọng số theo kích thước của từng lớp, giúp đánh giá mô hình công bằng hơn và phản ánh đúng hiệu suất trên toàn bộ tập dữ liệu.
- **Chiến lược kiểm định – Cross Validation (cv = 3):**
 - Tập dữ liệu được chia thành 3 phần, mỗi phần lần lượt đóng vai trò tập kiểm tra trong khi hai phần còn lại dùng huấn luyện.
 - Phương pháp này giúp giảm hiện tượng **overfitting** và tăng khả năng **tổng quát hóa** của mô hình khi áp dụng cho dữ liệu mới.
- **Tăng tốc xử lý:**
 - Sử dụng `n_jobs = -1` để tận dụng toàn bộ lõi CPU, rút ngắn thời gian tìm kiếm tham số.

4.5.2. Các tham số tinh chỉnh



Hình 4.3: Tinh chỉnh siêu tham số với *RandomizedSearchCV*

Bộ tham số cho RandomizedSearchCV

- **iterations** – Số cây được xây dựng (lớn → học kỹ hơn, nhưng dễ overfit).
- **learning_rate** – Tốc độ học (nhỏ → học chậm nhưng chính xác, lớn → học nhanh nhưng dễ bỏ sót thông tin).
- **depth** – Độ sâu cây (lớn → mô hình học mối quan hệ phức tạp, nhưng dễ overfit).
- **l2_leaf_reg** – Điều chỉnh mức phạt độ phức tạp (nhỏ → dễ học theo nhiễu, lớn → bảo thủ hơn).
- **border_count** – Số ngưỡng chia biến liên tục (quyết định độ chi tiết khi phân chia).
- **rsm** – Tỷ lệ chọn ngẫu nhiên feature cho mỗi cây (giảm → giảm overfit nhưng có thể giảm chính xác).
- **random_strength** – Mức độ ngẫu nhiên khi chọn split (tăng → giảm overfit).
- **bagging_temperature** – Điều chỉnh mức đa dạng mẫu huấn luyện giữa các cây (giúp chống overfit).

4.5.3. Kết quả mô hình tối ưu

Sau khi chạy **RandomizedSearchCV** (10 tổ hợp tham số, 3-fold cross validation), mô hình tốt nhất đạt các siêu tham số tối ưu và cho kết quả:

Accuracy: 0.96 → Mô hình dự đoán đúng 96% trường hợp.

F1-score (weighted): 0.96 → Cân bằng tốt giữa precision và recall, đặc biệt quan trọng khi dữ liệu có nhiều lớp khác nhau.

lớp	Precision	Recall	F1-score	Nhận xét
0	0.92	0.77	0.84	Precision cao nhưng recall thấp hơn → mô hình bỏ sót nhiều mẫu của lớp 0.
1	0.97	0.96	0.97	Rất tốt, dự đoán chính xác cả precision và recall.
2	0.95	0.96	0.96	Ổn định và chính xác cao.
3	0.71	0.62	0.67	Thấp nhất do số lượng mẫu ít (support = 8), dễ gây mất cân bằng khi huấn luyện.

- **Điểm mạnh**
- Mô hình cân bằng tốt giữa precision và recall ở các lớp chính (1, 2).
- Không bị quá khớp (overfit) rõ rệt vì recall và precision vẫn ổn định trên nhiều lớp.
- RandomizedSearchCV giúp tìm được bộ siêu tham số khá tối ưu.

CHƯƠNG 5: KẾT LUẬN

5.1. Kết quả đạt được đánh đạt được

Mô hình **CatBoost** được huấn luyện để dự đoán **Grade** của sinh viên (A/B/C/D) dựa trên **các thông tin cá nhân** (giới tính, tuổi, khoa, trình độ học vấn của cha mẹ, mức thu nhập gia đình), **thông tin học tập** (tỷ lệ tham dự, điểm giữa kỳ, điểm bài tập, điểm quiz, điểm dự án) và **thói quen sinh hoạt** (giờ học mỗi tuần, hoạt động ngoại khóa, mức độ căng thẳng, giờ ngủ mỗi đêm).

dự đoán điểm học kì tiếp theo - CatBoost Model

Dự đoán điểm Grade (A/B/C/D)

Nhập thông tin để dự đoán

Gender: Male

Age: 20

Department: Mathematics

Attendance (%): 90.00

Midterm_Score: 50.00

Final_Score: 50.00

Assignments_Avg: 50.00

Quizzes_Avg: 50.00

Participation_Score: 5.00

Projects_Score: 50.00

Total_Score: 50.00

Study_Hours_per_Week: 10.00

Extracurricular_Activities: Yes

Internet_Access_at_Home: Yes

Parent_Education_Level: High School

Family_Income_Level: Low

Stress_Level (1-10): 5

Sleep_Hours_per_Night: 8.00

Dự đoán Grade

Hình 5.1: Giao diện streamlit giúp người dùng nhập thông tin

Sau khi nhập đủ thông tin mô hình đưa ra dự đoán điểm ở học kì tiếp theo và các lời khuyên bổ ích phụ thuộc vào các thuộc tính như số giờ ngủ mỗi ngày, số giờ học mỗi tuần để cải thiện hiệu suất học tập.

Dự đoán Grade

Dự đoán điểm học kì tiếp theo của bạn: A

Xác suất dự đoán cho từng lớp:

	A	B	C	D
0	95.30%	2.52%	1.28%	0.59%

Lời khuyên để cải thiện hiệu suất học tập

- Ngủ thêm một chút nữa sẽ giúp bạn tỉnh táo và tiếp thu bài nhanh hơn.
- Hoàn thành bài tập đầy đủ và luyện tập thêm để nâng cao điểm trung bình.

Hình 5.2: Kết quả đạt được

- **Độ chính xác (Accuracy):** 0.96
- **F1-score** (lớp 1 – đối tượng quan tâm chính): 0.97
- **Precision:** 0.97
- **Recall:** 0.97

Mặc dù mô hình đạt **độ chính xác tổng thể 96%** kết quả này được xem là **cao** đối với bài toán dự đoán, nguyên nhân chủ yếu bao gồm:

Dữ liệu huấn luyện và kiểm tra có thể chưa đủ đa dạng hoặc phân bố tương tự nhau → dễ khiến mô hình dự đoán tốt hơn thực tế.

Lớp 1 chiếm tỷ lệ lớn (562/1000 mẫu), nên việc dự đoán đúng lớp này giúp nâng Accuracy và F1-score đáng kể.

Các lớp ít mẫu (ví dụ lớp 3) chưa được mô hình học tốt, tiềm ẩn nguy cơ giảm hiệu suất khi gặp dữ liệu thực tế khác biệt.

Kích thước tập dữ liệu: Số lượng mẫu tuy đạt mức chấp nhận được (5000 bản ghi) nhưng vẫn chưa đủ lớn để mô hình học được các mối quan hệ phức tạp giữa các biến.

5.2. Ứng Dụng và Tinh Chỉnh

5.2.1. Ứng dụng trong giáo dục (thực tế)

- **Cảnh báo sớm học sinh yếu kém:** Hệ thống tự động gửi thông báo đến giáo viên chủ nhiệm và phụ huynh khi phát hiện học sinh có nguy cơ rớt chuẩn đầu ra hoặc trượt kỳ thi quan trọng.
- **Lập danh sách ưu tiên hỗ trợ:** Giúp ban giám hiệu dễ dàng tạo danh sách học sinh cần tham gia lớp phụ đạo, bồi dưỡng kỹ năng học tập hoặc tư vấn tâm lý.
- **Xây dựng kế hoạch ôn tập cá nhân:** Học sinh có thể nhận lộ trình ôn tập riêng, tập trung vào các môn hoặc kỹ năng mà mô hình dự đoán là điểm yếu.

5.2.2. Tinh chỉnh mô hình

Quá trình tìm kiếm ngẫu nhiên giúp giảm thời gian thử qua từng tham số, đồng thời vẫn đảm bảo khả năng tìm ra bộ siêu tham số mang lại hiệu suất dự đoán cao nhất.

Bộ tham số này được xác định dựa trên kết quả đánh giá chéo (**cross-validation**) với thước đo Accuracy, đảm bảo mô hình vừa đạt độ chính xác cao vừa duy trì khả năng khái quát hóa trên dữ liệu mới.

- **iterations = 500**: Số vòng lặp lớn nhất trong danh sách thử nghiệm. Giúp mô hình học kỹ hơn từ dữ liệu, đồng thời nhờ các tham số điều chỉnh khác nên hạn chế được overfitting.
- **learning_rate = 0.05**: Nằm ở mức trung bình, đảm bảo tốc độ học vừa phải, mô hình ổn định hơn so với 0.1 hoặc 0.2 và nhanh hơn so với 0.01.
- **depth = 5**: Là giá trị tối đa trong dải thử nghiệm [3, 4, 5], phù hợp với dữ liệu ít để mô hình đủ khả năng học mối quan hệ phức tạp nhưng không quá sâu gây overfit.
- **l2_leaf_reg = 9**: Giá trị trung bình trong dải [7, 9, 11], giúp cân bằng giữa khả năng học chi tiết và tránh fitting theo nhiễu.
- **border_count = 64**: Mức trung bình trong [32, 64, 128], đủ chi tiết để phân tách dữ liệu liên tục mà không gây quá nhiều split.
- **rsm = 0.6**: Sử dụng 60% số đặc trưng cho mỗi cây, giúp mô hình đa dạng hóa cây và giảm nguy cơ overfit.
- **random_strength = 2**: Mức tăng nhẹ tính ngẫu nhiên khi chọn split, cải thiện khả năng khái quát của mô hình.
- **bagging_temperature = 1**: Giúp mỗi cây được huấn luyện trên tập dữ liệu hơi khác nhau, chống overfit mà không làm giảm quá nhiều độ chính xác.
- **eval_metric = 'Accuracy'**: Phù hợp với mục tiêu tối ưu tỷ lệ dự đoán đúng trong bài toán phân loại.

5.3. Hướng Phát Triển

- **Mở rộng quy mô dữ liệu:** Thu thập và huấn luyện mô hình trên tập dữ liệu lớn hơn, đến từ nhiều trường học và khu vực khác nhau, nhằm nâng cao khả năng khái quát hóa và giảm thiểu sai số khi áp dụng cho các bối cảnh đa dạng.
- **Ứng dụng xử lý ngôn ngữ tự nhiên (NLP):** Kết hợp phân tích dữ liệu văn bản như nhận xét của giáo viên, phản hồi của học sinh và phụ huynh, để bổ sung thông tin định tính giúp cải thiện độ chính xác của dự đoán.
- **Xây dựng hệ thống dashboard tương tác:** Tích hợp các công cụ như **Power BI** hoặc **Tableau** để trực quan hóa kết quả dự đoán và xu hướng học tập, hỗ trợ ban giám hiệu và giáo viên ra quyết định nhanh chóng.
- **Phát triển chatbot hỗ trợ học tập:** Xây dựng chatbot thông minh có khả năng tư vấn cá nhân hóa cho học sinh, giải đáp thắc mắc về nội dung học, đề xuất tài nguyên học tập và hướng dẫn cải thiện điểm số dựa trên dự đoán của mô hình.

5.4. Hạn Chế

- Dữ liệu huấn luyện giới hạn, chưa phản ánh đầy đủ đa dạng vùng miền và điều kiện học tập.
- Một số yếu tố ảnh hưởng đến kết quả học tập (động lực cá nhân, môi trường gia đình sâu hơn) chưa được thu thập.
- Mô hình chưa được kiểm nghiệm trên dữ liệu thời gian thực, nên độ chính xác trong môi trường triển khai thực tế cần được kiểm chứng thêm.
- Chưa kết hợp phân tích nguyên nhân sâu bằng **SHAP values** để giải thích mô hình ở mức từng học sinh.

TÀI LIỆU KHAM KHẢO

- [1] "Giới thiệu về Machine Learning," [Online]. Available: <https://machinelearningcoban.com/2016/12/26/introduce/>. [Accessed 2025].
- [2] Abhijat_Sarari, "CatBoost Parameters and Hyperparameters - GeeksforGeeks," GeeksforGeeks, Sanchhaya Education Private Limited, [Online]. Available: <https://www.geeksforgeeks.org/machine-learning/catboost-parameters-and-hyperparameters/>. [Accessed 2025].
- [3] Tarandeep_Singh, "Hyperparameter Tuning," [Online]. Available: <https://www.geeksforgeeks.org/machine-learning/hyperparameter-tuning/>. [Accessed 2025].
- [4] Akshi_Saxena, "CatBoost in Machine Learning - GeeksforGeeks," GeeksforGeeks, Sanchhaya Education Private Limited, [Online]. Available: <https://www.geeksforgeeks.org/machine-learning/catboost-ml/>. [Accessed 2025].