

VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING



Đồ án tổng hợp - hướng kỹ thuật dữ liệu (CO3127)

Project

“*Dự đoán chỉ số PISA*”

Instructor(s): Vũ Ngọc Tú

Students: Nguyễn Tuấn Long - 2311915 (*L03-Team 05, Leader*)
Đặng Vũ Anh Khoa - 2311578 (*L03-Team 05*)
Nguyễn Thanh Lâm - 2311824 (*L03-Team 05*)

HO CHI MINH CITY, DECEMBER 2025



Contents

1 Giới thiệu	3
1.1 Giới thiệu về bài toán:	3
1.2 Mô tả bài toán:	3
1.3 Mục tiêu bài toán và lí do chọn:	3
1.3.1 Mục tiêu Bài toán (Goals and Objectives)	3
1.3.1.1 Mục tiêu chung:	3
1.3.1.2 Mục tiêu cụ thể:	4
1.3.2 Lý do Lựa chọn Bài toán (Motivation and Rationale)	4
1.3.2.1 Ý nghĩa về Mặt Chính sách và Xã hội (Relevance)	4
1.3.2.2 Ý nghĩa về Mặt Dữ liệu và Khoa học (Feasibility and Novelty) .	4
1.3.2.3 Khả năng Mở rộng và Phát triển (Impact)	5
2 Giới thiệu về phương pháp:	5
2.1 Giới thiệu về Gradient boosting:	5
2.1.1 Tổng quan về thuật toán:	5
2.1.2 Thuật toán của Gradient boosting:	5
2.2 Giới thiệu về LightGBM:	7
2.2.1 Giới thiệu sơ qua về LightGBM:	7
2.3 Thuật toán của LightGBM	7
2.4 Giới thiệu về XGBoost:	8
2.5 Thuật toán của Xgboost:	9
3 Chi tiết về dữ liệu	11
3.1 Thông tin chung	11
3.2 Nội dung các cột	11
3.3 Kết quả EDA	12
3.3.1 Tổng quan về Bộ dữ liệu (Dataset Overview)	12
3.3.2 Phân tích Tương quan (Correlation Analysis)	12
3.3.3 Phân tích Phân phối Biến mục tiêu (Target Analysis)	12
3.3.4 Phân tích theo Giới tính và Thời gian (Gender & Time Analysis)	13
3.3.5 Phân tích các Đặc trưng khác (Feature Specific EDA)	14
3.4 Tiền xử lý	14
4 Ứng dụng vào bài toán	16
4.1 Áp dụng LightGBM	16
4.2 Áp dụng XGBoost	16
5 Tính toán VD mẫu số của 2 model	18
5.1 Tính toán VD của model XGBoost	18
5.1.1 Ví dụ mẫu 1	21
5.1.2 Ví dụ mẫu 2	22
5.1.3 Ví dụ mẫu 3	23
5.2 Tính toán VD của model LightGBM	25
5.2.1 Ví dụ mẫu 1	27
5.2.2 Ví dụ mẫu 2	27
5.2.3 Ví dụ mẫu 3	28



6 Dashboard	29
6.1 Data Visualization	29
6.2 Model Training	31
7 Phần code	32
8 Phần kết	33
8.1 Kết luận	33
8.2 Lời cảm ơn	33

1 Giới thiệu

1.1 Giới thiệu về bài toán:

Chúng ta dựa vào dữ liệu về giáo dục, xã hội và kinh tế của các nước khác nhau của các năm khác nhau và giới tính khác nhau để chúng ta có thể xây dựng mô hình để từ đó suy ra được chỉ số PISA của các nước dựa trên dữ liệu về các khía cạnh trên (với PISA là chỉ số quốc tế dùng để đo lường khả năng đọc, tư duy về toán và khoa học của đứa trẻ 15 tuổi).

1.2 Mô tả bài toán:

Bài toán nghiên cứu này thuộc lĩnh vực Khoa học Dữ liệu (Data Science) và Học máy (Machine Learning), cụ thể là Bài toán Hồi quy (Regression Problem). Mục tiêu cốt lõi là xây dựng một mô hình toán học nhằm định lượng và dự đoán sự thay đổi của Chỉ số PISA dựa trên các biến số vĩ mô của quốc gia.

Phạm vi dữ liệu cho phép mô hình nhìn nhận toàn diện về các điều kiện tác động đến chất lượng giáo dục:

- Dữ liệu đa quốc gia: Nghiên cứu không giới hạn ở một quốc gia mà bao gồm dữ liệu từ nhiều nền kinh tế khác nhau về cách thu thuế của các nước khác nhau và độ chênh lệch giữa người giàu và người nghèo, cho phép mô hình học được các mối quan hệ tổng quát và so sánh các chiến lược phát triển.
- Dữ liệu đa thời gian: Việc sử dụng dữ liệu từ các năm khác nhau (ví dụ: các kỳ thi PISA 2000, 2003, 2006,...) giúp mô hình nắm bắt được xu hướng và sự tiến triển của Chỉ số PISA theo thời gian và sự thay đổi của các biến vĩ mô.
- Phân tích nhân khẩu học: Việc đưa Giới tính vào làm biến đầu vào hoặc làm biến phân nhóm cho phép mô hình không chỉ dự đoán điểm PISA chung mà còn phân tích sự chênh lệch hiệu suất giữa học sinh nam và học sinh nữ, làm rõ các yếu tố ảnh hưởng đặc thù.
- Dữ liệu về kinh tế và giáo dục : Việc đưa giới tính và lượng tiền đầu tư của nhà nước vào giáo dục , thuế để có thể cho thấy tính quan trọng và tầm ảnh hưởng của nó vào PISA.
- Dữ liệu về các vấn đề xã hội : tỉ lệ tử vong , dân cư ở vùng quê .. cũng là một phần trong việc có thể ảnh hưởng đến chất lượng giáo dục

Dựa trên các biến trên để có thể từ đó chúng ta xây dựng được mô hình để từ đó chúng ta có thể tính toán ra được chỉ số PISA của những nước khác ở các thời điểm khác nhau dựa trên những thông tin ở trên.

1.3 Mục tiêu bài toán và lí do chọn:

1.3.1 Mục tiêu Bài toán (Goals and Objectives)

Mục tiêu bài toán được chia thành hai nhóm chính: mục tiêu chung (kết quả mong muốn) và mục tiêu cụ thể (các bước đạt được kết quả).

1.3.1.1 Mục tiêu chung: Mục tiêu chung của nghiên cứu là xây dựng thành công một mô hình dự đoán định lượng có độ chính xác cao để ước lượng Chỉ số PISA của các quốc gia dựa trên các chỉ số vĩ mô về kinh tế, xã hội, và giáo dục.



1.3.1.2 Mục tiêu cụ thể:

- Xây dựng Tập Dữ liệu Đa chiều: Thu thập, làm sạch và chuẩn hóa tập dữ liệu đa chiều, đa quốc gia (cross-sectional và time-series) bao gồm các biến về Kinh tế, Xã hội, Giáo dục, và Nhân khẩu học (Giới tính) để sẵn sàng cho quá trình mô hình hóa.
- Lựa chọn và Huấn luyện Mô hình: Áp dụng và so sánh hiệu suất của các thuật toán học máy (như Hồi quy Tuyến tính, Cây Quyết định, Random Forest, hoặc Gradient Boosting) để tìm ra mô hình dự đoán Chỉ số PISA tối ưu nhất.
- Phân tích Tác động: Định lượng hóa mức độ ảnh hưởng của từng nhóm biến (Kinh tế, Xã hội, Giáo dục) đến kết quả PISA. Từ đó, xác định đâu là yếu tố then chốt (Key Drivers) chi phối chất lượng giáo dục quốc gia.
- Phân tích Chênh lệch Giới tính: Phân tích sự khác biệt về điểm PISA dựa trên giới tính và xác định liệu các yếu tố vĩ mô có ảnh hưởng khác nhau đến kết quả của học sinh nam và học sinh nữ hay không.
- Đề xuất Thực tiễn: Dựa ra các kiến nghị dựa trên dữ liệu cho các nhà hoạch định chính sách giáo dục nhằm tối ưu hóa nguồn lực và cải thiện điểm PISA trong tương lai.

1.3.2 Lý do Lựa chọn Bài toán (Motivation and Rationale)

Việc lựa chọn bài toán dự đoán Chỉ số PISA là cần thiết và có ý nghĩa lớn về mặt khoa học và thực tiễn, xuất phát từ các lý do sau:

1.3.2.1 Ý nghĩa về Mật Chính sách và Xã hội (Relevance)

- PISA là Chuẩn mực Quốc tế: Chỉ số PISA được coi là thước đo đáng tin cậy về vốn nhân lực (Human Capital) và khả năng cạnh tranh kinh tế toàn cầu. Việc hiểu rõ cách đạt được điểm PISA cao là chìa khóa cho sự phát triển bền vững.
- Khoảng cách Giữa Nguồn lực và Kết quả: Nhiều quốc gia đầu tư lớn vào giáo dục nhưng kết quả PISA không tương xứng. Bài toán này giúp khám phá các biến số ẩn (hidden variables) ngoài chỉ tiêu đơn thuần (như chất lượng quản lý, văn hóa học tập, hoặc bất bình đẳng xã hội) đang thực sự quyết định kết quả.

1.3.2.2 Ý nghĩa về Mật Dữ liệu và Khoa học (Feasibility and Novelty)

- Tính Đa chiều của Dữ liệu: Bài toán tận dụng được sự phong phú của dữ liệu vĩ mô (từ Ngân hàng Thế giới, UNESCO, OECD, v.v.) kết hợp với dữ liệu nhân khẩu học (giới tính) và thời gian, tạo nên một tập dữ liệu mạnh mẽ cho các kỹ thuật Học máy phức tạp.
- Ứng dụng Kỹ thuật Dự đoán: Đây là cơ hội để áp dụng và so sánh các mô hình Hồi quy hiện đại, không chỉ dừng lại ở việc dự đoán mà còn tập trung vào khả năng giải thích (Interpretability) của mô hình để xác định nguyên nhân.

1.3.2.3 Khả năng Mở rộng và Phát triển (Impact)

- Tính Tổng quát: Mô hình được xây dựng trên dữ liệu đa quốc gia sẽ có tính tổng quát cao, có thể được áp dụng như một công cụ tham chiếu (benchmark) cho các quốc gia khác nhau.
- Hỗ trợ Cải cách: Kết quả nghiên cứu sẽ cung cấp cơ sở dữ liệu vững chắc để chứng minh rằng việc cải thiện các yếu tố xã hội và kinh tế (như giảm bất bình đẳng) có thể mang lại hiệu quả giáo dục tương đương hoặc lớn hơn việc chỉ tăng chi tiêu ngân sách.

2 Giới thiệu về phương pháp:

Với việc project này sử dụng 2 phương pháp là LightGBM và XGBoost thì cả 2 phương pháp này đều sử dụng chung cho mình một kiểu thuật toán học máy là gradient boosting . Chúng ta sẽ tìm hiểu về gradient boosting.

2.1 Giới thiệu về Gradient boosting:

2.1.1 Tổng quan về thuật toán:

Gradient boosting là một thuật toán tăng cường với việc kết hợp nhiều mô hình dự đoán yếu hơn để tạo thành một thể thống nhất. Các mô hình dự đoán yếu thường là các decisions được train liên tục để giảm thiểu loss function và tăng độ chính xác. Bằng cách kết hợp nhiều decision tree hồi quy và decision tree phân loại, gradient boosting có thể nắm bắt một cách hiệu quả các mối liên hệ phức tạp giữa các thông số.

Việc sử dụng gradient boosting đem lại lợi ích. Một trong những lợi ích lớn nhất là việc nó có khả năng giảm thiểu các loss function và dẫn đến cải thiện chính xác trong việc dự đoán. Tuy nhiên, việc này cũng dẫn đến việc là chúng ta phải lưu ý về overfitting, nó có thể xảy ra khi một mô hình trở nên chuyên biệt với dữ liệu được train và dẫn đến thất bại trong việc tạo ra các phiên bản tốn hơn. Ngoài ra, nó còn có thể giảm thiểu độ lệch của mô hình, có thể được sử dụng để xử lý các dữ liệu phức tạp và nó cũng đem lại hạn chế như việc tốn thời gian huấn luyện, khó khăn trong điều chỉnh các siêu tham số để tìm ra sự kết hợp tối ưu, nhạy cảm với các dữ liệu nằm ngoài (outliers).

Gradient boosting bắt đầu với việc sử dụng tập data training để thiết lập nền tảng cho mô hình, thường là các decision tree, thường được dựa trên các dự đoán ngẫu nhiên. Thường được chọn dựa trên tính dễ diễn giải, mô hình yếu hoặc nền tảng để sử dụng điểm khởi đầu một các tối ưu. Bước khởi tạo được dùng để xây dựng cho các vòng lặp tiếp theo. Sau đó chúng ta sẽ tính các sai số của dự đoán so với dữ liệu được train, tinh chỉnh bằng các sử dụng chính quy hóa. Tiếp theo, chúng ta sẽ train với mô hình tiếp theo dựa trên các sai số đã tính trước đó, chúng sẽ được tinh chỉnh sao cho phù hợp kết quả. Cập nhật tập hợp các decision tree và lặp lại cho đến khi điều kiện dừng đạt được (như số lần lặp lại tối đa, độ chính xác hay hiệu xuất giảm dần). Điều này giúp đảm bảo rằng dự đoán cuối cùng của mô hình đạt được sự cân bằng mong đợi giữa độ phức tạp và hiệu suất.

2.1.2 Thuật toán của Gradient boosting:

Chúng ta xét mô hình F để dự đoán giá trị $\hat{y} = F(x)$ để chúng ta có thể tối ưu hóa mô hình trên thì chúng ta cần giảm giá trị sai số toàn phương trung bình của mô hình, chúng ta xét các mô

hình học yếu hơn ở các bước m trong quá trình là F_m trong M bước mà mô hình cần thực hiện tông quá trình học dữ liệu. Để tối ưu hóa F_m thì chúng ta cần thêm một ước lượng $h_m(x)$. Hay nói cách khác là $h_m(x)$ sẽ là thứ mà chúng ta cần tối ưu. Do đó, mỗi bước sau đó của quá trình học sẽ tìm cách có thể tối ưu được phần dư đó và cũng xuất phát từ việc quan sát được phần dư đó tỷ lệ thuận với các gradient âm của sai số toàn phương trung bình. Từ đó, Gradient boosting dựa vào thuật toán giảm độ gradient bằng cách thay thế một loss function mới và gradient chính nó.

Các bước của thuật toán

- Bước khởi tạo: Cho một hệ thống chứa một biến output hoặc response y và tập hợp ngẫu nhiên của biến $x = x_1, x_2, \dots, x_n$ với mục tiêu là thu được một ước lượng hoặc xấp xỉ $\hat{F}(x)$, của hàm số $F^*(x)$ từ x qua y sao giảm thiểu giá trị kỳ vọng của một hàm mất mát $L(y, F(x))$ được chỉ định trên phân phối đồng thời của tất cả các giá trị (y, x) .

$$F^*(x) = \arg \min_F E_{x,y} L(y, F(x))$$

- Bước lặp lại cho $m = 1$ đến M (số lượng cây/mô hình tổ hợp):

- Tính toán pseudo-residuals r_{im} cho mỗi mẫu i bằng cách lấy gradient âm của hàm mất mát đối với giá trị dự đoán hiện tại cũng là "hướng dốc nhất" để giảm sai số

$$r_{im} = - \left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)} \right]_{F(x)=F_{m-1}(x)}$$

Trong hồi quy với MSE, hàm mất mát là $L(y, F) = \frac{1}{2}(y - F)^2$. Đạo hàm là $\frac{\partial L}{\partial F} = -(y - F)$. Do đó, Pseudo-Residuals chính là Phần dư thực tế (Actual Residuals):

$$r_{im} = -[-(y_i - F_{m-1}(x_i))] = y_i - F_{m-1}(x_i)$$

- Huấn luyện một mô hình cơ sở mới (thường là cây quyết định) $h_m(x)$ để dự đoán các pseudo-residuals r_{im} . Mô hình $h_m(x)$ được huấn luyện trên cặp dữ liệu (x_i, r_{im}) . Cây quyết định này chia gian dầu vào thành J_m vùng (leaves), R_{jm} . Giá trị γ_{jm} là bước nhảy tốt nhất tại vùng lá đó.
 - Tính toán giá trị đầu ra γ_{jm} (hệ số) tối ưu cho mỗi vùng lá R_{jm} của cây $h_m(x)$ để tối thiểu hóa hàm mất mát:

$$\gamma_{jm} = \operatorname{argmin}_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma)$$

- Cập nhật mô hình tổ hợp hiện tại $F_m(x)$ bằng cách thêm mô hình mới $h_m(x)$, có áp dụng Learning Rate (Tốc độ học) ν :

$$F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} \cdot \mathbf{I}(x \in R_{jm})$$

Với ν là tham số điều chỉnh (thường là $0 < \nu \leq 1$) giúp kiểm soát tốc độ học, giảm nguy cơ overfitting và cải thiện khả năng tổng quát hóa. $\mathbf{I}()$ là hàm chỉ thị, trả về 1 nếu điều kiện đúng, 0 nếu sai.

- Sau khi hoàn thành M vòng lặp, mô hình dự đoán cuối cùng là tổng của tất cả các mô hình cơ sở:

$$F_M(x) = F_0(x) + \nu \sum_{m=1}^M \sum_{j=1}^{J_m} \gamma_{jm} \cdot \mathbf{I}(x \in R_{jm})$$

2.2 Giới thiệu về LightGBM:

2.2.1 Giới thiệu sơ qua về LightGBM:

LightGBM là một gradient boosting framework sử dụng cây dựa trên thuật toán học máy. Nó được thiết kế để sử dụng với các lợi ích như tăng tốc độ train với hiệu quả cao hơn, ít tốn bộ nhớ, có độ chính xác cao hơn, hỗ trợ trong việc chạy song song, đơn và học máy GPU, có thể sử dụng để học các data có scale lớn.

Điểm khác của LightGBM so với Gradient boosting là thay vì phát triển theo level, tức là chia tách tất cả các nút (node) ở cùng một cấp độ trước khi chuyển sang cấp độ tiếp theo có thể giúp kiểm soát độ phức tạp tốt hơn và giảm nguy cơ overfitting thì LightGBM phát triển theo chiều lá, tìm lá (leaf) có độ giảm mất mát (loss reduction) lớn nhất trong toàn bộ cây và thực hiện chia tách tại lá đó, giúp cây sâu hơn và không cân bằng, đem lại lợi ích trong việc giảm mất mát nhanh hơn, dẫn đến độ chính xác cao hơn với cùng số lượng lá, và hội tụ nhanh hơn. Tuy nhiên nó cũng có hạn chế là dễ bị overfitting hơn trên các tập dữ liệu nhỏ do cấu trúc cây không cân bằng.

Ngoài ra, LightGBM sử dụng cách tiếp cận hoàn toàn khác để tìm điểm chia tách tối ưu đó là thuật toán dựa trên Histogram. Thay vì duyệt qua tất cả các giá trị đặc trưng liên tục để tìm điểm chia tách, LightGBM phân nhóm (bin) các giá trị đặc trưng thành các thùng (bin) rời rạc, tạo ra một biểu đồ (histogram). Việc tìm kiếm điểm chia tách chỉ cần duyệt qua các giá trị trong histogram (số lượng bin thường cố định và nhỏ, ví dụ 256), thay vì duyệt qua tất cả các điểm dữ liệu. Điều này giúp tăng tốc độ huấn luyện đáng kể (khoảng 8 lần so với phương pháp truyền thống) và giảm mức sử dụng bộ nhớ vì chỉ cần lưu trữ các giá trị bin.

2.3 Thuật toán của LightGBM

LightGBM có điều đặc biệt so với Gradient Boosting ở điểm sau khi đã tính toán pseudo-residuals r_{im} cho mỗi mẫu i thì thuật toán có thêm bước lấy mẫu với GOSS (Gradient-based One-Side Sampling) và gom bó Đặc trưng với EFB (Exclusive Feature Bundling):

- Bước lấy mẫu với GOSS (Gradient-based One-Side Sampling) : chia tập dữ liệu thành hai tập con dựa trên giá trị tuyệt đối của gradient (tức là $|r_{im}|$):
 - Tập A (High Gradient): Các mẫu có $|r_{im}|$ lớn nhất (ví dụ: top $a \times 100\%$ các mẫu). Giữ lại tất cả các mẫu trong tập này.
 - Tập B (Low Gradient): Các mẫu còn lại. Lấy mẫu ngẫu nhiên từ tập này với tỷ lệ b . Tạo tập huấn luyện \tilde{D} bằng cách kết hợp A và $B_{sampled}$.

Gán trọng số W_i cho các mẫu được lấy mẫu từ B để bù đắp:

$$W_i = \begin{cases} 1, & i \in A \\ \frac{1-a}{b}, & i \in B_{sampled} \end{cases}$$

- Trước khi xây dựng cây, áp dụng EFB để giảm số lượng đặc trưng:
 - Xác định các đặc trưng thừa và độc quyền (ít khi khác 0 cùng lúc).
 - Gom các đặc trưng này thành các bó đặc trưng (Feature Bundles) để giảm chiều dữ liệu.

- Và khi xây dựng cây sẽ sử dụng Thuật toán Dựa trên Biểu đồ (Histogram-based) được dùng để phân loại các giá trị đặc trưng thành các bin để tăng tốc độ tìm điểm phân chia và chiến lược Leaf-wise Growth để có thể chọn lá có mức giảm mất mát lớn nhất để phân chia tiếp, tạo ra các cây sâu, không đối xứng.
- Ngoài ra, khi tính toán hệ số lá và cập nhật mô hình sẽ sử dụng tính toán giá trị đầu ra γ_{jm} (hệ số tối ưu) cho mỗi vùng lá R_{jm} (với trọng số W_i):

$$\gamma_{jm} = \operatorname{argmin}_{\gamma} \sum_{x_i \in R_{jm}} W_i \cdot \frac{1}{2} (y_i - (F_{m-1}(x_i) + \gamma))^2$$

Với MSE, γ_{jm} là trung bình của các r_{im} trong vùng lá đó (đã nhân với trọng số GOSS):

$$\gamma_{jm} = \frac{\sum_{x_i \in R_{jm}} W_i \cdot r_{im}}{\sum_{x_i \in R_{jm}} W_i}$$

Cập nhật mô hình tổ hợp hiện tại $F_m(x)$ bằng cách thêm mô hình mới với Learning Rate ν

$$F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} \cdot \mathbf{I}(x \in R_{jm})$$

- Mô hình dự đoán cuối cùng là tổng của tất cả M cây:

$$F_M(x) = F_0(x) + \nu \sum_{m=1}^M h_m(x)$$

2.4 Giới thiệu về XGBoost:

XGBoost là một gradient boosting framework sử dụng cây quyết định làm mô hình con. Thay vì tạo mô hình tốt nhất có thể trên dữ liệu, nó đào tạo hàng nghìn mô hình trên nhiều tập con khác nhau của tập dữ liệu đào tạo và sau đó bỏ phiếu cho mô hình có hiệu suất tốt nhất. Nó được mở rộng và tối ưu Gradient Boosting truyền thống để nhanh hơn, chính xác hơn đồng thời tránh được overfitting tốt hơn. Nó là tập kết hợp nhiều mô hình nhỏ thành một mô hình mạnh hơn, mỗi mô hình mới sinh ra để sửa lỗi của mô hình trước. Nếu mô hình có dự đoán sai mẫu nào thì mô hình kế tiếp sẽ chú ý nhiều hơn vào mẫu đó.

Điểm khác giữa XGBoost và Gradient Boosting là nó bổ sung regularization (L1 Lasso và L2 Ridge) trực tiếp vào mục tiêu tối ưu, giúp kiểm soát độ phức tạp của cây và giảm hiện tượng overfitting. Bên cạnh đó nó còn được tối ưu tốc độ bằng đa luồng tính toán song song giúp tăng tốc độ quá trình huấn luyện, học nhận thức thưa (sparse-aware), hỗ trợ dữ liệu lớn và có khả năng xử lý giá trị thiếu một cách tự động.

Cách làm của XGBoost vẫn có ưu tiên lever-wise, nhưng có cắt tỉa (pruning) sau đó. Ban đầu là xây cây theo chiều sâu tạm thời (depth-first) sau đó tính toán Gain (Hiệu quả cải thiện loss) từng nhánh tiếp đến cắt bỏ các nhánh Gain nhỏ hơn 0 và chỉ giữ lại nhánh hiệu quả tạo cây tối ưu và gọn.

Tuy nhiên XGBoost vẫn có nhược điểm là khó điều chỉnh, việc tối ưu hóa tham số trong XGBoost có thể khó khăn và cần tham khảo nhiều. Đồng thời đây không phải là mô hình trực quan và

không dễ giải thích như một số thuật toán học máy khác. Dù sao thì XGBoost là một trong những thuật toán học máy mạnh mẽ và phổ biến nhất hiện nay. Với khả năng xử lý dữ liệu lớn, cải tiến tính toán song song, và khả năng điều chỉnh giúp tránh overfitting, XGBoost đã được áp dụng rộng rãi trong nhiều lĩnh vực như tài chính, y tế, marketing, và giao thông. Tuy nhiên, việc tối ưu hóa các tham số của XGBoost có thể yêu cầu kinh nghiệm và hiểu biết sâu sắc về thuật toán. Tuy vậy, khi được sử dụng đúng cách, XGBoost mang lại những kết quả ấn tượng và hiệu quả cho nhiều bài toán trong khoa học dữ liệu.

2.5 Thuật toán của Xgboost:

Thuật toán của Xgboost có điểm nâng cấp so với Gradient boosting ở chỗ sử dụng Gradient Bậc Hai để tối ưu hóa và áp dụng các thuật ngữ điều chỉnh (regularization) trực tiếp vào cấu trúc cây

- Bước khởi tạo: Đầu tiên là hàm măt măt và điều chỉnh

- Hàm Măt măt Hồi quy (MSE):

$$L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$

- Hàm Điều chỉnh (Regularization):

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^T \omega_j^2$$

- T: Số lá (nodes) trong cây.
 - ω_j : Giá trị đầu ra của lá j .
 - γ : Chi phí (phạt) cho việc thêm một lá mới.
 - λ : Tham số điều chỉnh L2.

- Khởi tạo mô hình dự đoán ban đầu $F_0(\mathbf{x})$ bằng một hằng số, thường là giá trị trung bình của mục tiêu:

$$F_0(x) = \text{avg}(y)$$

- Vòng lặp xây dựng Cây : Lặp lại cho $t = 1$ đến M (số lượng cây):

- Tính toán Gradient bậc nhất (g_i) và Gradient bậc hai (h_i) cho mỗi mẫu i , dựa trên dự đoán hiện tại $F_{t-1}(x_i)$:

- * Gradient bậc nhất (g_i):

$$g_i = \left[\frac{\partial L}{\partial \hat{y}} \right]_{\hat{y}=F_{t-1}(x_i)} = -(y_i - F_{t-1}(x_i))$$

g_i là Phản dư (Residual) thực tế nhưng mang dấu âm.

- * Gradient bậc hai (h_i):

$$h_i = \left[\frac{\partial^2 L}{\partial \hat{y}^2} \right]_{\hat{y}=F_{t-1}(x_i)} = 1$$

- * Đặc điểm quan trọng của MSE: h_i luôn là 1 (hằng số).

- Sử dụng thuật toán tìm kiếm Greedy để tìm cấu trúc cây (các điểm phân chia) tối ưu bằng cách tối đa hóa độ lợi (Gain) tại mỗi lần phân chia. Độ lợi (Gain) cho một lần phân chia:

$$\text{Gain} = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma$$

* $G_L = \sum_{i \in I_L} g_i$ (Tổng gradient bậc nhất của tập con trái).

* $H_L = \sum_{i \in I_L} h_i$ (Tổng gradient bậc hai của tập con trái).

* G_R, H_R : Tương tự cho tập con phải.

- Quy tắc dừng: Việc phân chia dừng lại khi Gain ≤ 0 (hoặc khi đạt đến độ sâu/số mẫu tối đa cho phép).

- Sau khi cây f_t được xây dựng, tính toán giá trị đầu ra ω_j^* tối ưu cho mỗi lá j dựa trên các mẫu I_j rơi vào lá đó:

$$\omega_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$$

- Vì $h_i = 1$ trong MSE, công thức đơn giản hóa thành:

$$\omega_j^* = \frac{\sum_{i \in I_j} (y_i - F_{t-1}(x_i))}{|I_j| + \lambda}$$

- Đây là trung bình của phần dư trong lá, được điều chỉnh bởi λ . Cập nhật mô hình dự đoán tổng hợp $F_t(x)$ bằng cách thêm cây mới $f_t(x)$, có áp dụng Tốc độ học (Learning Rate) ν :

$$F_t(x) = F_{t-1}(x) + \nu \cdot f_t(x)$$

- Mô hình dự đoán cuối cùng sau M lần lặp là tổng có trọng số của tất cả các cây:

$$F_M(x) = F_0(x) + \nu \sum_{t=1}^M f_t(x)$$

3 Chi tiết về dữ liệu

3.1 Thông tin chung

- Nguồn (đã bị xóa): [Kaggle: PISA Results 2000-2022 \(Economics and Education\)](#)
- Nguồn dự phòng: [Github](#)
- Tác giả: Walasse Tomaz
- Thời gian đăng tải: 2023
- Định dạng: csv (20 cột x 634 dòng)
- Nội dung chính: gồm chỉ số PISA và các thông tin liên quan đến kinh tế và giáo dục của 39 quốc gia trong khoảng thời gian từ năm 2003 đến năm 2018 (2003, 2006, 2009, 2012, 2015, 2018)

3.2 Nội dung các cột

- `index_code`: Mã, gồm tên quốc gia và năm khảo sát.
- `expenditure_on_education_pct_gdp`: Chi tiêu cho giáo dục tính theo phần trăm GDP.
- `mortality_rate_infant`: Tỷ lệ tử vong trẻ sơ sinh.
- `gini_index`: Chỉ số Gini (biểu thị độ bất bình đẳng trong thu nhập của một đất nước, nó có giá trị từ 0 (đều có thu nhập bình đẳng) đến 1 (bất bình đẳng)).
- `gdp_per_capita_ppp`: GDP bình quân đầu người theo sức mua tương đương.
- `inflation_consumer_prices`: Lạm phát giá tiêu dùng.
- `intentional_homicides`: Tội cố ý giết người.
- `unemployment`: Tỷ lệ thất nghiệp.
- `gross_fixed_capitalFormation`: Tổng hình thành vốn cố định theo phần trăm GDP.
- `population_density`: Mật độ dân số.
- `suicide_mortality_rate`: Tỷ lệ tử vong do tự tử.
- `tax_revenue`: Doanh thu thuế.
- `taxes_on_income_profits_capital`: Thuế thu nhập, lợi nhuận và lợi tức vốn.
- `alcohol_consumption_per_capita`: Tổng mức tiêu thụ rượu bình quân đầu người.
- `government_health_expenditure_pct_gdp`: Chi tiêu y tế của chính phủ tính theo phần trăm GDP.
- `urban_population_pct_total`: Tỷ lệ dân số đô thị trên tổng dân số.
- `country`: Quốc gia.
- `time`: Năm.
- `sex`: Giới tính.
- `rating`: Giá trị của chỉ số PISA.

3.3 Kết quả EDA

3.3.1 Tổng quan về Bộ dữ liệu (Dataset Overview)

- Kiểm tra giá trị thiếu (Missing Values):** Một số cột có lượng dữ liệu thiếu đáng kể, cụ thể:

- `alcohol_consumption_per_capita`: Thiếu nhiều nhất (chỉ có dữ liệu năm 2015).
- `gini_index`, `intentional_homicides`, `tax_revenue`: Cũng chứa các giá trị NaN cần xử lý.

3.3.2 Phân tích Tương quan (Correlation Analysis)

Để hiểu mối quan hệ giữa các biến số thực (numeric variables) và biến mục tiêu `rating`, một ma trận tương quan (Correlation Matrix) đã được xây dựng. Biểu đồ nhiệt (Heatmap) dưới đây hiển thị hệ số tương quan Pearson giữa các đặc trưng.

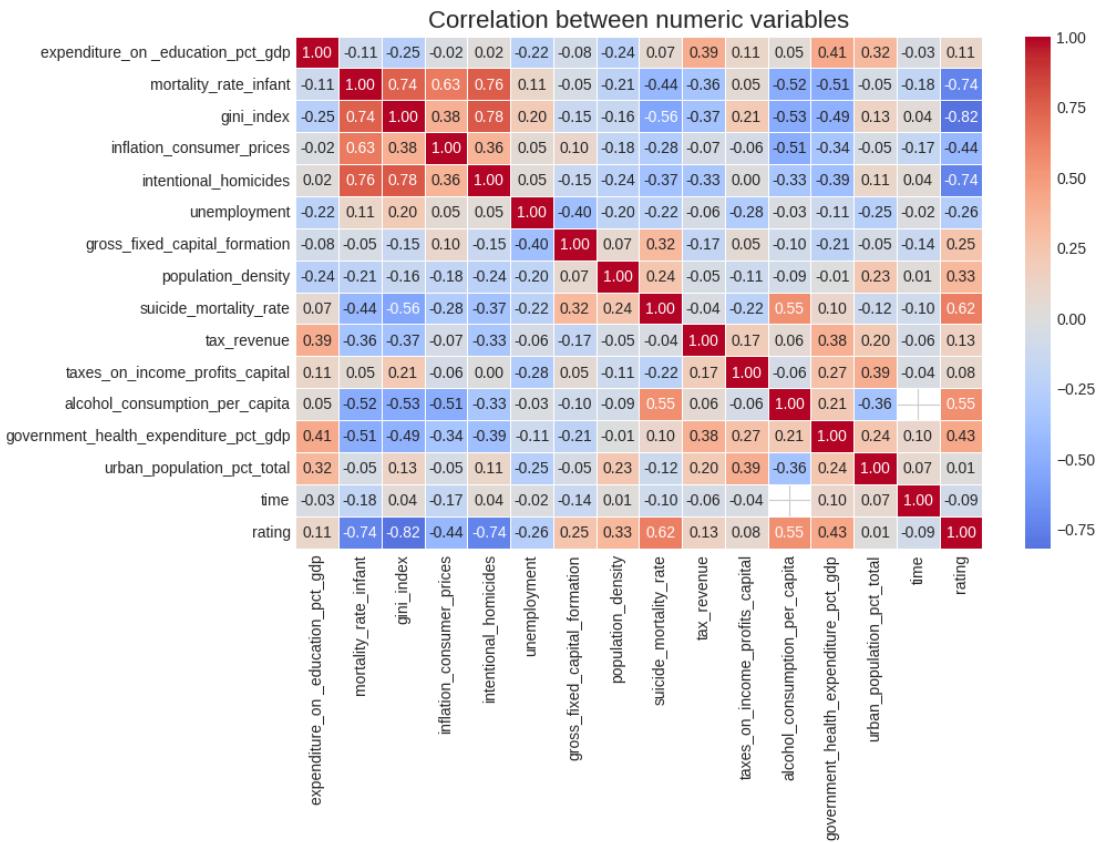


Figure 1: Ma trận tương quan giữa các biến số (Correlation Heatmap)

3.3.3 Phân tích Phân phối Biến mục tiêu (Target Analysis)

Biến mục tiêu `rating` (điểm đánh giá PISA) được phân tích phân phối để kiểm tra độ lệch và hình dạng.

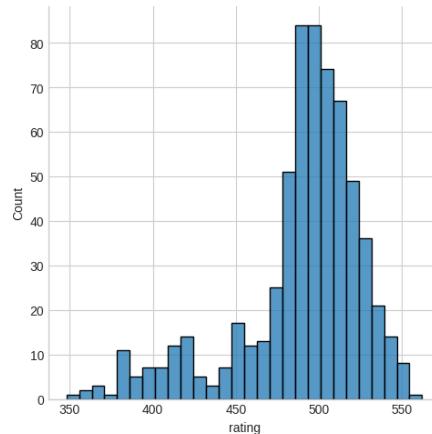


Figure 2: Phân phối của điểm đánh giá PISA (Rating Distribution)

3.3.4 Phân tích theo Giới tính và Thời gian (Gender & Time Analysis)

Chúng tôi đã thực hiện phân tích sâu hơn về sự khác biệt điểm số giữa các giới tính (BOY, GIRL, TOT):

1. **Phân phối theo giới tính:** Sử dụng Violin Plot để so sánh mật độ phân phối điểm số giữa nam và nữ.
2. **Xu hướng theo thời gian:** Biểu đồ đường (Line plot) thể hiện sự thay đổi của rating qua các năm, phân tách theo giới tính.
3. **Phân tích từng quốc gia:** Sử dụng FacetGrid để vẽ Boxplot cho từng quốc gia, giúp nhận diện ngoại lệ (outliers) cụ thể theo vùng lãnh thổ.



Figure 3: Biểu đồ Violin phân phối điểm số theo giới tính

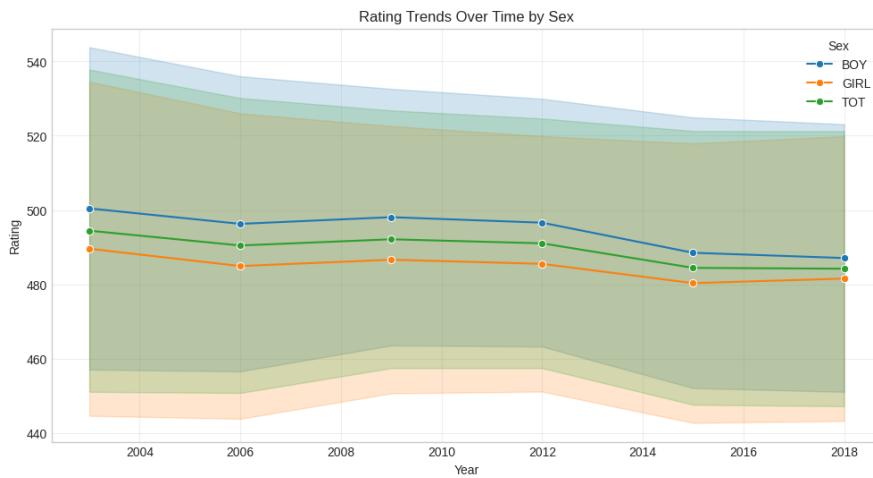


Figure 4: Xu hướng điểm số theo thời gian (Rating Trends Over Time)

3.3.5 Phân tích các Đặc trưng khác (Feature Specific EDA)

Một số đặc trưng cụ thể đã được kiểm tra chi tiết:

- **Chi tiêu giáo dục (expenditure_on_education_pct_gdp):** Kiểm tra phân phối.
- **Thuế (tax_revenue & taxes_on_income...):** Biểu đồ KDE (Kernel Density Estimate) được vẽ để so sánh phân phối của hai loại thuế này. Kết quả cho thấy sự chồng lấp và khác biệt về mật độ phân phối.
- **Tiêu thụ rượu (alcohol_consumption_per_capita):** Chỉ có dữ liệu trong năm 2015, do đó biến này sau đó đã được loại bỏ.

3.4 Tiền xử lý

Quá trình làm sạch và chuẩn bị dữ liệu bao gồm các bước sau:

1. **Loại bỏ cột (Drop Columns):** Loại bỏ cột `index_code` (không mang ý nghĩa dự báo) và `alcohol_consumption_per_capita` (do quá nhiều giá trị thiếu).
2. **Mã hóa dữ liệu (Encoding):**
 - **One-Hot Encoding:** Áp dụng cho biến `sex`, tạo ra các cột nhị phân `sex_BOY`, `sex_GIRL`, `sex_TOT`.
 - **Label Encoding:** Áp dụng cho các biến phân loại dạng chuỗi (object columns) còn lại để chuyển về dạng số.
3. **Điền giá trị thiếu (Imputation):** Sử dụng thuật toán **KNN Imputer** với tham số `n_neighbors = 7` để điền các giá trị thiếu dựa trên sự tương đồng giữa các mẫu dữ liệu, thay vì chỉ điền bằng trung bình.
4. **Phân nhóm mục tiêu (Target Binning):** Biến liên tục `rating` được chia thành 9 nhóm (từ 1 đến 9) thông qua hàm `rating_group` dựa trên các ngưỡng điểm (ví dụ: ≤ 357 , $357 - 410$, v.v.). Mục đích là để thực hiện phân chia tập train/test cân bằng (stratified split).

5. **Chia tập dữ liệu (Train-Test Split):** Dữ liệu được chia thành tập huấn luyện (Train) và kiểm tra (Test) với tỷ lệ **94:6**. Việc chia mẫu được thực hiện phân tầng (stratified) dựa trên biến `rating_groups` để đảm bảo phân phối điểm số tương đồng giữa hai tập.

- Kích thước tập Train: ~ 595 mẫu.
- Kích thước tập Test: ~ 39 mẫu.

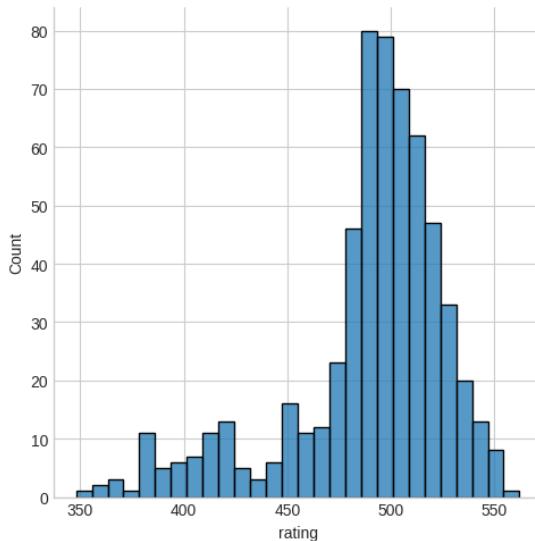


Figure 5: Phân phối của biến mục tiêu trên tập Train

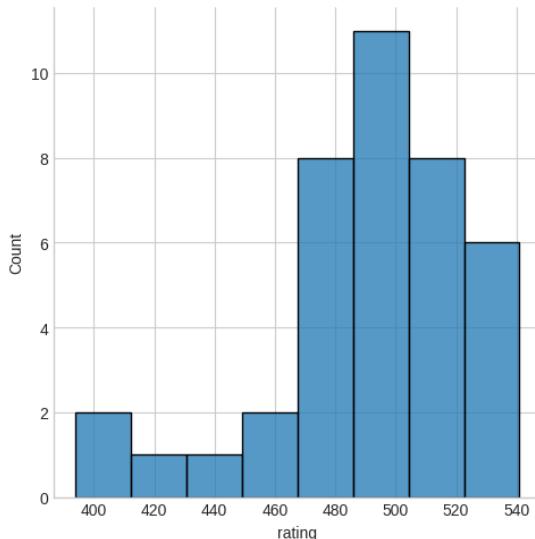


Figure 6: Phân phối của biến mục tiêu trên tập Test



4 Ứng dụng vào bài toán

4.1 Áp dụng LightGBM

1. Khởi tạo mô hình

Mô hình bắt đầu với dự đoán

$$F_0(x) = y_0(x)$$

Với $y_0(x)$ là giá trị trung bình khi cây thực hiện decision tree lần đầu

2. Tính toán sai số:

Sau khi đã thêm cây vào mô hình dự đoán chúng ta sẽ tính toán phần dư:

$$h_m(x) = F_m(x) - y(x),$$

với $h_m(x)$ là sự mất mát của mô hình ở bước thứ m.

3. Vòng lặp boosting

Số cây: $n_estimators = 500$. Mỗi cây thứ t được thêm vào mô hình để dự đoán sai số của giá trị mà mô hình tính toán trước đó:

$$F_t(x) = F_{t-1}(x) + \eta * h_m(x),$$

với $h_m(x)$ là hàm tối ưu độ loss của giá trị ở lần thứ t, $\eta = 0.1$ là learning rate.

4. Kết thúc

Sau khi đã thực hiện tất cả lần lặp thì chúng ta sẽ được một mô hình dùng để dự đoán giá trị với input mà chúng ta mong muốn là

$$F_M(x) = F_0(x) + \eta \sum_{t=1}^M h_t(x)$$

4.2 Áp dụng XGBoost

1. Khởi tạo mô hình

Mô hình bắt đầu với dự đoán:

$$\hat{y}^{(0)} = avg(y)$$

2. Vòng lặp additive boosting

Số cây: $n_estimators = 500$. Mỗi cây thứ t được thêm vào mô hình:

$$\hat{y}^{(t)} = \hat{y}^{(t-1)} + \eta f_t(x),$$

với $\eta = 0.1$ là learning rate.



3. Gradient và Hessian

Với objective='reg:squarederror':

$$g_i = \hat{y}_i^{(t-1)} - y_i, \quad h_i = 1$$

4. Tối ưu trọng số lá (Leaf Weight)

Cho lá j :

$$G_j = \sum_{i \in I_j} g_i, \quad H_j = \sum_{i \in I_j} h_i$$
$$w_j^* = -\frac{G_j}{H_j + \lambda}, \quad \lambda = 1$$

Hàm mục tiêu sau tối ưu:

$$\text{obj}^* = -\frac{1}{2} \sum_{j=1}^T \frac{G_j^2}{H_j + \lambda} + \gamma T$$

5. Gain của split lá

Khi tách lá thành 2 lá trái (L) và phải (R):

$$Gain = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma$$

6. Regularization

Dộ phức tạp của cây:

$$\Omega(f_t) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2$$

- $reg_alpha = 0 \rightarrow$ L1 không sử dụng - $reg_lambda = 1 \rightarrow$ L2 áp dụng trong leaf weight và gain

7. Tổng kết

Mô hình đã sử dụng:

- Khởi tạo dự đoán $\hat{y}^{(0)}$
- Additive update: $\hat{y}^{(t)} = \hat{y}^{(t-1)} + \eta f_t(x)$
- Gradient và Hessian
- Leaf weight tối ưu w_j^*
- Gain để chọn split lá
- Regularization $L_2 (\lambda)$

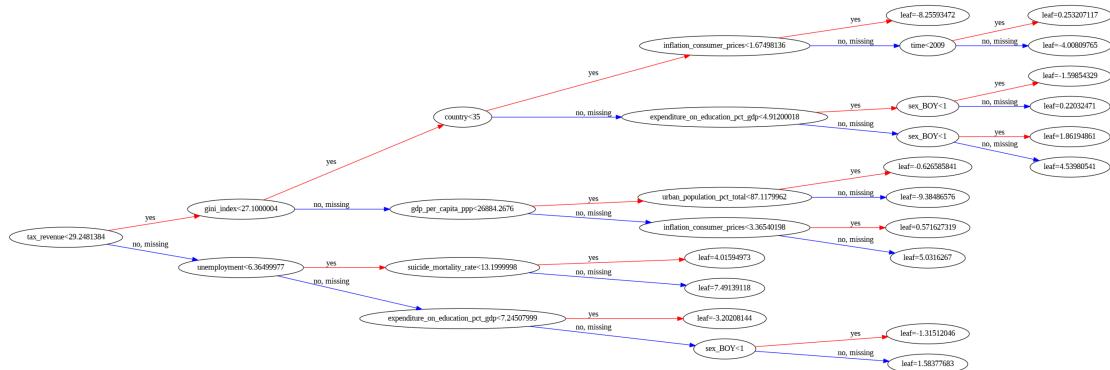
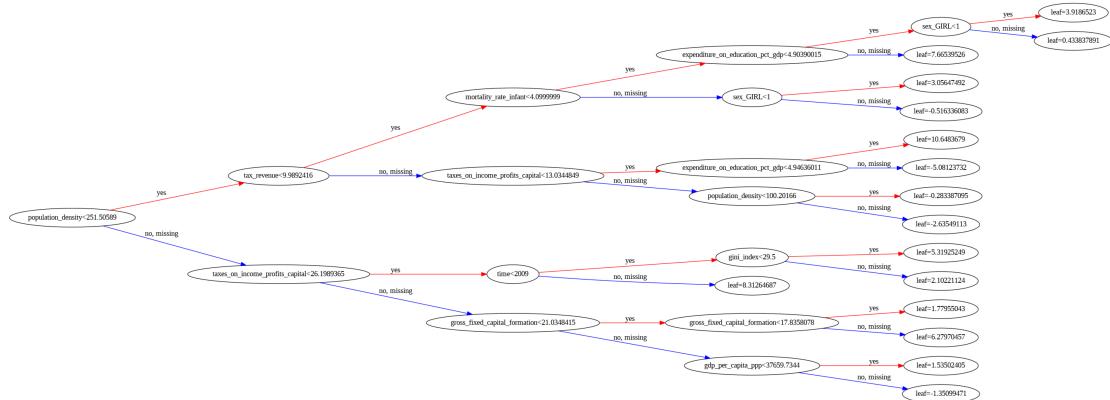
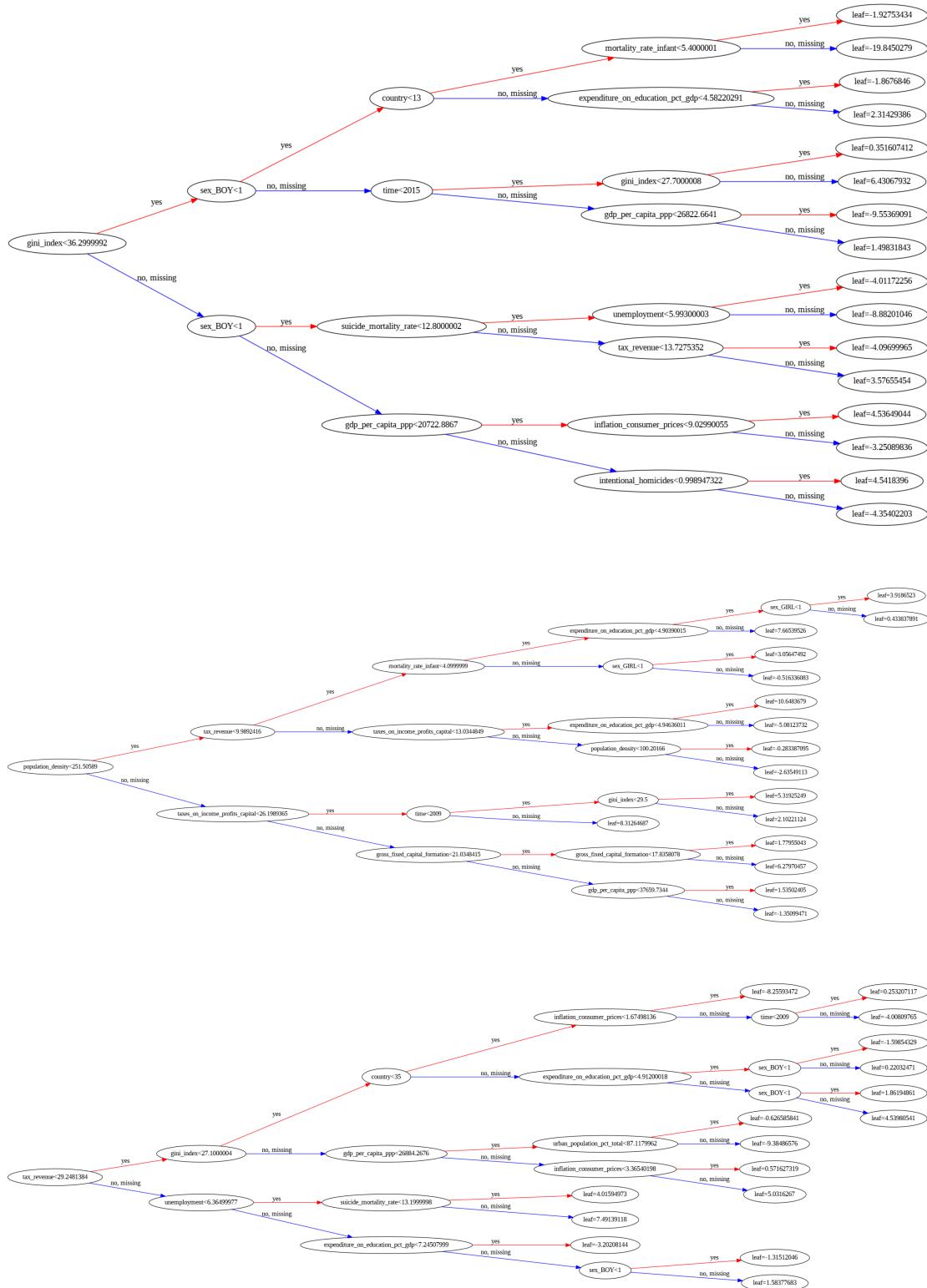
5 Tính toán VD mẫu số của 2 model

Vì mô hình mẫu có số lượng decision tree rất lớn là 500 cây nên chúng ta sẽ chỉ hiển thị của 10 cây decision tree đầu tiên

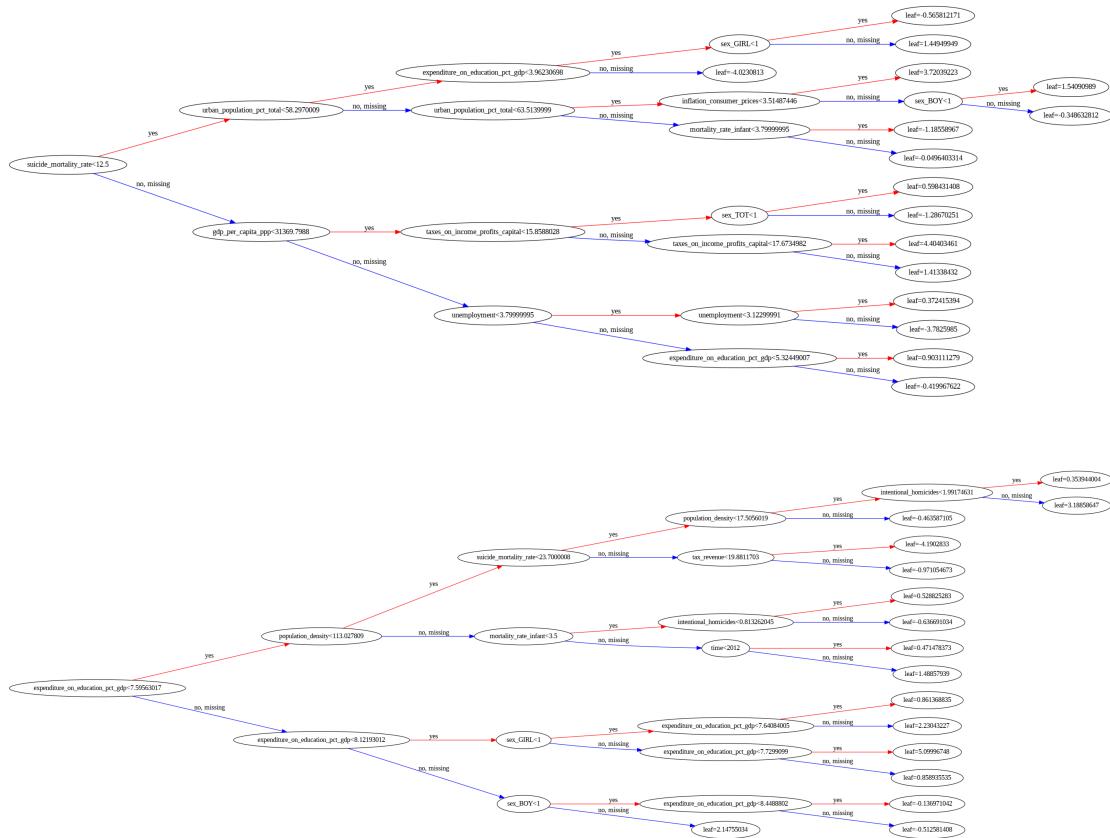
5.1 Tính toán VD của model XGBoost

Dưới đây là hình ảnh các decision tree của model XGBoost. Để tính toán ra input của một tập dữ liệu nhất định chúng ta chỉ cần đi qua tất cả các decision tree của model và có base score là 489.23932









5.1.1 Ví dụ mẫu 1

```

example = {
    'expenditure_on_education_pct_gdp': 6.0,
    'mortality_rate_infant': 12.8,
    'gini_index': 53.4,
    'gdp_per_capita_ppp': 15200.0,
    'inflation_consumer_prices': 3.73,
    'intentional_homicides': 22.7,
    'unemployment': 11.9,
    'gross_fixed_capitalFormation': 15.4,
    'population_density': 25.3,
    'suicide_mortality_rate': 7.2,
    'tax_revenue': 14.1,
    'taxes_on_income_profits_capital': 24.8,
    'government_health_expenditure_pct_gdp': 3.9,
    'urban_population_pct_total': 86.8,
    'time': 2019.0,
    'sex_BOY': 0.0,
    'sex_GIRL': 0.0,
    'sex_TOT': 1.0,
    'country': 3
}

```



}

Chúng ta sẽ đi qua các input trên, so sánh với các điều kiện phù hợp để tìm ra giá trị ở cuối. Từ đó chúng ta sẽ có các giá trị của các lá như sau:

```
leaf_value ={
    -10.450105
    -9.443799
    -8.534397
    -7.712566
    -6.780847
    -6.316905
    -5.708611
    5.408162
    -4.336209
    -4.480125
    -4.444702
    -4.034421
    -3.219854
    -2.498423
    -2.298098
    -2.068355
    -2.615762
    -1.638755
    .....
    -0.000419
    0.001319
    -0.001374
    0.000145
}
```

Sau đó ta sẽ tính ra giá trị dự đoán dự trên các giá trị lá theo công thức:

$$489.23947 + 0.1 * (\text{sum_of_leaf_value}) = 383.828437$$

5.1.2 Ví dụ mẫu 2

```
example= {
    'expenditure_on_education_pct_gdp': 4.9,
    'mortality_rate_infant': 3.1,
    'gini_index': 31.7,
    'gdp_per_capita_ppp': 56000.0,
    'inflation_consumer_prices': 1.5,
    'intentional_homicides': 0.9,
    'unemployment': 3.2,
    'gross_fixed_capitalFormation': 21.5,
    'population_density': 235.0,
    'suicide_mortality_rate': 12.3,
    'tax_revenue': 11.5,
```



```
'taxes_on_income_profits_capital': 12.0,  
'government_health_expenditure_pct_gdp': 11.0,  
'urban_population_pct_total': 77.4,  
'time': 2018.0,  
'sex_BOY': 0.0,  
'sex_GIRL': 0.0,  
'sex_TOT': 1.0,  
'country': 4  
}
```

Tương tự chúng ta sẽ có mẫu lá như sau:

```
leaf_value ={  
    2.452942  
    2.209340  
    1.175400  
    1.165170  
    1.738842  
    1.469539  
    1.323565  
    0.854045  
    1.099438  
    0.697128  
    0.282217  
    0.114572  
    1.070895  
    0.355455  
    0.429036  
    0.023944  
    0.033389  
    0.148541  
    .....  
    -0.000143  
    0.001456  
    0.000063  
    -0.001964  
}
```

Kết quả cuối là:

$$489.23947 + 0.1 * (\text{sum_of_leaf_value}) = 504.646960$$

5.1.3 Ví dụ mẫu 3

```
example= {  
    'expenditure_on_education_pct_gdp': 4.1,  
    'mortality_rate_infant': 16.5,  
    'gini_index': 35.7,  
    'gdp_per_capita_ppp': 8000.0,  
    'inflation_consumer_prices': 4.5,
```

```
'intentional_homicides': 1.5,  
'unemployment': 2.1,  
'gross_fixed_capitalFormation': 26.0,  
'population_density': 310.0,  
'suicide_mortality_rate': 5.5,  
'tax_revenue': 19.0,  
'taxes_on_income_profits_capital': 18.0,  
'government_health_expenditure_pct_gdp': 2.5,  
'urban_population_pct_total': 36.0,  
'time': 2018.0,  
'sex_BOY': 0.0,  
'sex_GIRL': 0.0,  
'sex_TOT': 1.0,  
'country': 5  
}
```

Chúng ta cũng sẽ có giá trị từ các decision tree như sau:

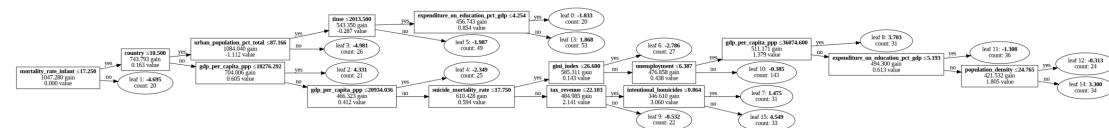
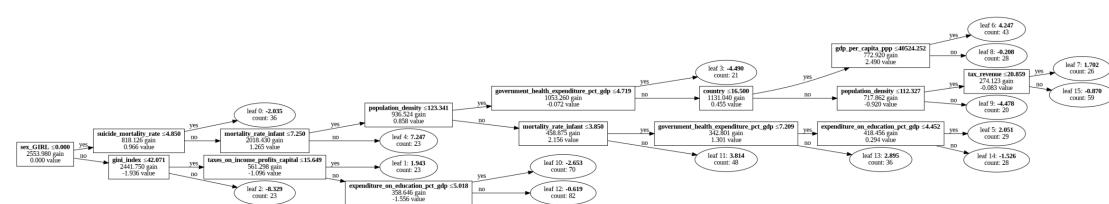
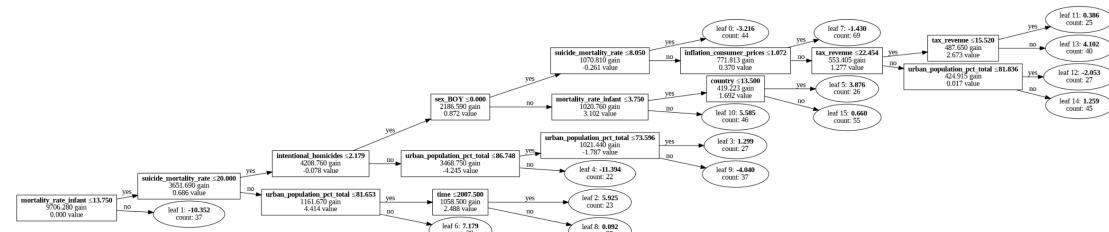
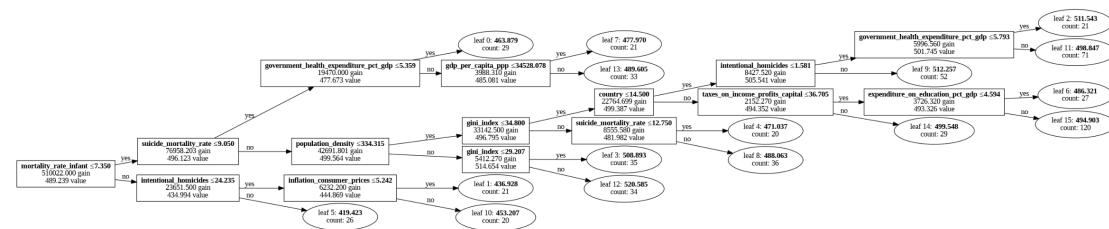
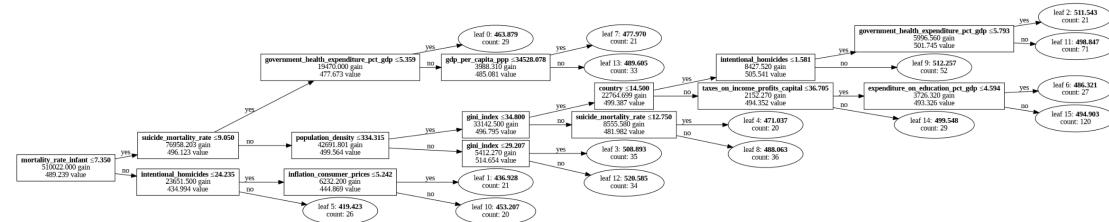
```
leaf_value ={  
    -5.158513  
    -2.606465  
    -4.464907  
    -2.148752  
    -1.674165  
    -4.573482  
    -4.031820  
    -3.834442  
    -2.757962  
    -3.033796  
    0.736859  
    -0.603394  
    -2.174437  
    -1.599532  
    -1.682142  
    -1.051218  
    -1.457840  
    -0.195544  
    .....  
    -0.000419  
    0.001319  
    -0.001374  
    -0.008194  
}
```

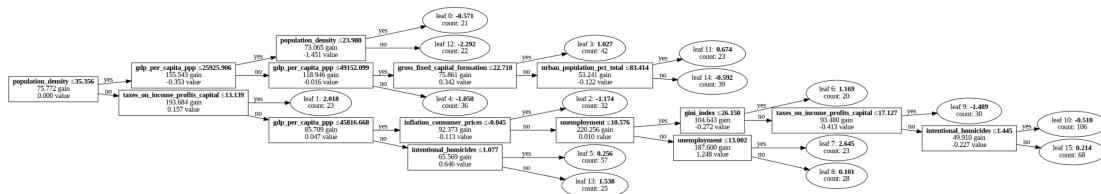
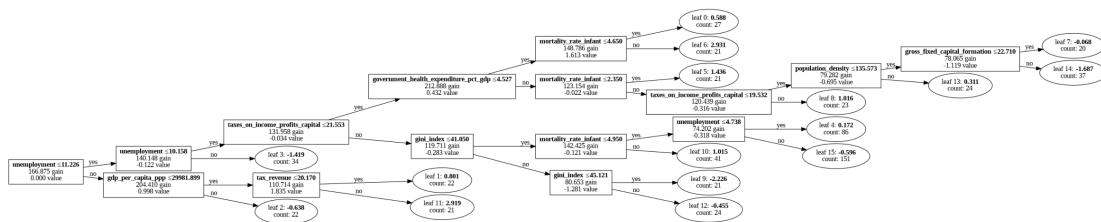
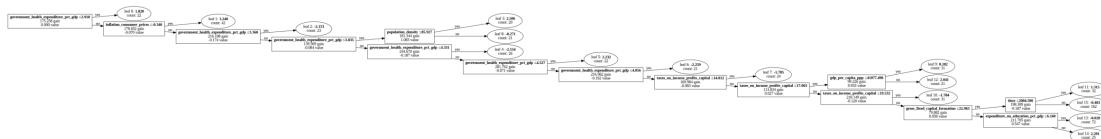
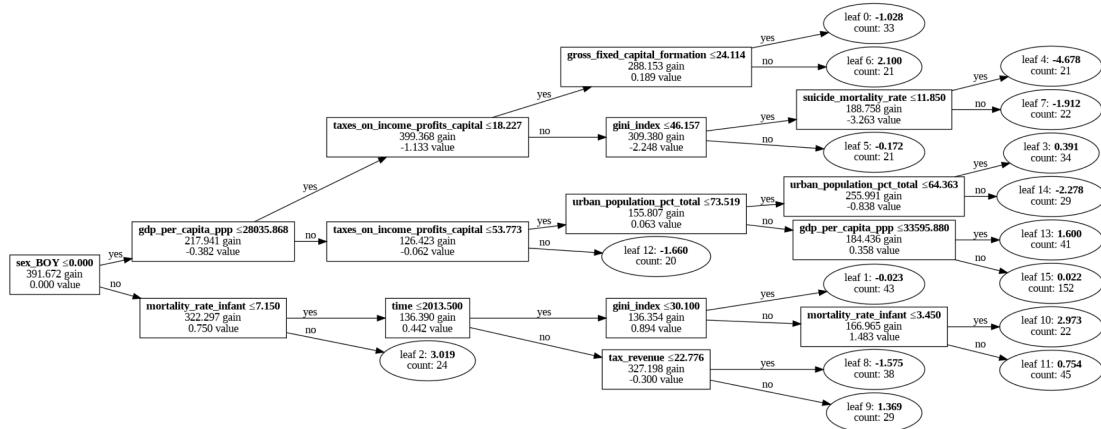
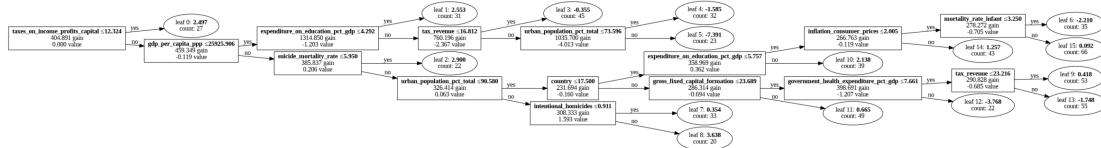
Và kết quả cuối cùng là:

$$489.23947 + 0.1 * (\text{sum_of_leaf_value}) = 436.261735$$

5.2 Tính toán VD của model LightGBM

Dưới đây là hình ảnh các decision tree của model LightGBM. Để tính toán ra input của một tập dữ liệu nhất định chúng ta chỉ cần đi qua tất cả các decision tree ở trên với learning rate là 0.1





Chúng ta cũng sẽ lấy input của 3 ví dụ trên đối với XGBoost để có thể thấy được sự khác nhau về giá trị.



5.2.1 Ví dụ mẫu 1

Chúng ta sẽ có giá trị của các lá là:

```
leaf_value ={
    478.789219
    9.443798
    -7.549456
    -7.807412
    -7.055587
    -6.183080
    -5.780752
    -5.449452
    -4.493056
    -4.433259
    -4.493234
    -4.078474
    -3.071509
    -2.830389
    -1.873924
    -1.489837
    -2.072076
    -1.448468
    .....
    0.002650
    -0.005829
    0.016500
    -0.008129
}
```

Chúng ta sẽ cộng lại tất cả giá trị trên để có thể ra được giá trị dự đoán là:

$$(\text{sum_of_leaf_value}) = 383.789692$$

5.2.2 Ví dụ mẫu 2

Giá trị của các lá là:

```
leaf_value ={
    490.674738
    1.293866
    1.243967
    0.558161
    0.889780
    0.831720
    0.608815
    0.841812
    1.233010
    0.464561
    0.643083
}
```

```
0.296441
0.140064
0.899228
0.304250
0.555461
0.598825
0.010099
.....
0.000062
0.007679
-0.000568
0.000553
}
```

Chúng ta sẽ cộng lại tất cả giá trị trên để có thể ra được giá trị dự đoán là:

$$(\text{sum_of_leaf_value}) = 506.780926$$

5.2.3 Ví dụ mẫu 3

Tương tự, ta có giá trị các lá là

```
leaf_value ={
    484.573925
    -4.223414
    -3.823301
    -3.461094
    -2.824744
    -2.800892
    -2.271969
    -1.420690
    -1.333661
    -1.619234
    -0.932799
    -1.242906
    -0.762764
    -0.692355
    -1.873924
    -2.436483
    -2.072076
    -0.833804
    .....
    0.003360
    -0.001241
    -0.003230
    0.004659
}
```

Chúng ta sẽ cộng lại tất cả giá trị trên để có thể ra được giá trị dự đoán là:

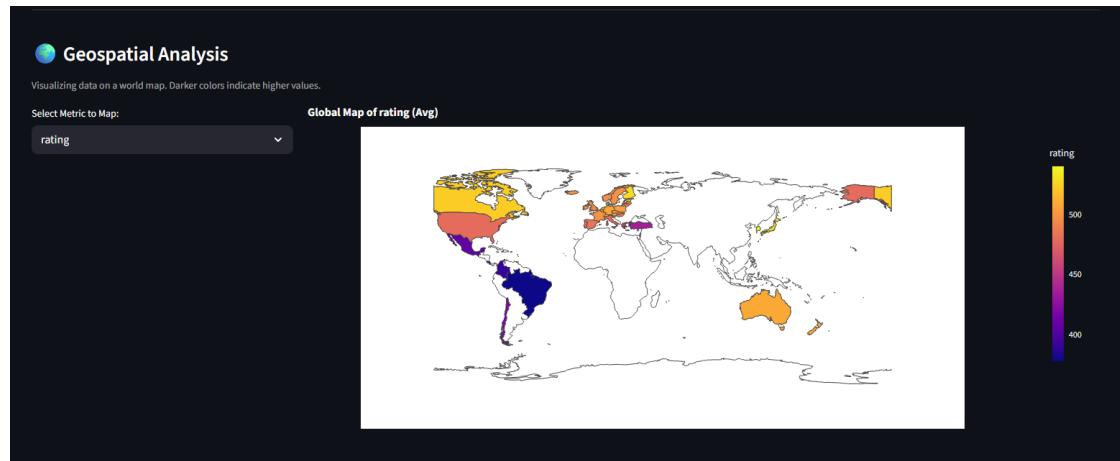
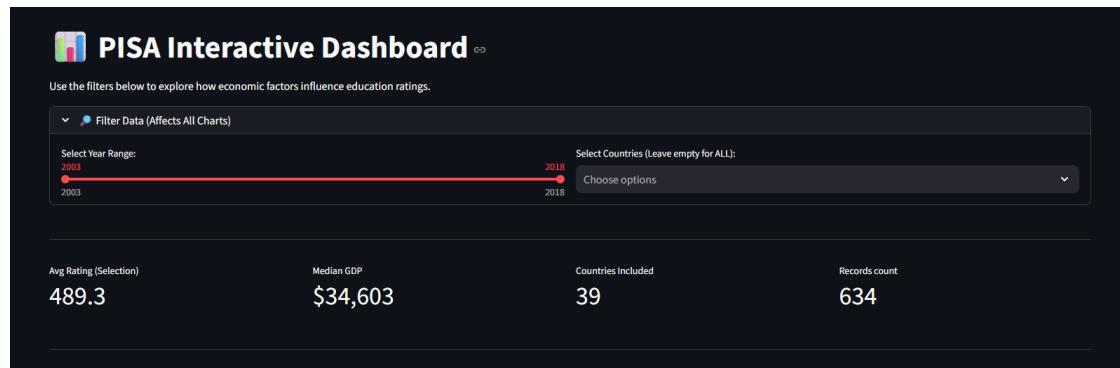


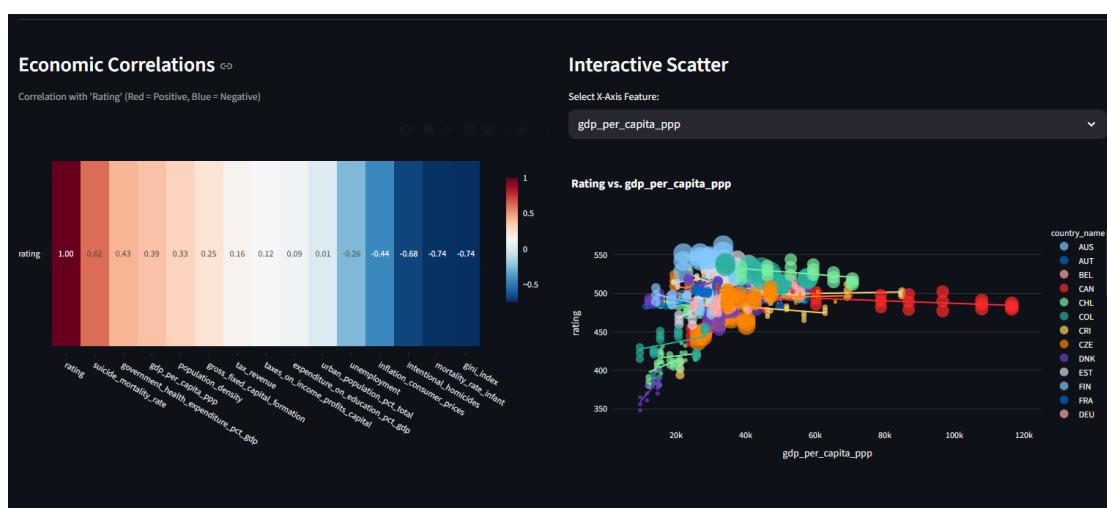
(sum_of_leaf_value) = 445.185580

6 Dashboard

6.1 Data Visualization

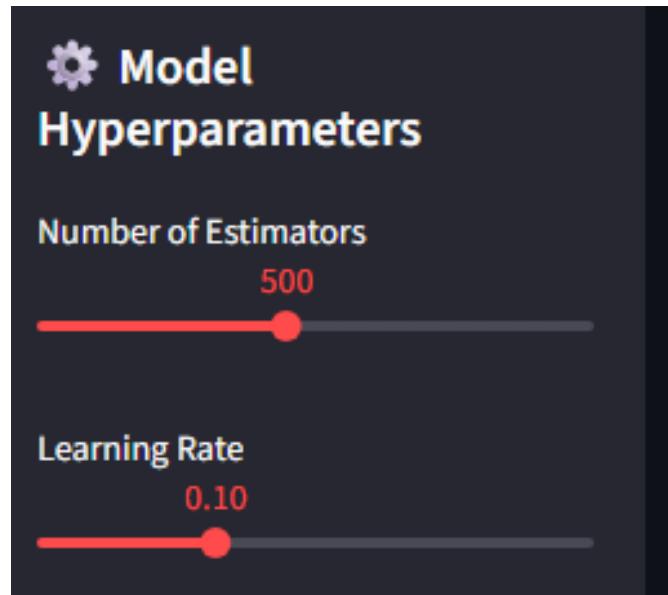
Đây là một số hình ảnh về phần dashboard của data visualization:





6.2 Model Training

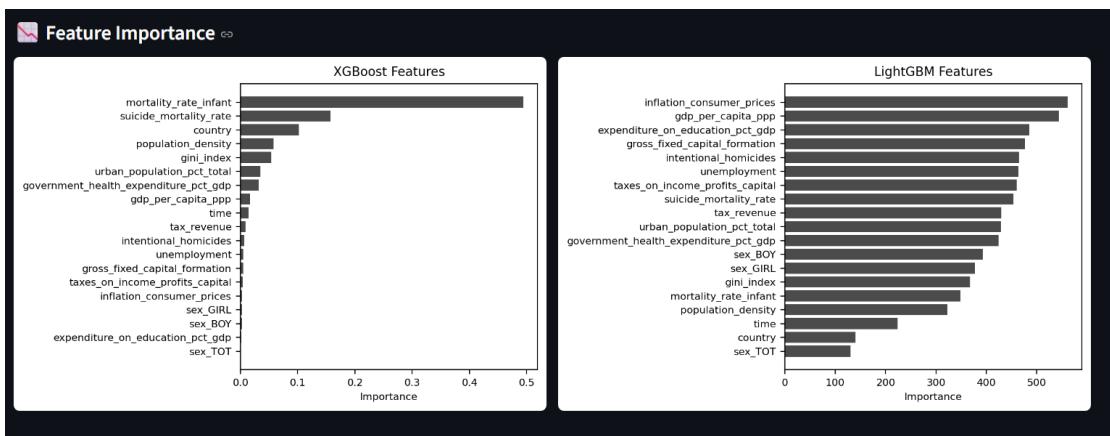
Chúng ta có thêm nút để có thể train model với thông số của các model có thể thay đổi như sau:



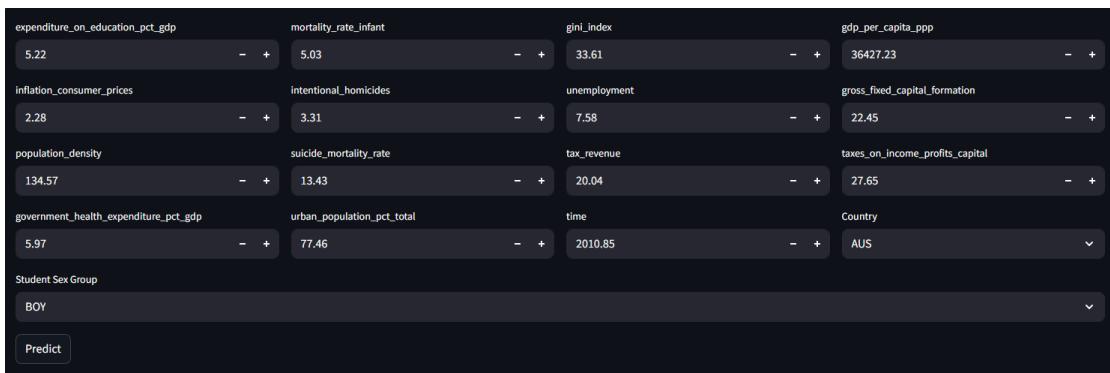
Cũng có bảng so sánh chênh lệch trong độ chính xác của 2 model XGBoost và LightGBM

Performance Metrics		
MAE	XGBoost	2.0769
	Mean Absolute Error (Lower is better)	2.0042
MSE	XGBoost	7.1667
	Mean Squared Error (Lower is better)	7.0844
RMSE	XGBoost	2.6771
	Root Mean Squared Error (Lower is better)	2.6617
R2 Score	XGBoost	0.9931
	R-Squared (Higher is better)	0.9932

Chúng ta cũng có biểu đồ để xem độ quan trọng của các feature đối với 2 model khác nhau:



Chúng ta cũng có thể nhập số liệu để model có thể dự đoán và xem dữ liệu sử dụng để test 2 model



7 Phân code

- Nguồn code: [Github](#)
- Dường dẫn app: [Streamlit project](#)
- Các folder đã được phân bố cụ thể, bao gồm các dataset (original và imputed), phân code dự đoán (ipynb và py) và phân visualization và training demo app (streamlit project)
- Cách chạy app trên local được đề cập trên file README.md



8 Phần kết

8.1 Kết luận

Tổng kết lại, đồ án đã hoàn thành trọn vẹn mục tiêu ban đầu về việc xây dựng một hệ thống dự đoán và phân tích chỉ số PISA dựa trên các dữ liệu kinh tế vĩ mô và xã hội. Thông qua quy trình làm việc bài bản từ thu thập, xử lý, làm sạch dữ liệu đến việc phân tích khám phá (EDA), nhóm đã làm rõ được mối tương quan mật thiết giữa các yếu tố như GDP, tỷ lệ thất nghiệp, chi tiêu chính phủ với chất lượng giáo dục. Việc áp dụng thành công hai thuật toán Gradient Boosting hiện đại là XGBoost và LightGBM đã chứng minh hiệu quả vượt trội trong việc xử lý dữ liệu bảng, mang lại độ chính xác cao và khả năng tổng quát hóa tốt cho bài toán hồi quy phức tạp này. Bên cạnh đó, sản phẩm cuối cùng là ứng dụng Dashboard trên nền tảng Streamlit không chỉ minh chứng cho tính thực tiễn của mô hình mà còn cung cấp một công cụ trực quan hỗ trợ người dùng trong việc tra cứu và đánh giá số liệu.

Tuy nhiên, nghiên cứu vẫn tồn tại một số hạn chế nhất định cần được khắc phục trong tương lai. Do giới hạn của bộ dữ liệu gốc, phạm vi nghiên cứu mới chỉ dừng lại ở 39 quốc gia và số liệu cập nhật đến năm 2018, chưa phản ánh hết những biến động kinh tế - xã hội trong giai đoạn gần đây. Ngoài ra, các yếu tố định tính như văn hóa giáo dục, chính sách vĩ mô đặc thù hay chỉ số hạnh phúc vẫn chưa được đưa vào mô hình, dẫn đến việc giải thích kết quả đôi khi chưa thực sự toàn diện. Trong các hướng phát triển tiếp theo, nhóm dự định sẽ mở rộng phạm vi thu thập dữ liệu, đồng thời áp dụng các kỹ thuật tối ưu hóa tham số nâng cao hơn để tiếp tục cải thiện hiệu suất dự đoán.

8.2 Lời cảm ơn

Để hoàn thành đồ án này, nhóm chúng em xin gửi lời cảm ơn chân thành và sâu sắc nhất đến thầy **Vũ Ngọc Tú**. Thầy đã tận tình hướng dẫn, định hướng đề tài và cung cấp những kiến thức nền tảng quý báu cũng như những nhận xét xác đáng giúp chúng em hoàn thiện bài báo cáo này một cách tốt nhất.

Chúng em cũng xin gửi lời cảm ơn đến quý Thầy, Cô thuộc Khoa Khoa học và Kỹ thuật Máy tính, trường Đại học Bách Khoa - DHQG TP.HCM đã truyền đạt kiến thức chuyên môn và tạo mọi điều kiện thuận lợi trong suốt quá trình học tập và nghiên cứu tại trường. Cuối cùng, xin cảm ơn cộng đồng mã nguồn mở và tác giả của bộ dữ liệu trên Kaggle đã cung cấp tài nguyên quan trọng để chúng em có cơ sở thực hiện đề tài này.

Mặc dù đã nỗ lực hết mình, nhưng do giới hạn về mặt kiến thức và thời gian, đồ án khó tránh khỏi những thiếu sót. Chúng em rất mong nhận được sự đóng góp ý kiến của Thầy để đồ án được hoàn thiện hơn. Chúng em xin chân thành cảm ơn!