INT3405E Ứng dụng học bán giám sát và tự giám sát cho dữ liệu dạng bảng trong bài toán CMI - PIU

*Nhóm 23 - noob: Rank 94/3559 - 0.456 Private score

Nguyễn Hữu Thế
22028155@vnu.edu.vn

Ngô Duy Hiếu
22028280@vnu.edu.vn

Nguyễn Hữu Tiến
22028180@vnu.edu.vn

***Abstract*—Học tự giám sát (Self-supervised learning) và học bán giám sát (Semi-supervised learning) đã đạt được nhiều bước tiến đáng kể trong những lĩnh vực liên quan đến dữ liệu ảnh hay dữ liệu văn bản, cho phép xây dựng các mô hình hoạt động một cách hiệu quả ngay cả khi có hạn chế về lượng dữ liệu đầy đủ nhãn trong khi lượng dữ liệu chưa có nhãn lại rất nhiều, dựa trên cấu trúc đặc biệt của dữ liệu ảnh (mối quan hệ không gian), dữ liệu văn bản (mối quan hệ ngữ nghĩa). Tuy nhiên đối với dữ liệu dạng bảng (tabular data), không có cấu trúc rõ ràng như dữ liệu dạng ảnh và và văn bản, khiến cho việc ứng dụng mô hình học tự giám sát (Self-supvervised learning) và học bán giám sát (Semi-supervised) gặp rất nhiều khó khăn và trở nên không hiệu quả. Ở trong bài báo cáo này, nhóm sẽ tập trung vào phân tích mô hình VIME (Value Imputation and Mask Estimation) [1], một cách đề xuất tiếp cận mới của học tự giám sát và bán giám sát đối với dữ liệu dạng bảng, và cũng như cách nhóm ứng dụng nó vào bài toán thực tế (CMI - PIU) để đưa ra được cái nhìn trực quan về tính hiệu quả của phương pháp đề xuất.**

***Index Terms*—CMI-PIU, VIME, Self-supervised learning, Semi-supervised learning, Supervised learning, Ordinal Classification**

I. Giới thiệu bài toán

CMI - PIU là kỳ thi được tổ chức trên nền tảng Kaggle, với mục tiêu là xây dựng mô hình có khả năng ước lượng mức độ sử dụng internet ở đối tượng trẻ vị thành niên dựa trên các chỉ số thể chất đo lường được. Từ đó, đối với phía cha mẹ, có thể sớm phát hiện tình trạng sử dụng internet của trẻ và đưa ra những biện pháp khuyến khích tham gia các hoạt động ngoài trời hoặc sử dụng intermet một cách hiệu quả và có kiểm soát. Bộ dữ liệu HBN (Healthy Brain Network) được cung cấp là một mẫu lâm sàng gồm khoảng 5000 người trong độ tuổi 5 - 22 đã trải qua sàng lọc lâm sàng và sàng lọc nghiên cứu. Hai yếu tố của nghiên cứu được sử dụng cho cuộc thi là dữ liệu hoạt động thể chất (dữ liệu máy đo gia tốc đeo cổ tay, đánh giá thể lực và bảng câu hỏi) và dữ liệu về hành vi sử dụng Internet. Từ đó, đánh giá được chỉ số *sii* (Severity Impairment Index), một chỉ số tiêu chuẩn về mức độ lạm dụng Internet.

Trên thực tế, việc thu thập dữ liệu chưa bao giờ là dễ dàng. Để có một bộ dữ liệu hoàn chỉnh gần như là bất khả thi, khi mà phải đối mặt với rất nhiều những thách thức. Và vấn đề này cũng phát sinh tương tự đối với bộ dữ liệu được cung cấp từ cuộc thi. Bên cạnh những dữ liệu trống ở cột predictors, thì ở cột mục tiêu (target) trống đến $\frac{1}{3}$ trên tổng số mẫu được cung cấp cho việc huấn luyện. Điều này đặt ra một thử thách rất lớn cho việc huấn luyện mô hình để đạt được kết quả mong muốn. VÀ đó là lý do nhóm đề xuất đến việc sử dụng đến mô hình VIME cho bài toán.

Mô hình VIME [1], hay viết tắt của Value Imputation and Mask Estimation, được đề xuất năm 2020 từ 4 nhà khoa học Jinsung Yoon, James Jordon, Yao Zhang, và Mihaela van der Schaar, đã tạo ra một sự đột phá trong việc ứng dụng học bán giám sát (Semi-supervised learning) và học tự giám sát (Self-supervised learning) trong dữ liệu dạng bảng. Hai phương pháp học máy trên đã đạt được những bước tiến đáng kể trong dữ liệu ảnh hay văn bản nhờ vào cấu trúc không gian và cấu trúc ngữ nghĩa đặc biệt của những loại dữ liệu này, khắc phục được vấn đề hạn chế về số lượng dữ liệu có nhãn trong khi lại có đủ nhiều dữ liệu không được gắn nhãn. Một mô hình học tự giám sát (Self-supervised learning) tiêu chuẩn, định nghĩa ra các ***pretext*** để có thể học được cách biểu diễn thông tin của dữ liệu. Đối với dữ liệu văn bản, BERT đề xuất 4 loại pretext tasks (như là dự đoán từ mới từ những từ có sẵn) để có thể học được biểu diễn của dữ liệu văn bản [2]. Hay đối với dữ liệu dạng ảnh, những pretext tasks như xoay [3], ghép hình [4], tô màu [5] được sử dụng để học cách biểu diễn của dữ liệu. Các phương pháp học bán giám sát (Semi-supervised learning) tiêu chuẩn cũng tương tự, khi mà chúng cũng dựa trên kiến thức về cấu trúc của những loại dữ liệu này. Tuy nhiên, khi xử lý dữ liệu dạng bảng (Tabular data), thì những phép xoay đơn giản hay như dự đoán predictor sau từ những predictors trước hoàn toàn không hợp lý, bởi vì không tồn tại mối quan hệ về không gian hay mối quan hệ ngữ nghĩa giữa các predictors với nhau.

Đối với học tự giám sát (Self-supervised learning), đề xuất ra *pretext tasks* chuyên biệt với dữ liệu dạng bảng, lần lượt *mask vector estimation* và *feature vector estimation*. Để có thể giải quyết được những *pretext tasks* này, cần phải tạo một encoder để có thể học được cách biểu diễn thông tin của các predictors từ dữ liệu khuyết nhãn. Với học bán giám sát (Semi-supervised learning), sử dụng encoder được huấn luyện từ giai đoạn học tự giám sát trước đó, với mục đích tạo ra nhiều mẫu dữ liệu tăng cường (*augmented samples*) từ một mẫu bằng cách che đi một hoặc nhiều predictors và gán một giá trị khác cho vị trí bị che đi. Cuối cùng, mô hình VIME (Value Imputation and Mask Estimation) được tạo ra từ kết hợp 2 ý tưởng học tự giám sát và bán giám sát ở trên, cho phép bài toán dự đoán

có thể giải quyết một cách hiệu quả trên dữ liệu dạng bảng, ngay cả khi chỉ có một số ít dữ liệu được gắn nhãn.

Trong bài toán CMI - PIU, nhóm đề xuất việc tăng cường hiệu quả của mô hình học có giám sát (Supervised Learning) từ việc sử dụng VIME để gán nhãn những mẫu bị khuyết nhãn. Thông qua việc tạo ra thêm nhiều dữ liệu chất lượng với đầy đủ nhãn, nhóm hi vọng có thể tạo ra kết quả mang tính đột phá ở cuộc thi.

II. Xử lý dữ liệu

A. Tổng quan về dữ liệu

Với những cá nhân tham gia vào sàng lọc lâm sàng và sàng lọc nghiên cứu, ngoài những chỉ số thể lực và sức khỏe liên quan được đánh giá trực tiếp tại thời điểm tiến hành đo lường, mà người tham gia còn sử dụng một thiết bị đeo tay có tác dụng đo lường gia tốc trong suốt thời gian đeo. Chính vì vậy dữ liệu cuộc thi gồm 2 nguồn, những tệp .parquet chứa giá trị đo lường được của những người đeo trong khoảng thời gian thực hiện thí nghiệm, và một tệp .csv là dữ liệu bảng chứa các chỉ số thể chất và sức khỏe của những người tham gia tại thời điểm tiến hành đo lường.

Dữ liệu bảng Bao gồm các thuộc tính trong các chỉ mục liên quan, bao gồm nhân khẩu học (tuổi và giới tính), thời gian sử dụng Internet theo ngày, thang đánh giá sức khỏe tâm thần, đánh giá sức khỏe thể chất (huyết áp, nhịp tim, chiều cao, cân nặng, vòng eo và hông), đánh giá sức khỏe tim mạch, đánh giá thể lực (khả năng hiếu khí, sức mạnh cơ bắp, sức bền cơ bắp, tính linh hoạt), đo lường các yếu tố chính cấu thành cơ thể (BMI, hàm lượng mỡ, khối lượng cơ và tỷ trọng nước), hoạt động tham gia trong 7 ngày gần nhất, thang đo phân loại rối loạn giấc ngủ, đo lường về mức độ hoạt động vật lý sinh thái, và bộ 20 câu hỏi là thang đo từ 1 - 5 về đặc điểm hành vi liên quan đến việc sử dụng Internet. Trong đó chỉ số sii có thể được đánh giá trực tiếp từ việc tổng kết từ 20 câu hỏi liên quan đến đặc điểm, hành vi sử dụng Internet. Tuy nhiên, chỉ có khoảng $\frac{2}{3}$ người tham gia hoàn thành đầy đủ bộ 20 câu hỏi, điều đó đồng nghĩa với việc có đến $\frac{1}{3}$ dữ liệu khuyết nhãn. Và ở bộ dữ liệu test của cuộc thi, bên cạnh việc không có cột sii chính là target của mô hình, thì không có dữ liệu về thang đo đặc điểm hành vi sử dụng Internet. Điều đó cũng chính là mục tiêu cuộc thi khi hướng đến việc xây dựng bằng chứng đáng tin cậy giữa thông tin về thể chất, hành vi, sức khỏe và mức độ lạm dụng Internet ở trẻ.

Dữ liệu về hoạt động theo thời gian được thu thập qua một thiết bị đeo tay - máy đo gia tốc liên tục được phát cho mọi người tham gia xét nghiệm và họ phải để đeo liên tục trong tối đa 30 ngày khi ở nhà và cuộc sống sinh hoạt hàng ngày. Thiết bị đeo tay đo lường các chỉ số bao gồm X, Y, Z là vị trí cảm nhận theo chiều mỗi trục, ENMO là một chỉ số phổ biến dùng để đo mức độ hoạt động với giá trị bằng 0 biểu hiện không có hoạt động nào, "non-wear flag" là chỉ số xác định liệu người tham gia thí nghiệm có đeo thiết bị hay không, và một số chỉ số liên quan đến thời gian đeo thiết bị như "time of day", "week day" và "quarter". Các chỉ số đo được từ thiết bị này giúp bổ sung vào tập dữ liệu những trường thông tin quan trọng liên quan đến thói quen sinh hoạt của người tham gia, lấy ví dụ như việc đánh giá về hoạt động thể chất mạnh, yếu hay không, trung bình bao nhiêu hoạt động một ngày, hay trung bình thời gian kéo dài mỗi lần hoạt động một ngày, trung bình về thời gian nghỉ hoạt động thể chất (nghỉ ngơi) một ngày,... Việc xử lý và áp dụng những góc nhìn khoa học trên tập dữ liệu về hoạt động theo thời gian, cho phép nhóm tạo ra những trường thông tin quan trọng, gắn liền với thói quen sinh hoạt của người tham gia, điều mà nhóm tin rằng có mối quan hệ rất lớn đối với việc đánh giá mức độ lạm dụng Internet ở trẻ.

B. Tiền xử lý dữ liệu

Xem thêm ở phần 1.1 Tabular data analysis, trong phần cài đặt của nhóm để có thêm góc nhìn về bộ dữ liệu. Nhìn vào hình 1, có thể quan sát được với tổng là 81 cột, chưa tính đến cột sii, thì chỉ có 4 trên tổng số 81 cột đầy đủ dữ liệu là các cột $\{"id", "Basic - Demos - Sex", "Basic - Demos - Age", "Basic - Demos - Enroll - Season"\}$. Còn lại predictors khác đều gặp phải tình trạng thiếu dữ liệu từ 10.6% cho đến lớn nhất là 88% ở 2 predictors là $\{"PAQ - A - PAQ - A - Total", "PAQ - A - Season"\}$, điều này gây ra nhiều thách thức cho việc xây dựng mô hình học máy có hiệu suất đủ tốt. Vậy nên, ngay từ đầu, nhóm chú trọng vào việc xử lý và làm sạch dữ liệu, đảm bảo nguồn dữ liệu chất lượng cho việc phát triển mô hình phía sau. Trước tiên nhóm lọc ra những predictors được xem xét là triển vọng và ổn định, nghĩa là chúng có phần trăm dữ liệu bị thiếu không quá nhiều, với những predictors có lên đến hơn 80% lượng dữ liệu thiếu, nhóm quyết định loại bỏ chúng, để tránh gây nhiễu và sai xót trong quá trình xây dựng mô hình. Tiếp đó, đối với phần bị thiếu còn lại của những predictors được chọn, nhóm xem xét việc sử dụng các kỹ thuật khác nhau với mục đích bổ sung vào những phần dữ liệu bị thiếu.

Đối với dữ liệu dạng số (numeric values), nhóm xem xét việc sử dụng kỹ thuật KNNImputer (Được cung cấp ở thư viện ScikitLearn). KNNImputer hoạt động dựa trên việc tìm kiếm K hàng xóm gần nhất đối với dữ liệu bị thiếu, sau đó sẽ bổ sung dựa vào tính toán giá trị trung bình từ K hàng xóm đã xác định trước đó. Trong đó, giá trị K, số lượng hàng xóm liền kề, sẽ được ước lượng thông qua kỹ thuật GridSearchCV. So sánh với một số kỹ thuật impute khác như SimpleImputer chỉ tính toán những giá trị trung bình hay trung vị, KNNImputer cho phép bảo toàn được mối tương quan giữa các features, đưa ra ít các giả định phân phối hơn và trả về một loạt các giá trị hợp lý thay vì chỉ một ước tính.

Đối với dữ liệu dạng thuộc tính (categorical values), nhóm xem xét những phần bị thiếu như một thuộc tính riêng, và quy chuẩn tất cả các thuộc tính sang dạng số nguyên, phục vụ cho việc xây dựng mô hình phía sau.

C. Feature engineering

1) Dữ liệu dạng bảng: Đối với dữ liệu dạng bảng, nhóm thực hiện tính toán một số chỉ số khoa học đặc trưng liên quan đến sức khỏe thể chất dựa trên các chỉ số đã đo lường được. Việc tạo thêm những chỉ số thể chất liên quan giúp nhóm có được một bộ dữ liệu toàn diện trong bài toán mục tiêu là xây

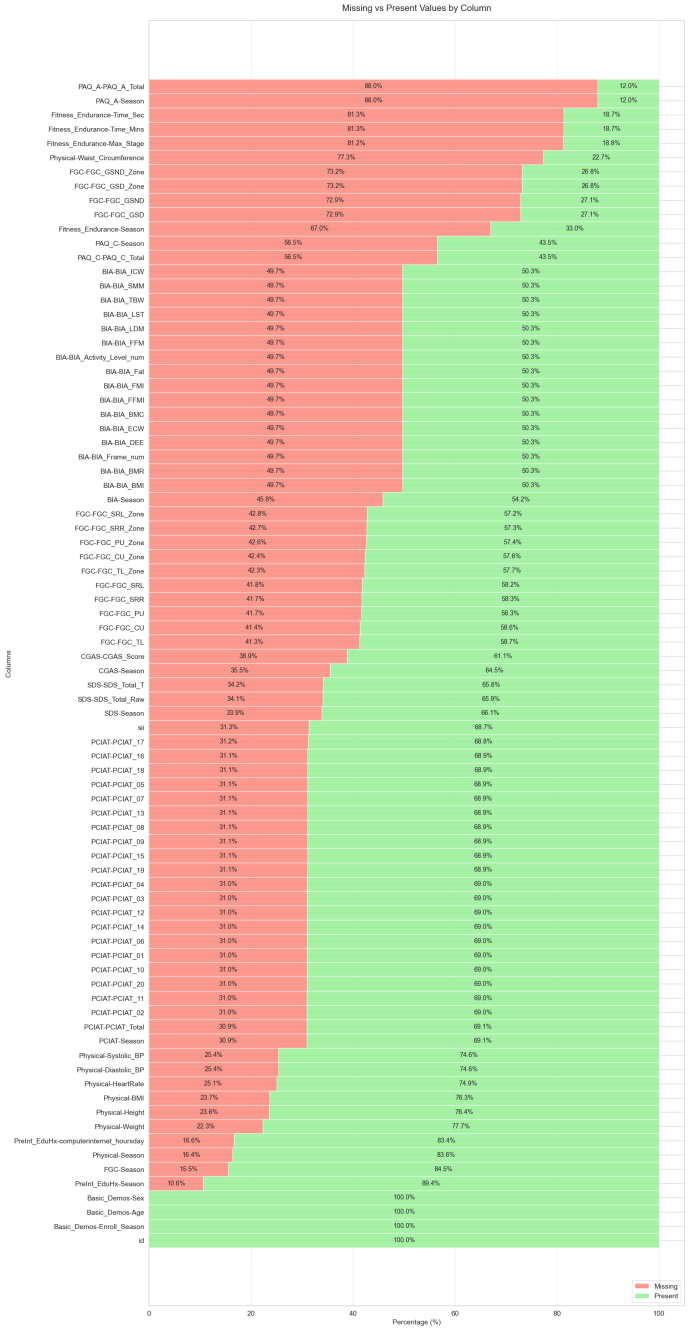

Figure 1. Tổng quan về phần trăm dữ liệu thiếu tại mỗi cột. Hình ảnh được tạo bởi matplotlib.

dựng mối quan hệ giữa hoạt động thể chất và tình trạng lạm dụng Internet ở trẻ.

Các chỉ số thể chất được nhóm phát triển thêm có thể kể đến như chỉ số BMI tương ứng với tuổi (BMI * Tuổi), phần trăm lượng nước cơ thể (Tổng lượng nước / Khối lượng), tỷ lệ phần trăn cơ và lượng mỡ,... (*Xem chi tiết ở phần Feature engineering helper function trong mục 1.3 Data loading & Processing ở phần cài đặt*).

2) Dữ liệu hoạt động theo thời gian: Đối với việc phân tích và xử lý dữ liệu đo lường theo thời gian, đầu tiên nhóm lọc ra những giá trị đo lường mà ở đó chỉ số "$non-worn-flag$" $= 0$, đồng nghĩa với việc người tham gia đang đeo thiết bị, và ngược lại nếu giá trị "$non-worn-flag$" $= 1$ thì đồng nghĩa việc người tham gia không đeo, những giá trị đo được ở thời điểm này đều là những giá trị ngẫu nhiên vô nghĩa.

Việc kết hợp góc nhìn khoa học về sức khỏe thể chất vào việc xử lý dữ liệu là quan trọng, điều này cung cấp cho bộ dữ liệu mà nhóm đang xây dựng có được những chỉ số gắn liền với thói quen sinh hoạt của người tham gia, bên cạnh những chỉ số đo lường thể chất. Điều này càng làm tăng thêm tính tin cậy của mô hình được xây dựng ở phần sau.

Chỉ số ENMO (Euclidean Norm Minus One) được sử dụng trong lĩnh vực học máy để phân tích dữ liệu chuyển động, tính toán từ dữ liệu gia tốc thu thập được qua thiết bị. ENMO được sử dụng để đánh giá các hoạt động thể chất, phân biệt giữa các hoạt động thể chất khác nhau, từ không vận động, ít vận động đến các hoạt động nhẹ và trung bình. Thông qua việc phân tích về sự thay đổi giá trị ENMO đo lường được từ thiết bị theo thời gian thực, có thể hiểu rõ hơn về mức độ và thói quen hoạt động của một người.

Ở đây, nhóm sử dụng ENMO để có thể tính được tổng thời gian và số lần không chuyển động trên ngày ($ENMO = 0$); trung bình, đỉnh điểm của giá trị ENMO theo từng giờ trong ngày; tổng thời gian và số lần của các hoạt động thể chất yếu, trung bình và mạnh theo từng ngày. Sau đó, nhóm tính toán các giá trị thống kế như Trung bình, trung vị, lớn nhất và độ lệch chuẩn và đưa vào kết hợp cùng với dữ liệu bảng trước đó. Ngoài ra, còn có tính toán chỉ số tương quan giữa chỉ số ENMO và cường độ sáng đo lường. (*Để chi tiết hơn, xem phần cài đặt của nhóm, phần Time-series Feature engineering helper function ở mục 1.3 - Data loading and processing*).

III. Phương pháp

Như đã đề cập ở các mục trước, bộ dữ liệu được cung cấp bởi cuộc thi đối mặt với vấn đề thiếu xót nghiêm trọng, đặt ra nhiều thách thức trong việc xây dựng được một mô hình đạt hiệu suất như mong muốn. Bên cạnh sự thiếu dữ liệu ở các cột predictors, thì có đến $\frac{1}{3}$ tổng số mẫu không được gắn nhãn (Xem hình 2). Đó là lý do nhóm đề xuất đến việc sử dụng VIME - Ứng dụng học bán giám sát (Semi-supervised learning) và tự giám sát (Self-supervised learning) cho dữ liệu bảng - để có thể gán nhãn phần dữ liệu khuyết nhãn, từ đó tạo ra thêm nhiều mẫu chất lượng cho bộ dữ liệu huấn luyện mô hình học giám sát (Supervised learning).

Hình 3 mô tả khái quát về phương pháp được nhóm ứng dụng xây dựng mô hình cho bài toán CMI - PIU. Bộ dữ liệu, bao gồm dữ liệu bảng và dữ liệu thời gian, sau khi được làm sạch, và thực hiện các kỹ thuật Feature Engineering (Chi tiết ở mục II - Xử lý dữ liệu), sẽ được sử dụng để huấn luyện mô hình học tự giám sát (Self-supervised learning) cho phép học thông tin biểu diễn của dữ liệu. Để có thể học được biểu diễn dữ liệu, một Encoder được huấn luyện để giải quyết các pretext tasks được định nghĩa bởi VIME dành cho dữ liệu dạng bảng. Sau đó, pha học bán giám sát (Semi-supervised learning) sẽ tận dụng Encoder, cùng với dữ liệu đã được gắn nhãn để gán nhãn trên dữ liệu khuyết nhãn. Và kết quả là một bộ dữ liệu

mới được tạo ra bởi dữ liệu có nhãn trước đó, và một bộ phận mẫu được gán nhãn mới chất lượng.

Tiếp đến, là pha huấn luyện mô hình học có giám sát từ bộ dữ liệu mới, ở đây nhóm đề xuất sử dụng 3 mô hình phân loại được sử dụng phổ biến trong các cuộc thi được tổ chức trên nền tảng Kaggle, bao gồm LightGBM [6], XGBoost [7], và CatBoost [8]. 3 mô hình trên đều được điều chỉnh tham số thông qua kỹ thuật RandomizedSearchCV (Được cài đặt ở thư viện Scikitlearn), cho phép khám phá không gian tham số rộng hơn và tiết kiệm thời gian khi so sánh với GridSearchCV. Cuối cùng, kết quả dự đoán trên tập test được rút ra từ cả 3 mô hình học giám sát thông qua một kỹ thuật Ensemble là Voting Regressor.

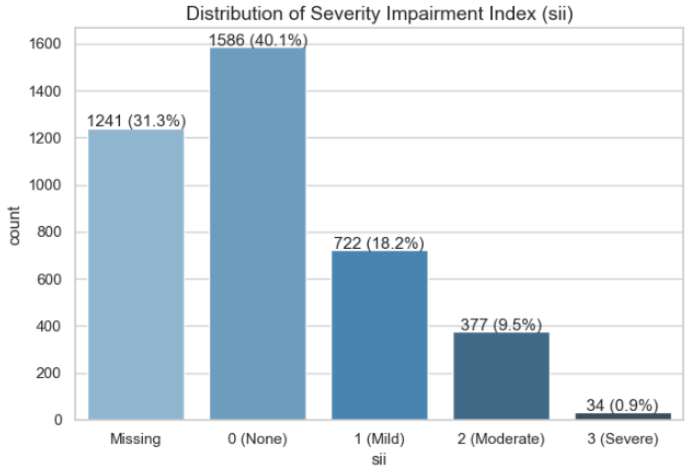

Figure 2. Số lượng mẫu có nhãn và không nhãn. Hình ảnh được tạo bởi matplotlib.

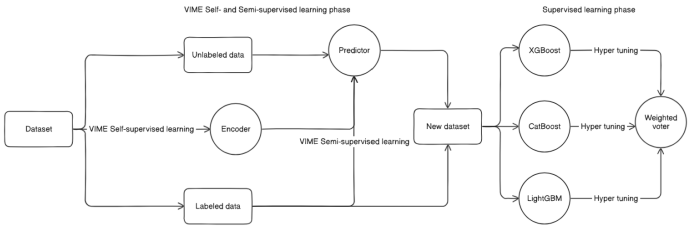

Figure 3. Mô tả về phương pháp được đề xuất. Hình ảnh được tạo bởi erase.io

A. *Ứng dụng VIME trong gán nhãn dữ liệu khuyết nhãn*

Mô hình VIME (Value Imputation and Mask Estimation), bao gồm 2 pha, lần lượt là pha học tự giám sát (Self-supervised learning) huấn luyện một Encoder học biểu diễn thông tin (informative representation) của dữ liệu ban đầu để giải quyết các pretext tasks được định nghĩa cho dữ liệu dạng bảng. Sau đó, pha học bán giám sát (Semi-supervised learning), sử dụng Encoder từ pha trước cùng với dữ liệu có nhãn để đưa ra dự đoán trên dữ liệu không được gán nhãn.

1) Pha học tự giám sát (Semi-supervised learning): Ở pha học tự giám sát (Semi-supervised learning), 2 pretext tasks được đề xuất cho dữ liệu dạng bảng lần lượt là *feature vector estimation* và *mask vector estimation*, từ đó đặt mục tiêu là huấn luyện một Encoder có thể khôi phục mẫu đầu vào (feature vector) từ những phép biến đổi trên nó và xác định phép biến đổi (mask vector), hay nói rộng là học thông tin biểu diễn của dữ liệu (Xem hình 4).

Hình 5 mô tả chi tiết các bước trong pha học tự giám sát (Semi-supervised learning). Đầu tiên, bộ sinh mask vector (mask vector generator), tạo ra một mask vector nhị phân $m = [m_1, ..., m_d]^T \in \{0,1\}^d, m_j \sim Bern(p_m)$. Sau đó, một bộ tạo pretext g_m lấy một mẫu $\mathbf{x}$ từ bộ dữ liệu và một mask vector $\mathbf{m}$ để tạo ra mẫu mới $\tilde{x}$ theo công thức:

$$\tilde{\mathbf{x}} = g_{\mathbf{m}}(\mathbf{x}, \mathbf{m}) = \mathbf{m} \odot \bar{\mathbf{x}} + (1 - \mathbf{m}) \odot \mathbf{x} \tag{1}$$

Chi tiết hơn về minh họa cho phép biến đổi, xem ở hình 6. Phép biến đổi đảm bảo $\tilde{\mathbf{x}}$ tương tự đối với những mẫu khác trong bộ dữ liệu huấn luyện. So sánh với một số phương pháp biến đổi mẫu khác, như thêm Gaussian noise, thay thế 0 bởi các dữ liệu trống, thì phép biến đổi trên cho phép tạo ra $\tilde{\mathbf{x}}$ khó phân biệt từ x.

Encoder (**e**) biến đổi mẫu được che giấu (masked and corrupted sample) $\tilde{\mathbf{x}}$ sang dạng biểu diễn **z**. Sau đó một mô hình dự đoán pretext cố gắng khôi phục **x** từ **z**. So sánh với các pretext tasks được đề xuất cho dữ liệu dạng ảnh hay dữ liệu văn bản, như là phép xoay ảnh hay tô màu độ xám. Một hình ảnh bị xoay hoặc bị tô màu xám vẫn chứa thông tin về các features ban đầu, và có thể khôi phục thông qua mối quan hệ không gian giữa các features. Ngược lại, trong bài toán dạng bảng, việc thay thế **x** bằng $\tilde{\mathbf{x}}$, trong đó một số features được thay thế bằng các mẫu ngẫu nhiên khác trong bộ dữ liệu. Chính vì vậy, bài toán ban đầu được chia ra thành 2 bài toán nhỏ hơn (2 pretext tasks):

- Mask vector estimation: Dự đoạn những features đã bị che.
- Feature vector estimation: Dự đoán giá trị gốc ban đầu của những features bị che.

Tương ứng với 2 pretext tasks được đề xuất, 2 mô hình tương ứng được huấn luyện để giải quyết, bao gồm:

- **Mask vector estimator**, $s_m : Z \longrightarrow [0,1]^d$, nhận z làm input và output là một vector $\hat{m}$ dự đoán những features đã bị che đi.
- **Feature vector estimator**, $s_r : Z \longrightarrow X$, nhận z làm input và output là một vector $\hat{x}$, ước lượng so với mẫu gốc x.

Encoder e và 2 estimators bao gồm $s_m \& s_r$ được huấn luyện trong bài toán tối ưu hóa theo công thức ở dưới:

$$\min_{e, s_m, s_r} \mathbb{E}_{\mathbf{x} \sim p_{\mathbf{X}}, \mathbf{m} \sim p_{\mathbf{M}}, \tilde{\mathbf{x}} \sim g_{\mathbf{m}}(\mathbf{x}, \mathbf{m})} \left[l_m(\mathbf{m}, \hat{\mathbf{m}}) + \alpha \cdot l_r(\mathbf{x}, \hat{\mathbf{x}}) \right] \tag{2}$$

$$\hat{\mathbf{m}} = (s_m \circ e)(\tilde{\mathbf{x}}) \text{và } \hat{\mathbf{x}} = (s_r \circ e)(\tilde{\mathbf{x}}) \tag{3}$$

2 hàm mất mát l_m là tổng của mất mát cross-entropy nhị phân tương ứng với mỗi chiều trong mask vector.

$$l_m(\mathbf{m}, \hat{\mathbf{m}}) = -\frac{1}{d}\left[\sum_{j=1}^{d} m_j \log[(s_m \circ e)_j(\tilde{\mathbf{x}})] + (1-m_j)\log[1-(s_m \circ e)_j(\tilde{\mathbf{x}})]\right] \quad (4)$$

và hàm mất mát l_r:

$$l_r(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{d}\left[\sum_{j=1}^{d}(x_j - (s_r \circ e)_j(\tilde{\mathbf{x}}))^2\right] \quad (5)$$

α là trade-off giữa 2 hàm mất mát l_m và l_r.

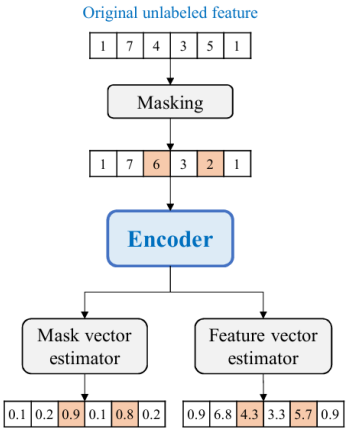

Figure 4. Huấn luyện một Encoder để giải quyết 2 pretext tasks, khôi phục dữ liệu mẫu đầu vào (Feature vector estimation), và mask vector dùng cho mẫu (Mask vector estimation)

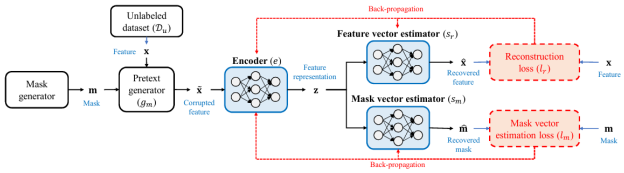

Figure 5. Chi tiết pha học tự giám sát trong mô hình VIME. Trong đó Mask generator tạo ra một một vector nhị nhân (binary mask vector), (**m**), sau đó được kết với hợp với một feature vector (**x**) để tạo ra một vector mới ($\tilde{x}$). Một Encoder (**e**) biến đổi $\tilde{x}$ sang **z**. Mask vector estimator (s_m) được huấn luyện để khôi phục mask vector (**m**), feature vector estimator (s_r) được huấn luyện để khôi phục mẫu đầu vào **x**. Cuối cùng, Encoder (**e**) sẽ được huấn luyện để giảm thiểu tổng hàm mất mát của khôi phục mask vector và feature vector.

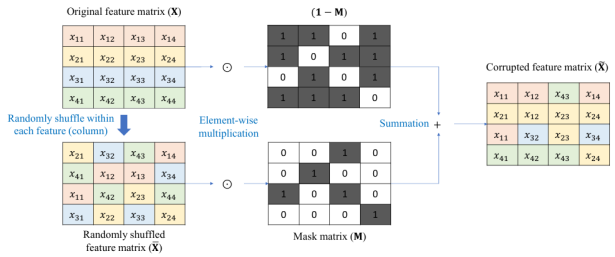

Figure 6. Pretext task cho dữ liệu dạng bảng

2) Pha học bán giám sát (Semi-supervised learning): Hình 7 mô tả chi tiết các bước tiến hành trong pha học bán giám sát (Semi-supervsied learning) của mô hình VIME, có sử dụng đến Encoder (e) đến từ pha học tự giám sát (Self-supervised learning) trước đó. Với $f_e = f \circ e$ và $\hat{y} = f_e(\mathbf{x})$., predictor f được huấn luyện để cực tiểu hóa hàm dưới đây:

$$\mathcal{L}_{final} = \mathcal{L}_s + \beta \cdot \mathcal{L}_u \quad (6)$$

Trong đó $\mathcal{L}_s$ là hàm supervised loss, được định nghĩa bởi công thức:

$$\mathcal{L}_s = \mathbb{E}_{(x,y)\sim p_{XY}}[l_s(y, f_e(\mathbf{x}))] \quad (7)$$

Ở trong phạm vi vấn đề được đặt ra bởi cuộc thi, là phân loại có thứ tự (oridnal classification), nhóm định nghĩa hàm mất mát l_s chính là Weighted Kappa Loss [9] được đề xuất năm 2017 bởi 3 nhà khoa học Jordi de la Torre, Domenec Puig, và Aida Valls dùng cho bài toán phân loại (multi-class classification) đối với dữ liệu thứ tự (ordinal data) trong các mô hình học sâu.

Hàm unsupervised loss hay consistency loss $\mathcal{L}_u$ được định nghĩa dựa theo công thức

$$\mathcal{L}_u = \mathbb{E}_{x\sim p_X, m\sim p_m, \tilde{x}\sim g_m(x,m)}[(f_e(\tilde{\mathbf{x}}) - f_e(\mathbf{x}))^2] \quad (8)$$

Tính xấp xỉ $\mathcal{L}_u$ được xác định bởi

$$\begin{aligned}\hat{\mathcal{L}}_u &= \frac{1}{N_b K}\sum_{i=1}^{N_b}\sum_{k=1}^{K}[(f_e(\tilde{\mathbf{x}}_{i,k}) - f_e(\mathbf{x}_i))^2] \\ &= \frac{1}{N_b K}\sum_{i=1}^{N_b}\sum_{k=1}^{K}[(f(\mathbf{z}_{i,k}) - f(\mathbf{z}_i))^2]\end{aligned} \quad (9)$$

Trong đó, N_b là batch size, K là số mẫu tăng cường $\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, ...\tilde{\mathbf{x}}_K$ được tạo từ mẫu gốc x thông qua bộ sinh pretext và bộ sinh mask vector.

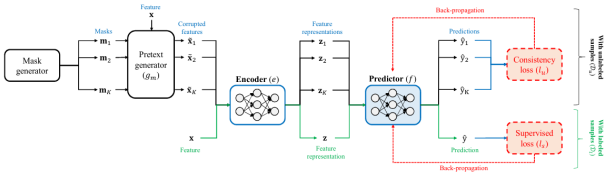

Figure 7. Chi tiết pha học bán giám sát trong mô hình VIME. (1) Mask generator tạo ra K mask vectors và kết hợp chúng với một mẫu x lấy ra từ bộ dữ liệu để biến đổi sang $\tilde{x}_k, k = 1, 2, ..., K$ thông qua bộ tạo pretext (g_m), (2) Encoder (e) được lấy từ pha học tự giám sát (Self-supervised learning) thực hiện biến đổi từ $\tilde{x}_k$ sang $z_k, k = 1, 2, ..., K$. (3) một mô hình được huấn luyện để tối thiểu hàm supervised loss (x, y) (Dữ liệu có gán nhãn), và consistency loss đối với $z_k, k = 1, 2, ..., K$

3) Gán nhãn dữ liệu khuyết nhãn: Sau khi hoàn thành huấn luyện mô hình VIME, tạo ra sau cùng là một predictor f cùng với đó là encoder e cho phép thực hiện gán nhãn đối với những mẫu khuyết nhãn trong bộ dữ liệu ban đầu. Sau khi gán nhãn, những mẫu được nhóm đánh giá là chất lượng sẽ được sử dụng để tạo bên bộ dữ liệu huấn luyện mới dành cho việc xây dựng mô hình học có giám sát (Supervised learning) ở phía sau. Bộ dữ liệu mới bao gồm các mẫu có nhãn trước đó, và các mẫu mới được gán nhãn chất lượng.

B. Xây dựng mô hình học có giám sát (Supervised learning)

Voting models Vì đặc thù dữ liệu, nhóm xem xét sử dụng 3 tree-based models khác nhau lần lượt LightGBM (LGBM), XGBoost (XGB) và CatBoost, và sau đó là kết hợp lại thông qua kỹ thuật ensemble vote nhằm giảm thiểu tình trạng overfitting. Các mô hình được khởi tạo là Regressor, với các tham số được chỉnh bằng công cụ RandomizedSearchCV, và sau đó là đưa vào Voting Regressor với trọng số lần lượt là 4.0, 4.0, 5.0 nhằm đưa ra kết quả cuối cùng là chỉ số ***sii***, chỉ số đánh giá mức độ lạm dụng Internet.

1) Mô hình LightGBM: LightGBM là một mô hình có cấu trúc cây dùng thuật toán Gradient Boosting phát triển bởi Microsoft [6]. Mô hình được xây dựng là một cây quyết định - decision tree và thường được dùng cho các bài toán phân loại. LGBM xây dựng cây theo hướng theo lá (leaf-wise) thay vì xây cây theo tầng (level-wise) hoặc theo độ sâu (depth-wise). Mô hình sẽ phát triển cây theo chiều đứng, tập trung vào các lá có độ mất mát cao và cải thiện độ chính xác. LGBM dùng thuật toán phân bố mật độ (histogram-based algorithm), tức là thay vì tìm điểm phân chia tập dữ liệu dựa trên việc sắp xếp các thuộc tính và phân chia dữ liệu dựa trên những thuộc tính liên tục thì nó dùng các 'bins' để chia nhỏ các thuộc tính thành các nhóm và dùng các nhóm để xây dựng các phân bố mật độ của thuộc tính.

$$\widetilde{V}_j(d) = \frac{1}{n}\left(\frac{\left(\sum_{x_i \in A_l} g_i + \frac{1-a}{b}\sum_{x_i \in B_l} g_i\right)^2}{n_l^j(d)} + \frac{\left(\sum_{x_i \in A_r} g_i + \frac{1-a}{b}\sum_{x_i \in B_r} g_i\right)^2}{n_r^j(d)} \right)$$

Figure 8. Công thức variance gain khi chia tập dữ liệu

Ngoài ra, mô hình còn sử dụng thuật toán *Gradient-based One-Side Sampling*, giúp tối ưu sự cân bằng giữa việc chọn độ lớn khi lấy tập con của data mà vẫn giữ được độ chính xác của việc xây cây quyết định. Thuật toán sẽ sắp xếp dữ liệu train bằng tối ưu hàm mất mát và chọn một tập con dữ liệu dựa trên đó. Với mỗi hàng dữ liệu, thuật toán sẽ tính gradient và thêm nó vào danh sách được sắp xếp theo thứ tự $\alpha\%$ gradient lớn và $(1-\alpha)\%$ gradient bé. Với những gradient được coi là lớn, thuật toán sẽ chọn chúng để phục vụ cho việc tìm điểm chia dữ liệu, còn với những gradient bé sẽ được chọn bởi một tham số gọi là *bagging friction*. Mô hình LGBM có các ưu điểm như có tốc độ train nhanh và hiệu quả nhờ vào các phương pháp histogram, sử dụng ít bộ nhớ vì chia dữ liệu vào các bins. Tuy có thể overfit trên tập dữ liệu nhỏ của cuộc thi, nhưng mô hình có thời gian chạy tốt và có thể tối ưu hóa các tham số nhằm đạt hiệu quả tốt hơn.

2) Mô hình XGBoost: XGBoost (Extreme Gradient Boosting) là một mô hình *distributed gradient boosting* được phát triển bởi các nhà khoa học ở Trường Đại học Washington, Mỹ nhằm hướng đến việc train hiệu quả hơn và có thể scale một cách dễ dàng [7]. Mô hình ngày càng được sử dụng nhiều trong việc giải các bài toán học máy và trở thành một trong các mô hình được sử dụng rộng rãi nhất dựa vào khả năng mở rộng, có hiệu suất tốt trên các tập dữ liệu lớn nhỏ khác nhau trong các bài toán phân lớp và hồi quy. XGB có khả năng giải quyết bài toán khi dữ liệu có nhiều trường bị thiếu, giúp mô hình có khả năng giải quyết các bài toán thực tế mà không cần các bước tiền xử lý quá phức tạp, và xử lý song song giúp mô hình có thể tăng khả năng xử lý các dữ liệu lớn giúp tiết kiệm thời gian hơn.

Mô hình hoạt động bằng cách áp trọng số vào từng biến độc lập, đưa chúng vào cây quyết định và khởi tạo các cây quyết định theo thứ tự. Các trọng số chưa hợp lý sẽ được tinh chỉnh bởi từng cây và đưa vào cây tiếp theo, sau đó áp dụng kỹ thuật ensemble để tạo nên một mô hình mạnh. Mô hình có thuật toán như sau:

1) Khởi tạo mô hình:

$$\hat{f}^{(0)}(x) = \arg\min_{\theta} \sum_{i=1}^{N} L(y_i, \theta).$$

2) Với $m = 1$ tới $m = M$:

a) Tính gradients và hessians:

$$\hat{g}_m(x_i) = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=\hat{f}^{(m-1)}(x)}$$

$$\hat{h}_m(x_i) = \left[\frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2}\right]_{f(x)=\hat{f}^{(m-1)}(x)}$$

b) Huấn luyện mô hình học máy yếu $\left\{x_i, \frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)}\right\}_{i=1}^{N}$ bằng cách giải phương trình sau:

$$\hat{\phi}_m = \arg\min_{\phi \in \Phi} \sum_{i=1}^{N} \frac{1}{2}\hat{h}_m(x_i)\left[\phi(x_i) - \frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)}\right]^2.$$

$$\hat{f}_m(x) = \alpha\hat{\phi}_m(x).$$

c) Cập nhật mô hình:

$$\hat{f}^{(m)}(x) = \hat{f}^{(m-1)}(x) - \hat{f}_m(x).$$

3) Kết quả:

$$\hat{f}(x) = \hat{f}^{(M)}(x) = \sum_{m=0}^{M} \hat{f}_m(x).$$

So với các thuật toán khác, như Random Forest sử dụng thuật toán *Boostrap Aggregating*, tức là xây dựng nhiều mô hình cây được train độc lập trên các tập dữ liệu con khác nhau, sau đó tổng hợp các dự đoán để đưa ra kết quả cuối cùng, giúp giảm khả năng overfitting và cải thiện độ chính xác. Với XGBoost, thuật toán lần lượt xây 1 cây tại một thời điểm, thay đổi các trọng số và cải thiện độ chính xác qua việc truyền tham số tuần tự qua các cây và thực hiện quá trình train, ngoài ra còn hỗ trợ các tham số L1 và L2 regularization giúp kiểm soát overfitting tốt hơn so với Random Forest. Với các ưu điểm như có thời gian chạy tốt, độ chính xác cao, có thể train trên các tập dữ liệu có các hàng bị thiếu thông tin,

và hỗ trợ tính toán song song giúp model trở thành một trong những mô hình học máy được sử dụng nhiều nhất tại các cuộc thi trên Kaggle.

3) Mô hình CatBoost: CatBoost (Categorical Boosting) cũng là một mô hình Gradient Boosting, hoạt động dựa trên hai thuật toán mới được phát triển bởi YanDex [8]. Mô hình được phát triển để giải các bài toán phân lớp hoặc hồi quy, với các dữ liệu thuộc tính (categorical) hay dữ liệu số (numberical) mà không cần các cách encoding truyền thống như One-Hot Encoding hay Label Encoding để chuyển kiểu của categorical về numerical. *Ordered Boosting* là thuật toán giúp tối ưu bài toán prediction shift, làm giảm target leakage bằng cách xây các hoán vị của dữ liệu gốc.

Algorithm 1 Ordered boosting

Require: $\{(\mathbf{x}_k, y_k)\}_{k=1}^{n}$, I
$\sigma \leftarrow$ random permutation of $[1, n]$
$M_i \leftarrow 0$ for $i = 1, \ldots, n$
for $t \leftarrow 1$ to I **do**
  for $i \leftarrow 1$ to n **do**
    $r_i \leftarrow y_i - M_{\sigma(i)-1}(\mathbf{x}_i)$
  end for
  for $i \leftarrow 1$ to n **do**
    $\Delta M \leftarrow \text{LearnModel}((\mathbf{x}_j, r_j) : \sigma(j) \leq i)$
    $M_i \leftarrow M_i + \Delta M$
  end for
end for
return M_n =0

Ordered target statistic là một thuật toán dựa trên việc lấy các mẫu của tập train theo tuần tự thời gian, tức là mỗi target statistic chỉ phụ thuộc vào observed history. Categorical features thường không thể được xử lý như dữ liệu số (numerical), do đó các mô hình tree boosting khác thường phải có bước tiền xử lý để chuyển chúng về dạng số. CatBoost có thể trực tiếp xử lý các cột đó thông qua tham số cat_columns, tự động hóa quá trình train mà không cần bước tiền xử lý. Mô hình sử dụng các phép đo để đánh giá độ chính xác như accuracy, precision, recall, F1-Score, ROC-AUC hay RMSE, giúp mô hình có thể giải quyết các bài toán hồi quy, phân lớp hay ranking. Ngoài ra, mô hình có đặc tính Symmetric trees (cây đối xứng), giúp làm tăng tốc quá trình huấn luyện và làm giảm tiêu thụ bộ nhớ vì các splits sẽ được thực hiện ở cùng một thuộc tính tại tất cả các nodes, phù hợp cho nhiều tập dữ liệu lớn nhỏ khác nhau, cùng với đó là tối ưu quá trình tính toán song song.

4) Voting Regressor: Sau khi đã tối ưu tham số, ba mô hình tree-based sẽ được khởi tạo thành các Regressors tương ứng, và nạp vào trong một hàm Voting Regressor để thực hiện phương pháp ensemble. Ensemble learning là một phương pháp học máy, ở bài này thì nhóm sử dụng ba mô hình độc lập, không liên quan đến nhau, không có sự tương tác giữa các mô hình trong quá trình train vì có độ phức tạp thấp, phù hợp với thời gian chạy. Voting Regressor có thể tận dụng thế mạnh của các mô hình, giúp hạn chế overfitting và đưa ra kết quả ổn định hơn. Nhóm đã khảo sát và lựa chọn các trọng số cho voting, lần lượt là ***4.0, 4.0, 5.0*** cho LightGBM, XGBoost và CatBoost.

	CatBoost	LightGBM	XGBoost
Categorical Features	Tự động xử lý mà không cần bước tiền xử lý	Dùng one-hot encoding	Cần bước tiền xử lý
Cách chia cây	Symmetric	Leaf-wise	Depth-wise
Độ hiệu quả	Tối ưu tốc độ và bộ nhớ	Tối ưu cho tập dữ liệu lớn	Tối ưu tốc độ và scalable

Figure 9. So sánh các model tree-based được sử dụng

CatBoost có trọng số cao hơn so với 2 mô hình còn lại, nhờ vào thuật toán ordered boosting giúp hạn chế overfitting, có những tùy chỉnh dành cho các tập dữ liệu nhỏ - đặc biệt ở bài toán này khi mà tập train chỉ có khoảng 4000 mẫu. Với Voting Regressor, nhóm đã tiến hành thử nghiệm và cho kết quả ổn định trên các phương pháp tiền xử lý dữ liệu khác nhau.

5) Hyperparameters tuning: Việc điều chỉnh tham số cho các mô hình là quan trọng, để có kết quả tốt trên bài toán CMI - PIU, nhóm đã tìm hiểu một vài phương pháp parameters tuning như GridSearchCV hay RandomizedSearchCV. GridSearchCV là một phương pháp chỉnh siêu tham số phát triển trên thư viện Scikit-learn, bằng cách định nghĩa một khoảng ước lượng cho các tham số, ví dụ như ***'lambda_l1': [8, 10, 12], 'lambda_l2': [0.005, 0.01, 0.02], 'n_estimators': [200, 250, 300]***. GridSearchCV sẽ thử với một tổ hợp tất cả các tham số đó, nhằm kiểm tra toàn diện, không bỏ sót bất kì giá trị nào, phù hợp khi mà đã biết được ước lượng cho phạm vi của tham số. Tuy nhiên, nó tốn quá nhiều thời gian để đánh giá từng tổ hợp, làm tăng thời gian chạy lên đáng kể, đặc biệt với bộ dữ liệu của cuộc thi. Nhóm đã thử áp dụng GridSeachCV với cú pháp

```
lgbm_grid = GridSearchCV(estimator=
lgbm_model, param_grid=lightgbm_param_grid,
scoring='accuracy', cv=3, verbose=2)
```

đánh giá bằng phương pháp cross-validation với 3 folds, nhận lại ước tính thời gian chạy là khoảng 75000 giây cho toàn bộ tập dữ liệu với mỗi mô hình, nếu chạy trên cả ba mô hình thì thời gian chạy hoàn toàn không đáp ứng được tiêu chí của cuộc thi và cũng như không thể có thời gian cho nhóm có thể đánh giá kết quả do GridSeachCV có thể kiểm tra nhiều tổ hợp không cần thiết và ảnh hưởng đến hiệu suất chung. Vì thế, nhóm đã đưa vào sử dụng phương pháp RandomizedSearchCV, thay vì kiểm tra tất cả các tổ hợp thì chỉ đưa một vài tổ hợp ngẫu nhiên để đánh giá. Với phương pháp này, nhóm có thể tiết kiệm nhiều thời gian chạy hơn và có thể tìm ở nhiều vùng tham số khác nhau, với cú pháp

```
lgbm_random = Randomized-
SearchCV(estimator=lgbm_model,
param_distributions=lightgbm_param_grid,
scoring='accuracy', cv=3, n_iter=10,
```

`verbose=2, random_state=SEED)`

Bằng cách lựa chọn số lần folds và số lần lặp, chương trình chỉ cần chạy 30 vòng với mỗi mô hình, giảm thời gian chạy xuống đáng kể mà vẫn giúp nhóm có thể tìm được tham số tối ưu.

IV. Kết quả

A. Phương pháp đánh giá

Theo quy định của cuộc thi, kết quả dự đoán của các mô hình học máy sẽ được đánh giá dựa trên hàm quadratic weighted kappa (QWK) và xếp hạng dựa trên thứ tự từ cao đến thấp trên leaderboards bằng kết quả chạy trên tập dữ liệu private. Điểm QWK có phân bố từ 0 đến 1, điểm càng cao thì độ chấp thuận (agreement) của hai đối tượng càng lớn, nếu độ chấp thuận quá thấp thậm chí hai đối tượng đối nghịch nhau, điểm QWK có thể đạt ngưỡng âm.

Để tính điểm QWK, trước hết tạo ba ma trận O, W, E, trong đó ma trận O là một ma trận tần suất N x N với O_{ij} là số đối tượng có giá trị thực tế là i và giá trị dự đoán là j. Ma trận W là ma trận trọng số N x N, được tính bằng công thức sau:

$$W_{i,j} = \frac{(i-j)^2}{(N-1)^2}$$

Ma trận E là ma trận tần suất cho đầu ra kỳ vọng, giả sử không có sự tương quan giữa các giá trị, được tính bởi tích ngoài của ma trận tần suất cho giá trị thực tế và ma trận tần suất của giá trị dự đoán. Điểm số cuối cùng sẽ được tính như sau:

$$\kappa = 1 - \frac{\sum_{i,j} W_{i,j} O_{i,j}}{\sum_{i,j} W_{i,j} E_{i,j}}.$$

B. Kết quả

Bảng I là kết quả chạy của 2 notebooks được nộp tại kì thi, lần lượt là "Dữ liệu có nhãn" và "Dữ liệu khuyết nhãn và có nhãn". Nhóm so sánh hiệu quả giữa việc ứng dụng VIME để tạo thêm những mẫu có nhãn chất lượng và việc chỉ dùng những mẫu có nhãn từ bộ dữ liệu gốc.

- **Dữ liệu có nhãn**: Tại notebook này, nhóm chỉ xây dựng 2 pha bao gồm pha xử lý dữ liệu (Xem ở mục II Xử lý dữ liệu) và xây dựng pha học có giám sát (Xem ở mục III.2).
- **Dữ liệu khuyết nhãn và có nhãn**: Nhóm triển khai phương pháp được đề xuất ở báo cáo, gồm 3 pha lần lượt là (1) Xử lý dữ liệu, (2) Ứng dụng VIME gán nhãn cho mẫu khuyết nhãn và (3) Xây dựng mô hình học giám sát (Supervised learning).

Nhóm sử dụng phương pháp Stratified K-Fold Cross-Validation cho quá trình training:

- Chia bộ dữ liệu thành k tập con
- Tính phân bố của các nhãn nhằm đảm bảo việc chia dữ liệu vẫn bảo toàn được phân bố dữ liệu.

Table I
KẾT QUẢ CHẠY HAI NOTEBOOK, TÍNH THEO ĐIỂM QWK

	Train	Test	Optimized	Public	Private
Dữ liệu có nhãn	0.7718	0.3837	0.460	0.432	0.456
Dữ liệu khuyết nhãn và có nhãn	0.8174	0.5371	0.591	0.391	0.415

- Với mỗi vòng lặp, dùng 1 fold để làm tập valid và (k-1) folds còn lại để phục vụ training, sau đó tính ra điểm QWK trung bình.

Hai notebook trên được chạy trên môi trường Conda được cung cấp bởi Kaggle, với cấu hình CPU Intel Xeon 2.00GHz 2 cores, 32GB RAM, NVIDIA T4 x2 GPU, trong đó thời gian chạy cho notebook đầu tiên là 8 phút 35 giây, trong khi đối với notebook thứ 2 dùng VIME là 19 phút. Kết quả chạy của nhóm được thể hiện qua Bảng I.

Nhìn vào bảng kết quả I và so sánh giữa hai notebooks, có thể thấy, việc dùng VIME để gán nhãn tạo tăng thêm lượng dữ liệu huấn luyện giúp cho mô hình học có giám sát (Supervised learning) đạt được điểm cao hơn ở tập huấn luyện 0.8174 và tập xác thực (validation) 0.5371, tuy nhiên khi được đánh giá ở tập dữ liệu test bao gồm cả tập test công khai và tập test bí mật thì việc chỉ sử dụng dữ liệu có nhãn ban đầu cho ra kết quả tốt hơn, với số điểm lần lượt là 0.432 và 0.456 khi so sánh với khi sử dụng VIME là 0.391 và 0.415

Và tại notebook thứ nhất, kết quả đạt được trên tập test ẩn là 0.456 giúp cho nhóm đạt xếp hạng 96 trên bảng xếp hạng cuộc thi (xem hình 10) và chứng nhận Silver Medal đến từ Kaggle.

Figure 10. Xếp hạng tại cuộc thi

V. Kết luận

Mô hình VIME đã được sử dụng và đem đến kết quả mang tính đột phá ở những bộ dữ liệu truyền thống nổi tiếng như bộ dữ liệu viết tay MNIST (chuyển từ ảnh về bảng), UCI Income, UCI Blog, bộ dữ liệu về gene liên quan đến 6 đặc điểm tế bào máu MRV, MPV, MCH, RET, PCT, MONO (Xem chi tiết tại bài báo VIME: Extending the Success of Self- and Semi-supervised Learning to Tabular Domain). Đó đều là những bộ dữ liệu đã được thu thập và tinh chỉnh. Nhưng trong thực tế bài toán thu thập dữ liệu rất khó để có thể đạt đến trạng thái lý tưởng, như ở bài toán được đề xuất trong cuộc thi CMI - PIU với lượng dữ liệu bị thiếu ở các cột predictors (Xem lại mục II.1 - Tổng quan dữ liệu và hình 1). Nhóm đã đánh giá và nhìn nhận những hạn chế trong việc ứng dụng mô hình VIME vào bài toán

- **Hạn chế về mặt dữ liệu**: Sự thiếu xót dữ liệu ở các cột predictors, mặc dù đã có cơ chế loại bỏ những predictors với lượng dữ liệu thiếu xót nhiều (Trên 80%) và xây dựng KNNImputer để bổ khuyết phần còn thiếu cho những predictors còn lại. Song điều đó cũng là một thách thức đối với mô hình VIME để có thể học được chính xác

"thông tin biểu diễn" (representative information) của dữ liệu thực tế.

- **Hạn chế về cơ chế chọn mẫu có nhãn mới**: Về mặt lý thuyết, bổ sung thêm nhiều mẫu chất lượng sẽ tăng tính tổng quát của mô hình, đạt được hiệu quả tốt hơn trên cả những dữ liệu mới chưa được học, hay còn gọi là tập test. Nhóm nhìn nhận còn những hạn chế trong việc lọc ra những dữ liệu có nhãn mới, chất lượng, để thêm vào bộ dữ liệu huấn luyện mô hình học có giám sát (Supervised learning). Nhóm cần xây dựng một bộ khung hoàn chỉnh để đánh giá chất lượng của những mẫu được gán nhán, hay gọi là mức độ tự tin của dự đoán đến từ VIME.

Mở rộng: Tuy nhiên, không thể phủ định được rằng, với một bộ dữ liệu thiếu xót nghiêm trọng, thì việc xây dựng một mô hình đạt hiệu suất lý tưởng gần như là bất khả thi. Thay vào đó, đích đến của cuộc thi là một mô hình ổn định, có hiệu quả tương đối, minh chứng là điểm số của 5 vị trí dẫn đầu lần lượt $0.482, 0.478, 0.478, 0.477, 0.477$. Sau khi xem xét các bài nộp đứng đầu, nhóm đã kết luận được rằng chìa khóa của cuộc thi nằm ở thao tác xử lý dữ liệu, bao gồm làm sạch, loại bỏ những predictors không ổn định, và áp dụng góc nhìn khoa học vào kỹ thuật Feature Engineering và kết hợp với kỹ thuật điều chỉnh tham số phù hợp vào các mô hình phân loại được sử dụng phổ biến bao gồm LightGBM, XGBoost và CatBoost. (Xem chi tiết ở các bài nộp đứng đầu Top 1, Top 2, Top 3, Top 4). Xác định được cốt lõi của bài toán, xây dựng 2 pha gồm pha Xử lý dữ liệu và pha Xây dựng mô hình học có giám sát (Supervised learning), đã giúp nhóm tiến vào vị trí 96 ở cuộc thi với số điểm 0.456.

REFERENCES

[1] Yoon, J., Zhang, Y., Jordon, J. and van der Schaar, M. VIME: Extending the Success of Self- and Semi-supervised Learning to Tabular Domain. *Advances in Neural Information Processing Systems*, Vol. 33, pp. 11033-11043 (2020)

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[3] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. arXiv preprint arXiv:1803.07728, 2018.

[4] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In European Conference on Computer Vision, pages 69–84. Springer, 2016.

[5] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In European conference on computer vision, pages 649–666. Springer, 2016.

[6] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and, Liu, T.-Y. (2017) LightGBM: A Highly Efficient Gradient Boosting Decision Tree. Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, December 2017, 3149-3157.

[7] Chen, T. & Guestrin, C. XGBoost: A Scalable Tree Boosting System. *Proceedings Of The 22nd ACM SIGKDD International Conference On Knowledge Discovery And Data Mining*. pp. 785-794 (2016,8), http://dx.doi.org/10.1145/2939672.2939785

[8] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. & Gulin, A. CatBoost: unbiased boosting with categorical features. (2019), https://arxiv.org/abs/1706.09516

[9] de la Torre, J., Puig, D. and Valls, A. Weighted kappa loss function for multi-class classification of ordinal data in deep learning. *Pattern Recognition Letters*, Vol. 105, pp. 144-154 (2018), https://doi.org/10.1016/j.patrec.2017.05.018