Trường đại học Công nghệ - ĐHQGHN

Khoa Công nghệ Thông tin

NGHIÊN CỨU VỀ PHƯƠNG PHÁP DỰ ĐOÁN KHẢ NĂNG NGỪNG SỬ DỤNG DỊCH VỤ VIỄN THÔNG CỦA KHÁCH HÀNH DỰA TRÊN ENSEMBLE LEARNING SỬ DỤNG FEATURE GROUPING

Hoàng Bảo Long (22024556)

Nguyễn Anh Đức (22024536)

Môn học: **Phân tích dữ liệu dự báo**

Giảng viên: **TS. Nguyễn Thị Hậu**

Tóm tắt

Trong những năm gần đây, thị trường viễn thông đã trở nên rất cạnh tranh. Chi phí để giữ chân khách hàng hiện tại thấp hơn so với việc thu hút khách hàng mới. Do đó, việc các công ty viễn thông cần hiểu rõ về tình trạng khách hàng rời bỏ dịch vụ thông qua hệ thống quản lý quan hệ khách hàng CRM là cần thiết. Vì vậy, các nhà phân tích CRM được yêu cầu dự đoán những khách hàng nào có khả năng rời bỏ dịch vụ.

Trong những năm gần đây, thị trường viễn thông đã trở nên rất cạnh tranh. Chi phí để giữ chân khách hàng hiện tại thấp hơn so với việc thu hút khách hàng mới. Do đó, việc các công ty viễn thông cần hiểu rõ về tình trạng khách hàng rời bỏ dịch vụ thông qua hệ thống quản lý quan hệ khách hàng CRM là cần thiết. Vì vậy, các nhà phân tích CRM được yêu cầu dự đoán những khách hàng nào có khả năng rời bỏ dịch vụ.

Nghiên cứu này đề xuất một hệ thống dự đoán khách hàng rời bỏ sử dụng kỹ thuật học máy tập hợp (ensemble learning) bao gồm mô hình stacking và soft voting. Các thuật toán học máy bao gồm XGBoost, hồi quy logistic, cây quyết định, và Naïve Bayes được chọn để xây dựng mô hình stacking gồm hai cấp độ, trong đó ba kết quả của cấp độ thứ hai sẽ được sử dụng cho soft voting. Quá trình xây dựng đặc trưng của tập dữ liệu khách hàng rời bỏ bao gồm việc phân nhóm khoảng cách đều các đặc trưng hành vi của khách hàng nhằm mở rộng không gian đặc trưng và khám phá thông tin tiềm ẩn từ tập dữ liệu.

Tập dữ liệu khách hàng rời bỏ ban đầu và tập dữ liệu mới được phân tích trong mô hình stacking với bốn chỉ số đánh giá khác nhau. Kết quả thực nghiệm cho thấy hệ thống dự đoán khách hàng rời bỏ đề xuất đạt độ chính xác 96.12% đối với tập dữ liệu ban đầu và 98.09% đối với tập dữ liệu mới. Những kết quả này tốt hơn so với các hệ thống nhận diện khách hàng rời bỏ tiên tiến hiện nay.

Từ khóa: khách hàng rời bỏ; CRM; học máy; học tập tập hợp; phân nhóm đặc trưng.

Mục lục

Danh sách hình vẽ

Danh sách bảng

Chương 1

Giới thiệu

1.1 Giới thiệu bài toán

Do sự cạnh tranh gay gắt giữa các công ty viễn thông, việc khách hàng rời bỏ dịch vụ là không thể tránh khỏi. Customer churn là hành động mà một khách hàng chấm dứt đăng ký dịch vụ của một nhà cung cấp và chuyển sang sử dụng dịch vụ của một công ty khác. Các công ty cần giảm thiểu tình trạng khách hàng rời bỏ, vì điều này làm suy yếu công ty. Một khảo sát cho thấy tỷ lệ khách hàng rời bỏ hàng năm trong ngành viễn thông dao động từ 20% đến 40%, và chi phí giữ chân khách hàng hiện tại thấp hơn từ 5 đến 10 lần so với chi phí thu hút khách hàng mới (García et al., 2017). Chi phí dự đoán khách hàng có khả năng rời bỏ cũng thấp hơn 16 lần so với chi phí thu hút khách hàng mới (Borja et al., 2013). Giảm tỷ lệ rời bỏ khách hàng xuống 5% có thể tăng lợi nhuận từ 25% đến 85% (Kotler, 1994). Điều này cho thấy rằng dự đoán khách hàng rời bỏ là yếu tố quan trọng trong ngành viễn thông.

Các công ty viễn thông coi quản lý quan hệ khách hàng (CRM) là yếu tố quan trọng trong việc giữ chân khách hàng hiện tại và ngăn chặn tình trạng khách hàng rời bỏ dịch vụ. Để giữ chân khách hàng, các nhà phân tích CRM cần dự đoán những khách hàng có khả năng rời bỏ và phân tích nguyên nhân dẫn đến việc này. Sau khi xác định được các khách hàng có nguy cơ cao, công ty cần triển khai các chiến dịch marketing để giữ chân những khách hàng này một cách tối đa. Do đó, dự đoán khách hàng rời bỏ là một phần quan trọng của CRM (Ngai et al., 2009)

Độ chính xác của các hệ thống dự đoán mà các nhà phân tích CRM sử dụng là rất quan trọng. Nếu các dự đoán không chính xác, công ty sẽ không thể thực hiện các chiến dịch giữ chân khách hàng. Nhờ sự phát triển gần đây của khoa học dữ liệu, khai phá dữ liệu và công nghệ học máy đã cung cấp các giải pháp cho bài toán này. Tuy nhiên, có một số hạn chế trong các mô hình hiện tại. Ví dụ, hồi quy logistic, một mô hình dự đoán churn phổ biến dựa trên các phương pháp khai phá dữ liệu cũ, có độ chính xác tương đối thấp. Hơn nữa, xây dựng đặc trưng (Motoda and Liu, 2001) thường bị bỏ qua trong quá trình phát triển mô hình. Vì vậy, cần có một hệ thống dự đoán churn tốt hơn.

Nghiên cứu này đề xuất một hệ thống dự đoán khách hàng rời bỏ mới cùng với việc xây dựng đặc trưng để cải thiện độ chính xác. Các đóng góp của nghiên cứu này có thể được tóm tắt như sau:

- Hệ thống dự đoán mới dựa trên học tập tập hợp (ensemble learning) với độ chính xác tương đối cao được đề xuất.
- Các đặc trưng mới được tạo ra từ việc phân nhóm khoảng cách đều (equidistant grouping)

các đặc trưng hành vi của khách hàng được sử dụng để cải thiện hiệu suất của hệ thống.

1.2 Phát biểu bài toán

Bài toán có thể được mô tả như sau: Đầu vào của là thông tin của người dùng (các thông tin cá nhân, cũng như là lịch sử khách hàng). Nhiệm vụ của bài toán là dự đoán khả năng ngừng sử dụng dịch vụ của người dùng trong tương lai

- Đầu vào: $X = [x_1, x_2, ...x_n]$ là biểu diễn các thông tin của người dùng, mỗi một giá trị x_i là mỗi đặc trưng
- Đầu ra: $Y \in \{0, 1\}$ Trong đó Y là nhãn của người dùng, 1 là người dùng có khả năng ngừng sử dụng dịch vụ trong tương lại, 0 là người dùng không có khả năng ngừng sử dụng dịch vụ trong tương lai.

1.3 Một số vấn đề liên quan

Quá trình triển khai thu thập và xử lí dữ liệu của bài toán sẽ có một số những vấn đề chính như sau:

1. Dữ liệu bị mất cân bằng: Dữ liệu bị mất cân bằng là một vấn đề phổ biến tồn tại trong những bài toán như trên. Việc dữ liệu hoàn toàn bị thiên lệch về một nhãn (trong bài toán này thì là nhãn 0 - tức là khách hàng không rời bỏ) là điều gần như chắc chắn sẽ xảy ra
2. Dữ liệu tồn tại ngoại lai: Điểm ngoại lai (outliers) là nhữ điểm dữ liệu hay quan sát (observation) trong dữ liệu có sự khác biệt rõ rệt so với các giá trị khác trong tập dữ liệu. Chúng có thể là các giá trị rất cao hoặc rất thấp so với phần lớn dữ liệu và có thể xuất hiện do nhiều nguyên nhân như lỗi đo lường, lỗi nhập liệu, hoặc là những đặc trưng thật sự của dữ liệu.
3. Dữ liệu không đồng nhất: trong trường hợp của bài toán này, dữ liệu không đồng nhất có thể được hiểu là những dữ liệu không có cùng một kiểu dữ liệu (data type), có thể là dữ liệu dạng số (numerical), dạng phân loại (categorical), hoặc là dữ liệu dạng nhị phân (0 và 1).

Các vấn đề trên cần được xử lí một cách cần trọng trong quá trình tiền xử lí dữ liệu, cũng như tinh chỉnh các mô hình sao cho hợp lí, để có thể cho ra một kết quả đủ tốt và đủ tin cậy.

1.4 Các nghiên cứu liên quan

Nhiều phương pháp như học máy và khai phá dữ liệu đã được sử dụng cho bài toán dự đoán khách hàng rời bỏ. Thuật toán cây quyết định là một phương pháp đáng tin cậy trong việc dự đoán churn (Edwards et al., n.d.). Ngoài ra, các phương pháp như mạng nơ-ron (Sharma and Panigrahi, 2013), data certainty (Amin et al., 2019), và tối ưu hóa bầy đàn (PSO) (Vijaya and Sivasankar, 2019) cũng đã được áp dụng trong dự đoán churn.

Hơn nữa, một nghiên cứu so sánh giữa mạng nơ-ron nhân tạo (ANN) và cây quyết định cho dự đoán churn của khách hàng (Umayaparvathi and Iyakutti, 2012) cho thấy rằng thuật toán cây quyết định vượt trội hơn so với ANN trong bài toán này.

A.T. Jahromi (Jahromi et al., 2010) đã nghiên cứu tác động của lòng trung thành khách hàng lên khả năng rời bỏ dịch vụ của các công ty điện thoại trả trước. Trong nghiên cứu này, các đặc

trưng đã được phân đoạn và nhiều thuật toán như cây quyết định và mạng nơ-ron được sử dụng để dự đoán dữ liệu đã qua xử lý. Kết quả cho thấy rằng phương pháp kết hợp (hybrid approach) hiệu quả hơn so với việc sử dụng từng thuật toán đơn lẻ. KNN-LR là một phương pháp kết hợp sử dụng hồi quy logistic và K-nearest neighbor (KNN) (Zhang et al., 2007). Các nhà nghiên cứu đã khảo sát KNN-LR, hồi quy logistic và mạng nền tảng hàm cơ sở xuyên tâm (RBF), và nhận thấy rằng KNN-LR cho kết quả tốt nhất.

Y. Zhang (Reichheld and Sasser, 1990) đã đề xuất một khung phân tán cho các kỹ thuật khai phá dữ liệu nhằm dự đoán churn. Khung này cải thiện chất lượng dịch vụ CRM.

Ruba Obeidat đã đề xuất một phương pháp kết hợp dựa trên lập trình di truyền (Obiedat et al., 2013). Nghiên cứu này sử dụng thuật toán K-means và lập trình di truyền để dự đoán khách hàng rời bỏ. Sahar F. Sabbeh đã sử dụng AdaBoost dựa trên thuật toán boosting (Sabbeh, 2018), trong đó tóm tắt các kỹ thuật học máy hiện có và chỉ ra rằng AdaBoost cho kết quả tốt nhất. Hossam Faris đã đề xuất một mô hình mạng nơ-ron thông minh kết hợp (Faris, 2018), trong đó kết hợp tối ưu hóa bầy đàn với mạng nơ-ron lan truyền tiến để dự đoán churn. Kết quả cho thấy mô hình có thể cải thiện khả năng nhận diện khách hàng rời bỏ.

Tổng kết chương:
Chương 1 đã giới thiệu cái nhìn tổng quan, cũng như các nghiên cứu liên quan về bài toán Dự đoán khả năng ngừng sử dụng dịch vụ của người dùng trong lĩnh vực viễn thông. Chương hai sẽ trình bày cơ sở lý thuyết về các thuật toán phân lớp được sử dụng để xây dựng mô hình giải quyết bài toán

Chương 2

Cơ sở lý thuyết mô hình thuật toán phân lớp

2.1 Hồi quy Logistic (Logistic Regression)

Hồi quy Logistic (Chatterjee and Simonoff, 2020) là một thuật toán học máy phổ biến thuộc loại học có giám sát (supervised learning) dùng để phân loại. Mặc dù tên của nó chứa từ ”hồi quy,”nó thực sự được sử dụng cho các bài toán phân loại nhị phân và đa lớp (multiclass), chứ không phải cho các bài toán hồi quy như ”hồi quy tuyến tính.”

Hồi quy Logistic dựa trên ý tưởng sử dụng một hàm Sigmoid hoặc Logistic function để ánh xạ bất kỳ giá trị thực nào thành xác suất giữa 0 và 1. Từ đó, nó có thể được sử dụng để phân loại vào hai lớp khác nhau (nhị phân). Hàm Sigmoid có công thức cụ thể như sau:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2.1}$$

trong đó:

- z là tổ hợp tuyến tính của các biến đầu vào (features) và các trọng số (weights), tức là $z = w_1x_1 + w_2x_2 + ... + w_nx_n + w_0$
- e là hằng số Euler, có giá trị xấp xỉ bằng 2.71828

Mô hình sử dụng hàm Sigmoid để tính xác suất rằng một mẫu thuộc về một lớp cụ thể. Nếu xác suất đó lớn hơn một ngưỡng (thường là 0.5), mẫu đó sẽ được gán vào lớp ”1”, nếu không thì gán vào lớp ”0”.

Hồi quy logistic sử dụng hàm mất mát cross-entropy (còn gọi là log-loss):

$$Loss = -\frac{1}{N}\sum_{i=1}^{N}\Big(y_i log(p_i) + (1 - y_i) log(1 - p_i)\Big) \tag{2.2}$$

trong đó:

- y_i: Nhãn thực của mẫu thứ i (0 hoặc 1).
- p_i: Xác suất dự đoán rằng mẫu thứ i thuộc về lớp 1.

Hồi quy logistic giả định rằng mối quan hệ giữa biến phụ thuộc và các biến độc lập có thể được mô hình hóa tuyến tính thông qua một tổ hợp tuyến tính, sau đó được ánh xạ bằng hàm sigmoid. Các biến độc lập không nên có tương quan cao với nhau (vấn đề đa cộng tuyến - multicollinearity).

2.2 Cây quyết định (Decision Tree)

Thuật toán cây quyết định (Suzuki, 2020) là một thuật toán học máy có giảm sát, có thể được áp dụng vào cả hai bài toán phân lớp và hồi quy. Trong thuật toán cây quyết định, quá trình phân lớp được mô hình quá thông qua việc sử dụng một tập các quyết định theo thứ bậc dựa trên các biến đặc trưng và được sắp xếp theo cấu trúc cây.

Trong cây quyết định có hai loại nút khác nhau, nút trong cây và nút lá. Các quyết định của cây, hay còn được gọi là điều kiện rẽ nhánh, được thể hiện tại các nút trong cây. Các nút lá các nút thể hiện đầu ra của thuật toán cây quyết định. Về tổng quan, thuật toán của cây quyết định sẽ bắt đầu với toàn bộ tập dữ liệu huấn luyện tại nút ngọn của cây sau đó thực hiện đệ quy phân chia dữ liệu xuống các nút trong cây phía dưới dựa trên các điều kiện rẽ nhánh. Sau cùng, thuật toán của cây quyết định dừng lại khi đã đạt điều kiện dừng.

Các thuộc tính được chọn phân chia dữ liệu sao cho giảm thiểu độ không chắc chắn (uncertainty) hoặc tăng tính thuần khiết (purity) của các nhánh. Mục tiêu của điều kiện rẽ nhánh là có thể tối đa hóa được sự tách biệt giữa các lớp trong các nút con. Việc xây dựng các điều kiện rẽ nhánh dựa trên bản chất của các đặc trưng:

- Đặc trưng nhị phân (Binary feature): Chỉ có một điều kiện rẽ nhánh duy nhất, và cây quyết định là cây nhị phân. Mỗi nhánh của nút tương ứng với một trong hai giá trị nhị phân.
- Đặc trưng dạng phân loại (Categorical feature): Nếu một đặc trưng dạng phân loại có r giá trị khác nhau, có nhiều điều kiện rẽ nhánh khác nhau. Cách tiếp cận đơn giản là có thể sử dụng r điều kiện rẽ nhánh khác nhau với mỗi nhánh là một giá trị cụ thể của đặc trưng, hoặc có thể sử dụng các điều kiện rẽ nhánh nhị phân thông qua việc kiểm tra $2^r - 1$ tổ hợp (hoặc nhóm) các giá trị của đặc trưng và lựa chọn tổ hợp tốt nhất, tuy nhiên phương pháp này tốn nhiều thời gian khi số r lớn.
- Đặc trưng dạng số: Nếu đặc trưng dạng số chỉ bao gồm r giá trị rời rạc, có thể tạo ra r điều kiện rẽ nhánh khác nhau cho mỗi giá trị rời rạc, tuy nhiên đối các đặc trưng số liên tục, điều kiện rẽ nhánh thường được thực hiện bằng một điều kiện nhị phân, $x \leq a$ với x là giá trị thuộc đặc trưng và a là một giá trị cố định.

Có nhiều phương pháp được áp dụng tìm điều kiện rẽ nhánh tốt nhất từ các đặc trưng của bộ dữ liệu và từ các giá trị của từng đặc trưng:

- Tỷ lệ lỗi (Error rate): Đặt p là tỉ lệ nhãn chiếm đa số trong phần dữ liệu S, tỷ lệ lỗi được định nghĩa bằng $1 - p$. Nếu phần dữ liệu S có r điều kiện rẽ nhánh và chia phần dữ liệu thành các phần dữ liệu con $S_1, ...S_r$, tỉ lệ lỗi của phần dữ liệu S ban đầu bằng trung bình có trọng số tỷ lệ lỗi của các phần dữ liệu con, với trọng số là số điểm dữ liệu trong các phần dữ liệu con. Điều kiện rẽ nhánh có tỷ lệ lỗi thấp nhất sẽ được lựa chọn.
- Chỉ số Gini (Gini Index): Chỉ số Gini $G(S)$ của phần dữ liệu S được định nghĩa theo công thức:

$$G(S) = 1 - \sum_{j=1}^{k} {p_j}^2 \tag{2.3}$$

  với k là số các nhãn có trong phần dữ liệu S, $p_1, ...p_k$ là phân phối các nhãn của phần dữ liệu S. Chỉ số Gini tổng thể của phần dữ liệu S, nếu có r điều kiện rẽ nhánh và chia phần dữ

liệu thành các phần dữ liệu con $S_1, ..., S_r$, được tính bằng trung bình có trọng số các chỉ số Gini của từng phần dữ liệu con, với trọng số là số điểm dữ liệu trong các phần dữ liệu con:

$$Gini - split(S \to S_1, ..., S_r) = \sum_{i=1}^{r} \frac{|S_i|}{|S|} G(S_i) \tag{2.4}$$

với k là số các nhãn có trong phần dữ liệu S, $|S|$ là số điểm dữ liệu có trong phần dữ liệu S, $|Si|$ là số điểm dữ liệu có trong phần dữ liệu con S_i. Điều kiện rẽ nhánh có chỉ số Gini thấp nhất sẽ được lựa chọn.

- Entropy: Phương pháp tính entropy $E(S)$ cho phần dữ liệu S được định nghĩa như sau:

$$E(S) = -\sum_{j=1}^{k} p_j log_2(p_j) \tag{2.5}$$

với k là số các nhãn có trong phần dữ liệu S, $p_1, ..., p_k$ là phân phối các nhãn của phần dữ liệu S. Phương pháp tìm entropy tổng thể của phần dữ liệu S, nếu có r điều kiện rẽ nhánh và chia phần dữ liệu thành các phần dữ liệu con $S_1, ..., S_r$, được tính bằng trung bình có trọng số các Entropy của từng phần dữ liệu con, vớitrọng số là số điểm dữ liệu trong các phần dữ liệu con:

$$Entropy - split(S \to S_1, ..., S_r) = \sum_{i=1}^{r} \frac{|S_i|}{|S|} E(S_i) \tag{2.6}$$

với k là số các nhãn có trong phần dữ liệu S, $|S|$ là số điểm dữ liệu có trong phần dữ liệu S, $|S_i|$ là số điểm dữ liệu có trong phần dữ liệu con S_i. Các giá trị entropy thấp sẽ có xu hướng được lựa chọn.

- Lợi thông tin (Information Gain): Lợi thông tin đo lường mức độ giảm độ không chắc chắn khi phân chia dữ liệu theo thuộc tính:

$$InformationGain = E(S) - Entropy - split(S \to S_1, ..., S_r) \tag{2.7}$$

Giá trị lợi thông tin lớn cho thấy thuộc tính đó là một lựa chọn tốt cho việc phân chia. Điều này có nghĩa là điều kiện rẽ nhánh tối ưu sẽ giảm thiểu entropy hoặc Gini index.

Điều kiện dừng của việc xây dựng cây quyết định có sự liên quan mật thiết đến chiến lược cắt tỉa cây. Cây quyết định có thể thể phát triển cho tới khi tất cả các nút lá chỉ bao gồm các điểm dữ liệu thuộc một lớp duy nhất, lúc này độ chính xác của mô hình sẽ đạt 100% trên tập huấn luyện. Tuy nhiên, kết quả của mô hình thường giảm mạnh khi kiểm thử trên tập kiểm thử độc lập, lúc này mô hình cây quyết định đã overfitting.

Có hai phương pháp phổ biến trong việc giảm overfitting cho cây quyết định, đó là Thiết lập các điều kiện dừng (Stopping Criteria) và Cắt tỉa cây (Pruning). Cây quyết định rất nhạy cảm với các thay đổi dù chỉ là nhỏ nhất ở trong dữ liệu, nên việc điều chỉnh các tham số điều kiện dừng không mấy khả quan. Cắt tỉa cây (Pruning) là phương phap phổ biến hơn cả trong việc giảm overfitting cho cây. Cụ thể, có các cach sau:

- Reduced Error Pruning: là một phương pháp cắt tỉa hậu (post-pruning) đơn giản và trực quan trong cây quyết định. Quy trình này bao gồm việc cắt bỏ dần các nhánh của cây và kiểm tra xem điều này có làm giảm tỷ lệ lỗi trên tập dữ liệu kiểm thử hay không. Nếu tỷ lệ lỗi không tăng, nhánh đó sẽ bị loại bỏ. Dữ liệu được chia ra làm tập huấn luyện và kiểm

thử, tập huấn luyện sẽ được sử dụng để xây dựng cây đầy đủ. Bắt đầu từ các nhánh lá (từ dưới lên trên), thử cắt bỏ nhánh và thay thế nó bằng một nút lá chứa nhãn phổ biến nhất trong nhánh đó, sau đó kiểm tra tỷ lệ lỗi trên tập kiểm thử. Nếu tỷ lệ lỗi không tăng hoặc giảm, giữ lại việc cắt tỉa (tức là nhánh đó bị loại bỏ). Nếu tỷ lệ lỗi tăng lên, hoàn tác việc cắt tỉa (giữ lại nhánh). Tiếp tục cắt tỉa cho đến khi không còn nhánh nào có thể bị loại bỏ mà không làm tăng tỷ lệ lỗi trên tập kiểm thử.

- Cost Complexity Pruning: là một phương pháp cắt tỉa cây quyết định dựa trên sự đánh đổi giữa lỗi dự đoán và độ phức tạp của cây. Bắt đầu với một cây quyết định hoàn chỉnh, nơi cây có thể phân chia tối đa các nhánh cho đến khi đạt được điều kiện dừng sau đó tính lỗi dự đoán R(T) trên tập kiểm thử (validation set) và chi phí tổng:

$$R_\alpha(T) = R(T) + \alpha|T| \tag{2.8}$$

trong đó $R(T)$ là lỗi dự đoán của cây, |T| là ố lượng nút trong cây, và α là tham số điều chỉnh giữa độ lỗi và độ phức tạp. Việc cắt tỉa bắt đầu từ các nhánh dưới cùng của cây (những nhánh gần lá), thử cắt tỉa các nhánh đó. Nếu việc cắt tỉa làm giảm chi phí tổng $R_\alpha(T)$, thì giữ lại việc cắt tỉa. Nếu chi phí tổng tăng, hoàn tác việc cắt tỉa. Tiếp tục cắt tỉa cho đến khi không còn nhánh nào có thể cắt mà không làm tăng chi phí tổng. Việc lựa chọn tham số α đòi hỏi phải có sự thử nghiệm để tìm giá trị phù hợp nhất.

2.3 Naive Bayes Classifier

Naive Bayes Classifier (Larose and Larose, 2019) là một thuật toán phân loại dựa trên Định lý Bayes với giả định rằng các thuộc tính trong dữ liệu là độc lập với nhau, nghĩa là không có mối quan hệ giữa các thuộc tính, đây là giả định ”naive”(ngây thơ). Định lý Bayes mô tả xác suất của một sự kiện xảy ra dựa trên thông tin đã biết về các sự kiện khác, công thức:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \tag{2.9}$$

Trong đó $P(A|B)$ là xác suất xảy ra sự kiện A khi biết B đã xảy ra, P(B|A) là xác suất của sự kiện B khi biết A đã xảy ra.

Xét bài toán classification với C lớp $1, 2, ..., C$. Giả sử có một điểm dữ liệu $X \in R^d$, bài toán đặt ra sẽ là tính xác suất điểm dữ liệu trên thuộc về lớp c hay tính $P(c|X)$. Định lý Bayes cho phép tính xác suất này như sau:

$$P(c|X) = \frac{P(X|c) * P(c)}{P(X)} \tag{2.10}$$

Vì $P(X)$ là hằng số không phụ thuộc vào lớp, ta có thể bỏ qua nó khi tối đa hóa. Từ đó, để xác định lớp c, ta tối đa hóa:

$$argmax_c P(X|c) * P(c) \tag{2.11}$$

tức là chọn lớp c sao cho xác suất $P(X|c) * P(c)$ là cao nhất.

$P(c)$ có thể được hiểu là xác suất để một điểm rơi vào lớp c tức tỉ lệ số điểm dữ liệu rơi vào lớp này chia cho tổng số lượng dữ liệu. Thành phần còn lại là $P(X|c)$ tức phân phối của các điểm dữ liệu trong lớp c, thường rất khó tính toán vì x là một biến ngẫu nhiên nhiều chiều, cần rất rất

nhiều dữ liệu để có thể xây dựng được phân phối đó. Để giúp cho việc tính toán được đơn giản, giả thiết một cách đơn giản nhất rằng các thành phần của biến ngẫu nhiên x là độc lập

$$P(X|c) = P(x_1, x_2, ..., x_d|c) = \prod_{i=1}^{d} P(x_i|c) \tag{2.12}$$

Giả thiết các chiều của dữ liệu độc lập với nhau là quá chặt và ít khi tìm được dữ liệu mà các thành phần hoàn toàn độc lập với nhau. Tuy nhiên, giả thiết "ngây ngô"này lại mang lại những kết quả tốt bất ngờ

2.4 Extreme Gradient Boosting

Thuật toán phân lớp XGBoost (Chen and Guestrin, 2016) là một thuật toán phân lớp dựa trên giải thuật cây tăng cường độ dốc (Gradient boosting tree), thuật toán phân lớp đã giành được nhiều kết quả cao trong nhiều bài toán khác nhau. XGBoost thuộc nhóm các thuật toán ensemble learning, cụ thể là phương pháp boosting. Thay vì tạo ra một mô hình mạnh ngay từ đầu, boosting xây dựng nhiều mô hình yếu (các cây quyết định nhỏ) theo cách tuần tự, mỗi cây mới được thêm vào nhằm giảm lỗi còn sót lại từ các cây trước đó.

Trong Extreme Gradient Boosting, tăng cường độ dốc là phương pháp mà ở mỗi bước, một mô hình mới (thường là một cây quyết định) được thêm vào để cải thiện dự đoán của mô hình trước đó. Mỗi mô hình mới được huấn luyện để giảm thiểu sai số của mô hình tổng thể. Bắt đầu với một dự đoán ban đầu (thường là giá trị trung bình cho các bài toán hồi quy).

$$f_0(x) = mean(y) \tag{2.13}$$

Tại mỗi bước t, mô hình mới sẽ được thêm vào. Để tìm ra mô hình tốt nhất để thêm vào, XGBoost sử dụng gradient và hessian (đạo hàm bậc nhất và bậc hai của hàm mất mát) để xác định hướng cải thiện.

$$g_i = \frac{\partial L(y_i, \hat{y_i})}{\partial \hat{y_i}} \tag{2.14}$$

$$h_i = \frac{\partial^2 L(y_i, \hat{y_i})}{\partial \hat{y_i}^2} \tag{2.15}$$

trong đó:

- $L(y_i, \hat{y_i})$: Hàm mất mát, là hàm số thể hiện sự khác biệt giữa giá trị thực tế y_i và giá trị dự đoán $\hat{y_i}$
- y_i Giá trị thực tế của mẫu thứ i
- $\hat{y_i}$ Giá trị dự đoán của mẫu thứ i

Gradient cho biết hướng điều chỉnh mà mô hình cần đi để giảm thiểu sai số. Hessian giúp xác định độ lớn của bước điều chỉnh (có thể xem như độ "nhạy cảm" của hàm mất mát).

Mô hình cây quyết định mới được huấn luyện để dự đoán giá trị của gradient (g) cho mỗi mẫu. XGBoost tối ưu hóa việc phân tách cây bằng cách sử dụng cả gradient và hessian để tính toán mức độ tối ưu hóa của mỗi nút trong cây. Dự đoán cuối cùng được cập nhật bằng cách thêm dự đoán từ cây mới vào mô hình hiện tại:

$$f_t(x) = f_{t-1}(x) + \alpha h_t(x) \tag{2.16}$$

trong đó

- $f_t(x)$ là mô hình sau khi thêm cây thứ t
- α là tốc độ học (learning rate)
- $h_t(x)$ là dự đoán từ cây thứ t

Quá trình này sẽ lặp lại cho đến khi đạt đến số lượng cây quyết định mong muốn hoặc khi cải thiện độ chính xác đạt đến một ngưỡng nhất định. Tất nhiên, việc tinh chỉnh các siêu tham số (hyper-parameter tuning) là cần thiết đề đạt được kết quả mong muốn, nhưng cũng đồng thời tránh mô hình bị overfiting.

2.5 Stacking Model

Stacking (Ting and Witten, 1999) là một trong những dạng của Ensemble Learing. Ensemble learning là một phương pháp trong học máy, nơi mà nhiều mô hình được kết hợp lại để tạo ra một mô hình tổng thể mạnh mẽ hơn so với các mô hình riêng lẻ. Phương pháp này dựa trên nguyên tắc rằng ”nhiều đầu tốt hơn một đầu”, tức là sự kết hợp của các mô hình khác nhau có thể dẫn đến độ chính xác cao hơn và khả năng tổng quát tốt hơn

Ensemble Learing gồm có ba dạng phổ biến: Bagging (Bootstrap Aggregating), Boosting, và Stacking

	Bagging	Boosting
Mục tiêu	Giảm thiểu phương sai	Giảm thiểu bias
Kỹ thuật	Lấy mẫu ngẫu nhiên với hoàn lại	Tăng trọng số cho các mẫu khó phân loại
Kết quả	Kết hợp trung bình	Kết hợp trọng số
Tính ổn định	Thường ổn định hơn	Có thể nhạy cảm hơn với nhiễu

Bảng 2.1: So sánh giữa Bagging và Boosting

Stacking cho nhiều mô hình dự đoán khác nhau được kết hợp để cải thiện hiệu suất so với sử dụng một mô hình duy nhất. Điểm đặc biệt của stacking là nó không chỉ kết hợp kết quả từ nhiều mô hình mà còn học cách kết hợp chúng một cách tối ưu thông qua một mô hình ”tổng hợp”(meta-model).

Các mô hình cơ sở (hay còn gọi là level-0 models) là những mô hình dự đoán độc lập. Chúng có thể là bất kỳ mô hình học máy nào, từ hồi quy tuyến tính, cây quyết định, đến các mô hình phức tạp hơn như Random Forest, Gradient Boosting, hay neural networks. Mỗi mô hình cơ sở sẽ học từ dữ liệu và đưa ra dự đoán riêng.

Sau khi các mô hình cơ sở tạo ra dự đoán, kết quả dự đoán của chúng sẽ được sử dụng làm đầu vào cho một mô hình khác gọi là meta-model (hay level-1 model). Meta-model học cách kết hợp các dự đoán từ các mô hình cơ sở để đưa ra dự đoán cuối cùng chính xác hơn. Mô hình meta thường là các mô hình đơn giản như hồi quy tuyến tính, logistic regression.

Quá trình huấn luyện và dự đoán trong stacking thường được tổ chức thành hai giai đoạn:

- Huấn luyện các mô hình cơ sở: Dữ liệu ban đầu được chia thành tập huấn luyện và tập kiểm tra. Các mô hình cơ sở được huấn luyện trên tập huấn luyện, sau đó sử dụng tập kiểm tra để tạo ra các dự đoán (các dự đoán này là dữ liệu huấn luyện cho meta-model).

- Huấn luyện meta-model: Từ các dự đoán của các mô hình cơ sở trên tập kiểm tra, dữ liệu mới được tạo ra để huấn luyện meta-model. Khi mô hình meta học xong cách kết hợp các dự đoán từ các mô hình cơ sở, nó sẽ đưa ra dự đoán cuối cùng.

Stacking có khả năng giảm sai số tổng thể tốt (phương sai và bias), từ đó cải thiện hiệu suất cho mô hình (so sánh với Bagging và Boosting ở Bảng 2.1). Nhưng đánh đổi lại, đó chính là quá trình xây dựng thiết kế mô hình và khả năng tính toán yêu cầu cao.

2.6 Soft Voting

Soft voting (Zhou, 2012) là một trong hai phương pháp kết hợp dự đoán trong học máy, phương pháp còn lại là hard voting. Trong soft voting, các mô hình không chỉ đưa ra dự đoán cuối cùng mà còn cung cấp xác suất cho từng lớp (class) dự đoán. Từ những xác suất này, soft voting sẽ tính toán xác suất tổng thể cho từng lớp và chọn lớp có xác suất cao nhất làm dự đoán cuối cùng.

Trong soft voting, mỗi mô hình sẽ dự đoán xác suất cho từng lớp, thay vì chỉ đưa ra dự đoán thuộc về lớp nào với các bước chính sau:

- Dự đoán xác suất: Mỗi mô hình sẽ đưa ra dự đoán xác suất cho từng lớp.
- Tính toán xác suất trung bình: Xác suất cho từng lớp từ các mô hình sẽ được tính trung bình
- Chọn lớp có xác suất cao nhất: Lớp với xác suất cao nhất sẽ được chọn làm dự đoán cuối cùng.

Công thức cho Soft Voting như sau:

$$P(c) = \frac{1}{N} \sum_{i=1}^{N} P_i(c) \tag{2.17}$$

Trong đó N là số lượng mô hình, $P_i(c)$ là xác suất mà mô hình i dự đoán đối tượng thuộc về lớp c. Dự đoán cuối cùng sẽ thuộc về lớp có xác suất trung bình là lớn nhất, hay:

$$PredictedClass = argmax_c P(c) \tag{2.18}$$

Trong bài toán này, một phương pháp khác của Soft Voting đó là Weighted Soft Voting sẽ được sử dụng. Weighted Soft Voting sẽ gán thêm trọng số cho mỗi một mô hình dự đoán, một mô hình có độ tin cậy và chính xác cao thì sẽ được ưu tiên mang trọng số lớn, và ngược lại, một mô hình có độ tin cậy và chính xác không cao thì sẽ mang trọng số nhỏ hơn

Giả sử ta có N mô hình dự đoán và các trọng số tương ứng w_1, w_2, ..., w_N. Đối với một mẫu dữ liệu cụ thể, xác suất dự đoán của mô hình thứ i cho lớp C_k là $P_i(C_k)$. Khi đó, xác suất có trọng số cho lớp C_k sẽ được tính bằng

$$P(C_k) = \frac{\sum_{i=1}^{N} (w_i \cdot P_i(C_k))}{\sum_{i=1}^{N} w_i} \tag{2.19}$$

trong đó $P(c)$ là xác suất tổng thể cho lớp c, K là số lượng mô hình, $P_i(c)$ là xác suất mà mô hình thứ i dự đoán cho lớp c.

Tổng kết chương:
Chương 2 của báo cáo đã trình bày một cách ngắn gọn và đầy đủ về cơ sở lý thuyết các mô hình thuật toán được sử dụng cho bài toán phân lớp được áp dụng. Tiếp theo tại chương 3, báo cáo sẽ phân tích bộ dữ liệu, cách tiền xử lí dữ liệu và xây dựng mô hình cho bài toán

Chương 3

Mô hình giải quyết bài toán đề xuất

3.1 Bộ dữ liệu

Đây là bộ dữ liệu nguồn mở (open-source) (Larose and Larose, 2014) gồm 21 đặc trưng (feature) và 3333 mẫu (observations). Đặc trưng 'Churn' cho biết khách hàng có rời bỏ dịch vụ hay không dựa trên các điều kiện hiện có. Khoảng 14.5% của nhãn 'Churn' được đánh dấu là 'T' (True) biểu thị việc khách hàng rời bỏ, trong khi 84.5% nhãn là 'F' (False) biểu thị khách hàng không rời bỏ. Trong nghiên cứu này, 80% dữ liệu (2666 mẫu) được sử dụng cho tập huấn luyện và 20% dữ liệu (667 mẫu) được sử dụng cho tập kiểm tra. Các đặc trưng và mô tả được thể hiện ở Bảng 3.1

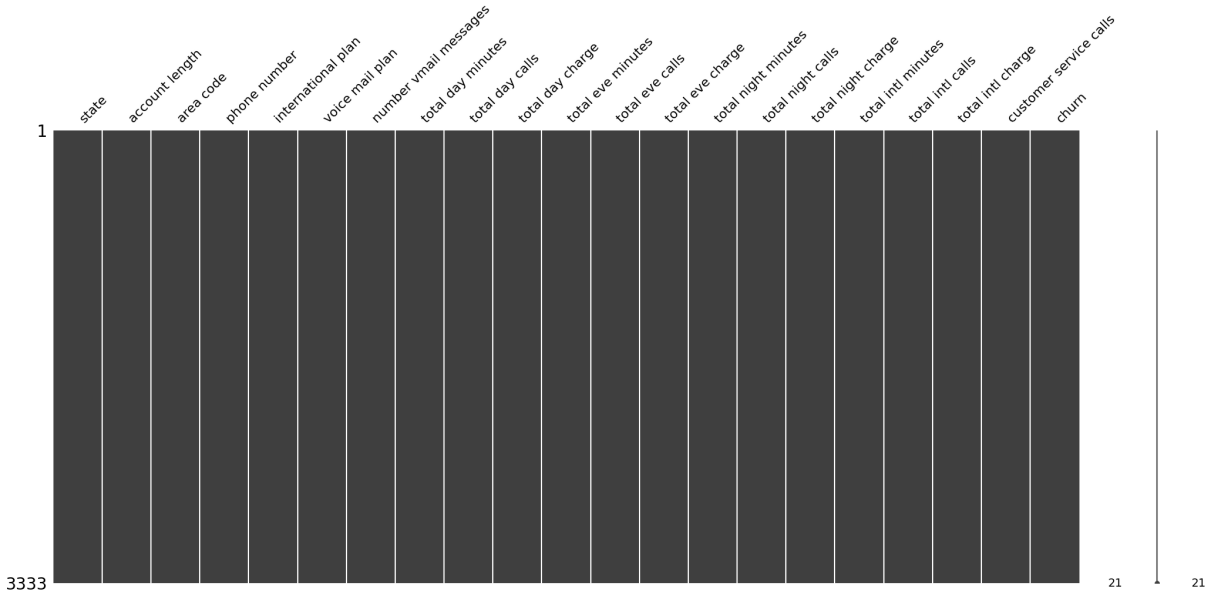
Tên đặc trưng	Mô tả	Kiểu dữ liệu
State	Bang khách hàng cư trú	object
Account length	Số ngày khách hàng sử dụng dịch vụ	int64
Area code	Mã vùng	object
Phone number	Số điện thoại	object
International Plan	Dịch vụ liên lạc quốc tế	bool
Voice mail plan	Dịch vụ voice mail	bool
Number Vmail message	Số lượng voice mail	int64
Total day minutes	Số phút gọi ban ngày	float64
Total day calls	Số cuộc gọi ban ngày	int64
Total day charge	Số tiền phải trả ban ngày	int64
Total eve minutes	Số phút gọi trong buổi tối	float64
Total eve calls	Số cuộc gọi trong buổi tối	int64
Total eve charge	Số tiền phải trả trong buổi tối	float64
Total night minutes	Số phút gọi trong buổi đêm	float64
Total night calls	Số cuộc gọi trong buôi đêm	int64
Total night charge	Số tiền phải trả trong buổi đêm	float64
Total intl minutes	Số phút gọi quốc tế	float64
Total intl calls	Số cuộc gọi quốc tế	int64
Total intl charge	Số tiền phải trả cho dịch vụ gọi quốc tế	float64
Customer service calls	Số cuộc gọi chăm sóc khách hàng	int64
Churn	Người dùng có hủy dịch vụ không	bool

Bảng 3.1: Tổng quan bộ dữ liệu

3.2 Phân tích khai phá dữ liệu

3.2.1 Gía trị mất (Missing Value)

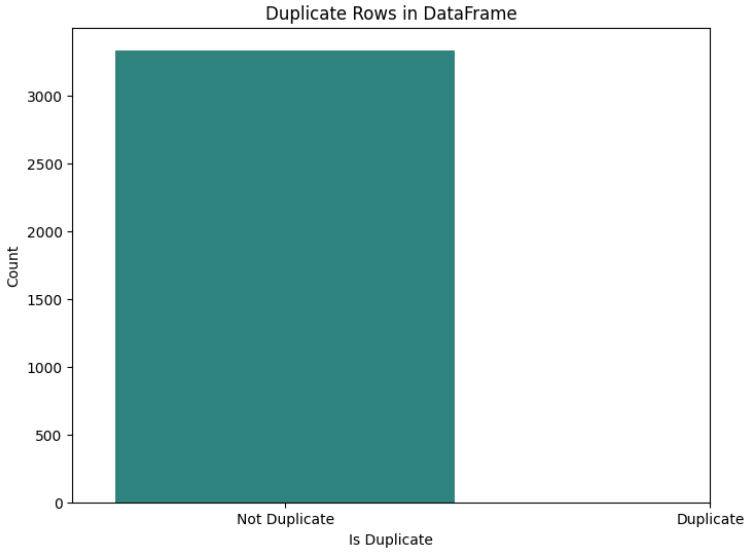
Giá trị mất (missing values) trong bộ dữ liệu là những giá trị không có sẵn hoặc không được ghi nhận trong quá trình thu thập dữ liệu. Giá trị mất có thể làm cho các dự đoán của mô hình trở nên sai lệch, đặc biệt nếu các giá trị mất có mối quan hệ với nhãn mục tiêu. Bộ dữ liệu không tồn tại giá trị mất (missing value), Hình 3.1



Ảnh 3.1: Missing Value

3.2.2 Gía trị lặp (Duplication)

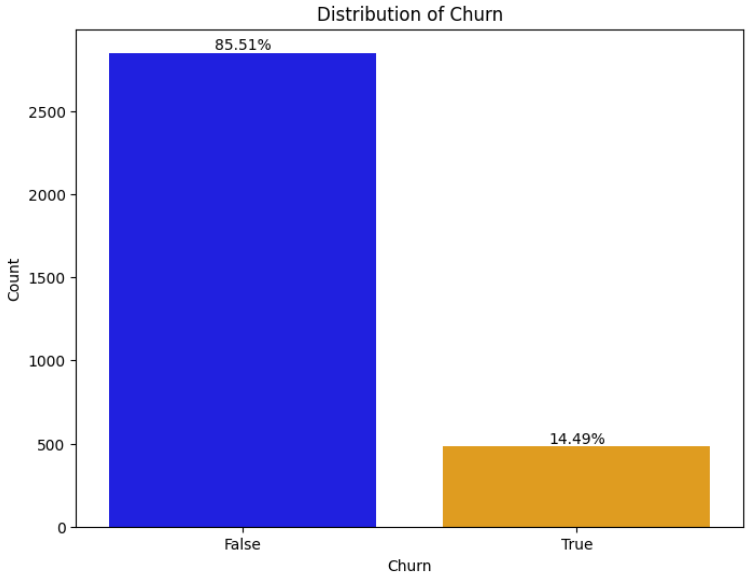
Các mẫu lặp nhau trong bộ dữ liệu, hay còn gọi là giá trị trùng lặp (duplicate values), là những bản ghi có cùng các giá trị trong tất cả các cột hoặc trong một số cột nhất định. Nhiều bản ghi trùng lặp trong tập dữ liệu khiến mô hình học máy có thể bị thiên lệch vì nó coi các bản ghi này như các điểm dữ liệu độc lập, dẫn đến việc tạo ra các dự đoán không chính xác. Các bản ghi trùng lặp có thể làm tăng kích thước của tập dữ liệu, dẫn đến việc xử lý chậm hơn trong các mô hình học máy. Các phép đo như trung bình, phương sai, và các thống kê mô tả khác có thể bị sai lệch do sự hiện diện của các giá trị trùng lặp. Bộ dữ liệu không tồn lại các mẫu lặp nhau, Hình 3.2



Ảnh 3.2: Duplication

3.2.3 Mất cân bằng dữ liệu (imbalance data) và lấy lại mẫu dữ liệu (resampling)

Mất cân bằng dữ liệu giữa các nhãn được coi là một trong những vấn đề phổ biến trong các bài toán học máy phân lớp [21], vấn đề xảy ra khi số lượng bản ghi của một nhãn quá nhỏ so với số lượng bản ghi của các nhãn còn lại. Việc này khiến cho các thuật toán phân lớp phân lớp có xu hướng bỏ qua các nhãn có số lượng bản ghi thấp và tập trung vào các nhãn có số lượng bản ghi lớn hơn. Một trong những phương pháp được sử dụng để xử lý vấn đề mất cân bằng dữ liệu giữa các nhãn là phương pháp lấy mẫu lại dữ liệu với mục đích cân bằng số bản ghi giữa các nhãn trên tập huấn luyện trước khi tiến hành huấn luyện thuật toán phân lớp. Trong bộ dữ liệu trên, việc mất cân bằng dữ liệu được thể hiện rất rõ ràng qua Hình 3.3, với 85.51% nhãn ’F’ và 14.49% nhãn ’True’. SMOTE (Synthetic Minority Oversampling Technique) sẽ là phương pháp được sử dụng để khắc phục mất cân bằng dữ liệu trong bài trên. SMOTE là phương pháp lấy lại mẫu dữ liệu thông qua việc tạo ra các điểm dữ liệu tổng hợp cho các nhãn thiểu số từ đó khắc phục vấn đề mất cân bằng dữ liệu giữa các nhãn trong tập dữ liệu.



Ảnh 3.3: Phân bố của nhãn

3.2.4 Điểm ngoại lai (Ouliers)

Điểm ngoại lai là điểm dữ liệu có sự sai lệch rất nhiều so với các điểm dữ liệu khác có trong bộ dữ liệu. Việc xác định các điểm ngoại lại là việc quan trọng do các điểm ngoại lệ có thể là biểu hiện của dữ liệu xấu. Có thể do đặc trưng của điểm dữ liệu có thể bị mã hóa không chính xác hoặc một tiến trình ghi dữ liệu được thực hiện sai và tạo ra một giá trị quá lớn hoặc quá nhỏ tại một đặc trưng của điểm dữ liệu. Việc xuất hiện các giá trị ngoại lai này trong quá trình huấn luyện có thể làm sai lệch quá trình huấn luyện và có thể khiến thuật toán phân lớp thể hiện kém hơn trong quá trình kiểm thử và ứng dụng thuật toán phân lớp trong thực tế. Những cũng có những trường hợp, các điểm ngoại lai không phải là một điểm dữ liệu lỗi trong bộ dữ liệu, mà đơn thuần, điểm ngoại lai này là phân bổ tự nhiên của dữ liệu (natural variation of data). Tức là, các điểm dữ liệu này biểu thị cho những trường hợp đặc biệt, hiếm có xuất hiện trong phân bổ dữ liệu, và nó vẫn mang những thông tin giá trị trong bộ dữ liệu

Dải phân vị (Interquartile Range hay IQR), là một phương pháp phổ biến để xác định điểm ngoại lai. Dải phân vị của một đặc trưng là khoảng nằm giữa hai giá trị Q1 và Q3 của đặc trưng đó, Hình 3.4, hay:

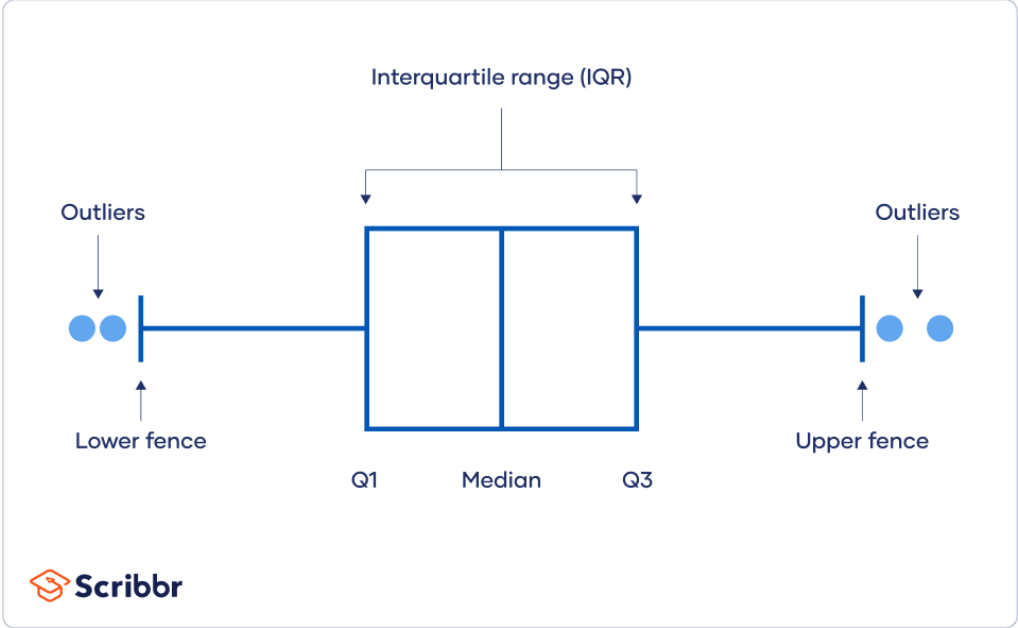
$$IQR = Q3 - Q1 \tag{3.1}$$

từ đó xác định hai ngưỡng T_{max} và T_{min} theo công thức:

$$T_{max} = Q3 + 1.5 * IQR \tag{3.2}$$

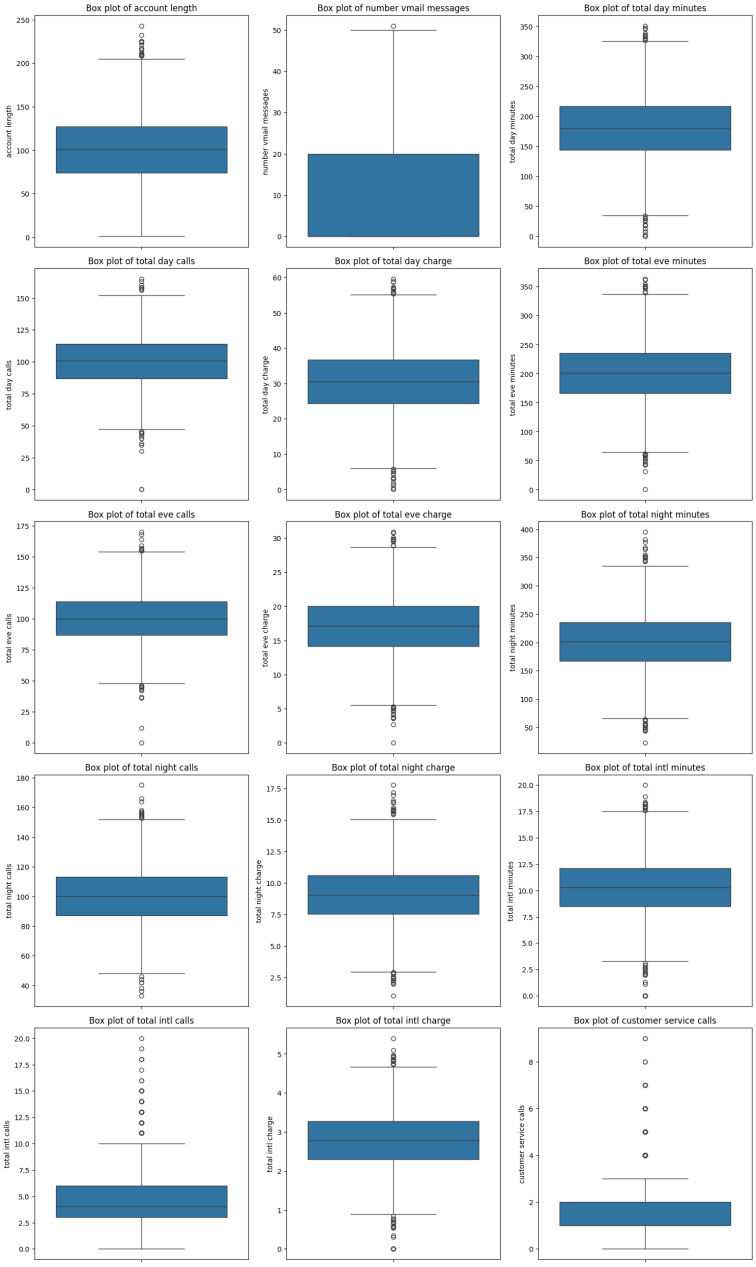
$$T_{min} = Q1 - 1.5 * IQR \tag{3.3}$$

Các điểm không thuộc khoảng $[T_{min}, T_{max}]$ thì được xác định là điểm ngoại lai của đặc trưng.



Ảnh 3.4: Dải phân vị (Nguồn: Scribbr)

Biểu đồ boxplot cho các đặc trưng mang giá trị số (numerical), hình 3.5



Ảnh 3.5: Biểu đồ boxplot các đặc trưng mang giá trị số

3.2.5 Lựa chọn siêu tham số (hyper-parameter) cho các mô hình phân loại

Siêu tham số (hyper-parameters) là các tham số trong mô hình học máy được thiết lập trước khi quá trình huấn luyện bắt đầu, nó điều khiển hành vi của thuật toán học máy. Siêu tham số được định nghĩa và điều chỉnh thủ công. Lựa chọn siêu tham số (hyper-parameter tuning) là một bước quan trọng trong quá trình huấn luyện mô hình học máy, việc lựa chọn và tinh chỉnh các siêu tham số sẽ ảnh hưởng lớn đến hiệu suất của mô hình.

Các thuật toán phân lớp: Các thuật toán phân lớp được sử dụng từ thư viện Scikit-learn[1], một thư viện rất nổi tiếng với cộng đồng Học máy nói chung và cộng đồng Học máy bằng Python nói

[1] Scikit-learn

riêng. Cùng với đó là thư viện XGBoost[2], thư viện được xây dựng dành riêng cho mô hình thuật toán XGBoost. Đầu vào của thuật toán là bộ dữ liệu sau giai đoạn tiền xử lí dữ liệu, được biểu diễn gồm các vector, mỗi vector là một mẫu, mỗi chiều của vector là thể hiện mỗi đặc trưng của mẫu. Đầu ra của thuật toán là giá trị *0* hoặc *1*, với *0* là tương ứng với việc khách hàng không có khả năng ngừng sử dụng dịch vụ, và *1* là khách hàng có khả năng ngừng sử dụng dịch vụ.

Việc lựa chọn các siêu tham số được thực hiện qua thuật toán Tìm kiếm trên lưới (Grid Search), được triển khai với hàm có sẵn trong thư việc Scikit-learn, GridSearchCV.

- Logistic Regression: Các siêu tham số điều chỉnh

    1. *C* (Tham số điều chuẩn): Siêu tham số này điều chỉnh cường độ của regularization (phép điều chuẩn) trong Logistic Regression. *C* là nghịch đảo của cường độ regularization. Nếu *C* nhỏ, mô hình sẽ áp dụng regularization mạnh (penalize large coefficients nhiều hơn); nếu *C* lớn, regularization sẽ yếu hơn.
    2. *penalty* (Loại điều chuẩn): Siêu tham số này xác định loại penalty (hình phạt regularization) được áp dụng. Logistic Regression thường có các hình phạt L1 và L2, hoặc có thể kết hợp cả hai. L1 sẽ có xu hương loại bỏ hoàn toàn các đặc trưng ít quan trọng bằng cách đưa trọng số của đặc trưng về 0, còn L2 chỉ làm giảm giá trị của chúng.
    3. *solver* (Thuật toán tối ưu hóa): Siêu tham số này chỉ định thuật toán tối ưu được sử dụng để tối thiểu hóa hàm mục tiêu trong Logistic Regression.
    4. *max_iter* (Số lượng vòng lặp tối đa): Tham số này quy định số lượng vòng lặp tối đa mà thuật toán tối ưu hóa sẽ chạy trước khi dừng lại. Tham số này sẽ đóng vai trò quan trọng trong việc giúp thuật toán hội tụ (convergence), hay chính là hàm mất mát đạt giá trị tối ưu.

- Decision Tree: Các tham số điều chỉnh

    1. *criterion* (Tiêu chí phân chia): Siêu tham số này quyết định tiêu chí nào được sử dụng để đánh giá sự phân chia tại mỗi nút trong cây quyết định. *gini* thường nhanh hơn về mặt tính toán do công thức đơn giản hơn, nhưng cả *gini* và *entropy* đều cho kết quả tương tự trong nhiều trường hợp.
    2. *max_depth* (Độ sâu tối đa của cây): Siêu tham số này xác định độ sâu tối đa mà cây có thể đạt được. Độ sâu càng lớn, cây càng có nhiều khả năng ghi nhớ chi tiết, từ đó đưa ra dự đoán chính xác, nhưng có thể dẫn đến hiện tượng overfitting.
    3. *min_sample_leaf* (Số mẫu tối thiểu tại một nút lá): Siêu tham số quy định số lượng mẫu tối thiểu phải có tại mỗi nút lá của cây. Nút lá là nơi mà cây dừng lại và đưa ra dự đoán cuối cùng cho một phần của dữ liệu. Siêu tham số này giúp kiểm soát kích thước tối thiểu của các nút nhằm tránh các nút lá quá nhỏ, chứa quá ít mẫu.
    4. *min_samples_split* (Số mẫu tối thiểu để tách một nút): Siêu tham số xác định số lượng mẫu tối thiểu cần có tại một nút để thực hiện phép tách (split), ngăn chặn việc phân chia các nút mà không có đủ dữ liệu, điều này có thể dẫn đến việc tạo ra các nút không có đủ thông tin để học hỏi từ đó. Nếu một nút có ít mẫu, việc chia nó có thể không mang lại thông tin bổ sung có giá trị.

- XGBoost: Các tham số cần điều chỉnh

[2] XGBoost

1. *learning_rate* (Tốc độ học): Siêu tham số này quyết định đến lượng thông tin mà mỗi cây mới đóng góp vào dự đoán cuối cùng của mô hình kiểm soát tốc độ hội tụ của thuật toán tối ưu hóa.
2. *n_estimators*: Số lượng cây quyết định tồn tại trong mô hình.
3. *colsample_bytree*: Tỉ lệ các đặc trưng (feature) sẽ được sử dụng để xây dựng một cây quyết định thành phần, giới hạn số lượng đặc trưng mà mỗi cây có thể sử dụng.
4. *max_depth*: Độ sâu tối đa của mỗi cây quyết định thành phần. Độ sâu càng lớn, cây càng có nhiều khả năng ghi nhớ chi tiết, từ đó đưa ra dự đoán chính xác, nhưng có thể dẫn đến hiện tượng overfitting.
5. *subsample*: Tỉ lệ các mẫu (sample) sẽ được sử dụng để xây dựng một cây quyết định thành phần, giới hạn số lượng mẫu mà mỗi cây được sử dụng.
6. *eval_metric*: Hàm mất mát (loss function) được sử dụng để đánh giá hiệu suất mô hình.

Cụ thể các siêu tham số được tinh chỉnh thể hiện ở Bảng 3.2

Mô hình thuật toán	Các siêu tham số
Logistic Regression	*C*: [0.001, 0.01, 0.1, 1, 10] *penalty*: ['l1', 'l2'] *solver*: ['liblinear', 'newton-cholesky'] *max_iter*: [100, 200, 500, 1000, 5000]
Decision Tree	*max_depth*: [5, 10, 15, 20, 25, 30] *min_samples_split*: [2, 5, 10] *min_samples_leaf*: [1, 2, 5, 7, 9] *criterion*: ['gini', 'entropy']
XGBoost	*n_estimator*: [100, 200, 300] *max_depth*: [3, 5, 7] *learning_rate*: [0.001, 0.01, 0.1] *subsample*: [0.8, 1.0] *colsample_bytree*: [0.8, 1.0]

Bảng 3.2: Các siêu tham số cho mỗi mô hình

3.3 Phương pháp tiền xử lý dữ liệu

3.3.1 Xử lí Giá trị mất (Missing value) và Mẫu lặp (Duplication)

Như đã đề cập tại Mục 3.2, và Hình 3.1 3.2, bộ dữ liệu không có giá trị mất (missing value) và mẫu lặp (duplication), nên bước tiền xử lí này sẽ được bỏ qua.

3.3.2 Mã hóa dữ liệu (Data Encoding)

Mã hóa dữ liệu là việc chuyển tất cả dữ liệu về kiểu mà máy có thể hiểu được để có thể thực hiện quá trình tính toán. Hai phương pháp mã hóa dữ liệu được sử dụng sẽ là One-hot Encoding và Label Encoding

- One-hot Encoding: là một kỹ thuật được sử dụng để chuyển đổi giá trị biến phân loại (categorical variables) thành một dải giá trị nhị phân. Gỉa sử với bộ dữ liệu có *n* mẫu, tại đặc trưng *i* có *k* giá trị duy nhất (unique values), One-hot Encoding sẽ tạo thêm *k* đặc trưng mới tương ứng với *k* giá trị duy nhất, mỗi đặc trưng có dịnh danh danh 'is_value' (với value là một giá trị duy nhất trong *k* giá trị), các đặc trưng này sẽ có kiểu dữ liệu 'bool'. Mẫu tại đặc trưng *i* mang giá trị 'value', thì tại đặc trưng 'is_value' sẽ mang giá trị *1*, còn lại sẽ là *0*. Trong bộ dữ liệu nầy, One-hot Encoding sẽ được áp dụng với hai đặc trưng, Bảng 3.3

Đặc trưng	Số giá trị duy nhất
state	51
area code	3

Bảng 3.3: Các đặc trưng sử dụng One-hot Encoding

- Label Encoding: là một kỹ thuật được sử dụng để chuyển đổi giá trị biến phân loại (categorical variables) thành một giá trị só nguyên (integer). Cụ thể, mỗi giá trị duy nhất (unique values) sẽ được chuyển thành một số nguyên, bắt đầu từ giá trị 0. Trong bộ dữ liệu nầy, Label Encoding sẽ được áp dụng với hai đặc trưng, Bảng 3.4

Đặc trưng	Số giá trị duy nhất
international plan	2
voice mail plan	2

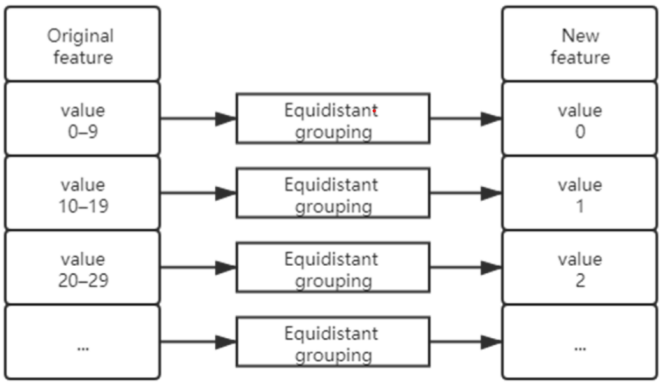
Bảng 3.4: Các đặc trưng sử dụng Label Encoding

3.3.3 Tạo đặc trưng mới (Features Construction)

Tạo đặc trưng mới (Feature Construction) là quá trình cải thiện dữ liệu bằng cách tạo ra các đặc trưng mới từ các đặc trưng đã có. Trong bộ dữ liệu, có thể các đặc trưng có sẵn chưa thể hiện được hết tất cả những thông tin ẩn có giá trị mà bộ dữ liệu mang. Việc tạo đặc trưng mới sẽ khai thác được lượng thôn tin ẩn đó, làm giàu dữ liệu và cải thiện khả năng dự đoán của mô hình học máy (Domingos, 2012).

Trong bộ dữ liệu này, phương pháp Equidistant Grouping (Phân nhóm đều khoảng cách) sẽ được sử dụng. Equidistant Grouping là một dạng của Rời rạc hóa các giá trị Liên tục (Discretization of Continuous Variables)[3], minh họa tại Hình 3.6

[3]Chuyển các biến mang giá trị liên tục (continuous) về dạng rời rạc (discrete)



Ảnh 3.6: Ví dụ về Equidistant Grouping

Việc sử dụng phương pháp này như đang ngầm chia khách hàng vào các nhóm có hành vi sử dụng dịch vụ khác nhau. Các khách hàng có cùng hành vi sử dụng dịch vụ, có khả năng sẽ có cùng một quyết định (churn hay not churn) (García-Torres et al., 2015). Số lượng các nhóm sẽ được chia theo công thức Sturges[4] (Sturges, 1926), công thức như sau:

$$K = 1 + log_2(n) \tag{3.4}$$

trong đó K là số nhóm, n là khoảng giá trị (value range) của đặc trưng, thường lấy giá trị lớn nhất (max) trừ đi giá trị nhỏ nhất (min). Mỗi nhóm sẽ có một khoảng giá trị xác định, đặc trưng mang giá trị thuộc khoảng giá trị của nhóm nào thì sẽ được đánh số của nhóm đó. Trong bộ dữ liệu này, sẽ có 12 đặc trưng được sử dụng để thực hiện quá trình nói trên, Bảng 3.5

Đặc trưng	Khoảng giá trị
Number Vmail message	0 - 51
Total day minutes	0 - 350.8
Total day calls	0 - 165
Total day charge	0 - 59.64
Total eve minutes	0 - 363.7
Total eve calls	0 - 170
Total eve charge	0 - 30.91
Total night minutes	23.2 - 295.0
Total night calls	33 - 175
Total night charge	1.04 - 17.77
Total intl minutes	0 - 20.0
Total intl calls	0 - 20

Bảng 3.5: Các đặc trưng sử dụng Equidistant Grouping

Tổng kết chương:
Chương 3 của báo cáo đã cho cái nhìn tổng quan về bộ dữ liệu, các vấn đề tồn tại trong bộ dữ liệu, các phương pháp tiền xử lí dữ liệu, cùng với đó là giải thích chỉ tiết về các mô hình thuật toán sẽ được sử dụng. Chương 4 sẽ đưa ra kết quả sau hoàn tất quá trình thực nghiệm bài toán.

[4]Công thức được sử dụng để tìm số lượng nhóm phù hợp trong bảng phân phối tần suất

Chương 4

Thực nghiệm và Kết quả

4.1 Quá trình tiền xử lí dữ liệu

Sau quá trình tiền xử lí dữ liệu, các đặc trưng của dữ liệu đã được đưa hết về dạng số (numerical). Bộ dữ liệu trước khi và sau khi Equidistant Grouping (mục 3.3.3) sẽ đều được sử dụng cho quá trình huấn luyện mô hình. Cụ thể:

- Bộ dữ liệu trước khi Equidistant Grouping (kí hiệu: ODS - Original Dataset): Bộ dữ liệu sẽ bao gồm 3333 mẫu (sample) và 72 đặc trưng (feature). Các đặc trưng 'international plan' và 'voice mail plan' đã được đưa về dạng số theo kiểu nhị phân (0 và 1). Các đặc trưng 'state' và 'area code' sau quá trình One-hot Encoding tạo ra thêm lần lượt 51 và 3 đặc trưng mới. Đặc trưng 'phone number' được coi như là một dạng ID (định danh) của khách hành, do số điện thoại của mỗi khách hàng không thể trùng nhau, đặc trưng này không mang giá trị thông tin giúp cho việc dự đoán của mô hình, vậy nên sẽ bị loại bỏ.
- Bộ dữ liệu sau khi Equidistant Grouping (kí hiệu: NDS - New Dataset): Bộ dữ liệu sẽ bao gồm 3333 mẫu (sample) và 84 đặc trưng (feature). Ngoài các đặc trưng tương tự như đã mô tả tại bộ dữ liệu trước khi Equidistant Grouping, bộ dữ liệu sẽ có thêm 12 đặc trưng mới bởi quá trình Equidistant Grouping, Bảng 4.1: Các đặc trưng mới sẽ có định danh 'tên đặc trưng + group' (ví dụ: total day calls group)

Feature	Value	K
Number vmail messages	0–51	6
Total day minutes	0–350.8	10
Total day calls	0–165	8
Total day charge	0–59.64	6
Total night minutes	0–363.7	10
Total night calls	0–170	8
Total night charge	0–30.91	7
Total eve minutes	0–395	10
Total eve calls	0–175	8
Total eve charge	0–17.77	5
Total intl minutes	0–20	5
Total intl calls	0–20	5

Bảng 4.1: Đặc trưng và số lượng nhóm tương ứng

4.2 Lựa chọn siêu tham số (hyper-parameter cho các mô hình phân loại)

Như đã đề cập tại mục 3.2.5 và Bảng 3.2, Grid Search sẽ là phương pháp được sử dụng để tìm ra tham số tối ưu nhất, kết quả được thể hiện tại Bảng 4.2

Mô hình thuật toán	**Bộ dữ liệu**	Tham số tối ưu
Logistic Regeression	ODS	'C': 1 'max_iter': 100 'penalty': 'l1' 'solver': 'liblinear'
	NDS	'C': 0.1 'max_iter': 100 'penalty': 'l2' 'solver': 'newton-cholesky'
Decision Tree	ODS	'criterion': 'gini' 'max_depth': 5 'min_samples_leaf': 7 'min_samples_split': 2
	NDS	'criterion': 'gini' 'max_depth': 5 'min_samples_leaf': 7 'min_samples_split': 2
XGBoost	ODS	colsample_bytree': 0.8 'learning_rate': 0.1 'max_depth': 7 'n_estimators': 200 'subsample': 0.8
	NDS	'colsample_bytree': 1 'learning_rate': 0.1 'max_depth': 7 'n_estimators': 100 'subsample': 1

Bảng 4.2: Các mô hình thuật toán và siêu tham số tối ưu

4.3 Phương pháp đánh giá (Evaluation Metric)

Accuracy là phương pháp đánh giá hiệu suất mô hình được sử dụng phổ biến trong các bài toán phân loại. Nó phản ảnh tỉ lệ phần trăm số lượng mẫu được phân loại đùng, so với tổng sô mẫu tồn tại trong tập dữ liệu. Accuracy được tính bằng công thức:

$$\text{Accuracy} = \frac{\text{Số lượng dự đoán đúng}}{\text{Tổng số mẫu của tập dữ liệu}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4.1}$$

trong đó:

- TP (True Positive): Số lượng dự đoán đúng với mẫu thực là dương (Dương tính thực)
- TN (True Negative): Số lượng dự đoán đúng với mẫu thực là âm (Âm tính thực)

- FP (False Positive): Số lượng dự đoán sai với mẫu thực là dương (Dương tính giả)
- FN (False Negative): Số lượng dự đoán sai với vỡi thực là âm (Âm tính giả)

4.4 Kết quả huấn luyện mô hình

Mô hình thuật toán	ODS		NDS	
	Thực nghiệm	Bài báo	Thực nghiệm	Bài báo
LR	Accuracy: 0.8391 ROC: 0.8283	Accuracy: 0.8482	Accuracy: 0.8526 ROC: 0.8383	Accuracy: 0.8586
DT	Accuracy: 0.8430 ROC: 0.8459	Accuracy: 0.8513	Accuracy: 0.8589 ROC: 0.8508	Accuracy: 0.8605
NBC	Accuracy: 0.8711 ROC: 0.8413	Accuracy: 0.8514	Accuracy: 0.8621 ROC: 0.8519	Accuracy: 0.8575
XGB	Accuracy: 0.9565 ROC: 0.8768	Accuracy: 0.9543	Accuracy: 0.9550 ROC: 0.9001	Accuracy: 0.9554
XGB + LR	Accuracy: 0.9550 ROC: 0.9166	Accuracy: 0.9463	Accuracy: 0.9505 ROC: 0.9057	Accuracy: 0.9585
XGB + DT	Accuracy: 0.9506 ROC: 0.9171	Accuracy: 0.9493	Accuracy: 0.9415 ROC: 0.8964	Accuracy: 0.9560
XGB + NBC	Accuracy: 0.9525 ROC: 0.9003	Accuracy: 0.9471	Accuracy: 0.9490 ROC: 0.9008	Accuracy: 0.9535
WSV	Accuracy: 0.9602 ROC: 0.9183	Accuracy: 0.9612	Accuracy: 0.9702 ROC: 0.9058	Accuracy: 0.9809

Bảng 4.3: So sánh hiệu suất của các mô hình trên ODS và NDS

4.5 Nhận xét

1. Kết quả đầu ra: Kết quả đầu ra của quá trình thực nghiệm gần như đạt được kết quả của bài báo (còn có sự chênh lệnh hơn kém khoảng một vài phần trăm). Vậy nên, quá trình thực nghiệm đã hoàn thành được tốt nhiệm vụ.

2. Phương pháp đánh giá: Chỉ số đánh giá chính của bài báo sử dụng là Accuracy (như đã trình bày tại 4.3). Phương pháp đánh giá này có thể sẽ không phản ánh được chính xác độ hiệu quả của mô hình, với một bộ dữ liệu có hiện tượng mất cân bằng, thể hiện tại hình 3.3. Việc tinh chỉnh các siêu tham số như đã đề cập tại 3.2.5 có thể phần nào đã xử lí được việc mất cân bằng dữ liệu, song có thể vẫn chưa triệt để. Phần thực nghiệm có đưa ra thêm một chỉ số đánh giá khác, đó là ROC[1] - một chỉ số đánh giá phù hợp cho những bài toán có bộ dữ liệu bị mất cân bằng, kết quả thể hiện tại bảng 4.3 cho thấy hiệu suất của mô hình vẫn ở mức tốt, nhưng không đạt được ngưỡng như Accuracy (độ chênh lệch khoảng 1% đến 7% tùy mô hình khác nhau).

3. Hướng đi Model-Centric của bài báo: Hướng đi Model-Centric của bài báo đã cho ra một kết quả khá tốt. Song, đối với một bộ dữ liệu không quá lớn như trên, lượng thông tin và tri thức

[1] (Receiver Operating Characteristic) là một đồ thị thể hiện hiệu suất của một mô hình phân loại nhị phân khi ngưỡng phân loại thay đổi. ROC giúp đánh giá khả năng phân biệt giữa hai lớp của mô hình

tiềm ẩn trong dữ liệu không nhiều, việc có thể chuyển hướng tiếp cận sang Data-Centric có thể sẽ còn cho ra một kết quả tốt hơn nữa.

4.6 So sánh với các phương pháp khác

1. Các mô hình độc lập (Sabbeh, 2018)

Tiêu chí	**Bài báo 'Machine-Learning Techniques for Customer Retention: A Comparative Study'**	**Bài báo này**
Thuật toán chính	Các mô hình đơn lẻ: Logistic Regression, Decision Tree, Naive Bayes, Random Forest, và Neural Networks. Mỗi mô hình được huấn luyện độc lập và so sánh độ chính xác với nhau mà không có sự kết hợp.	Stacking Models, một dạng Ensemble Learning
Cách tiếp cận thuật toán (Model Approach)	Cách tiếp cận đơn giản hơn, vì mỗi thuật toán được áp dụng độc lập. Các mô hình không được kết hợp hay tối ưu hóa theo cách đặc biệt nào khác.	Cách tiếp cận stacking giúp kết hợp sức mạnh của nhiều mô hình để cải thiện độ chính xác, với XGBoost là model level 1, LR, DT và NBC là các model level 2
Khả năng khắc phục hạn chế của thuật toán	Các mô hình đơn lẻ sẽ có điểm yếu và điểm mạnh riêng ở từng mô hình, khả năng tự khắc phục của một mô hình là điều khó có thể.	XGBoost là một thuật toán boosting mạnh mẽ, có khả năng tối ưu hóa lỗi sai từng bước, giảm thiểu bias và variance. Kết hợp với stacking, nó có thể khắc phục những hạn chế mà các mô hình đơn lẻ gặp phải.
Tối ưu hóa	Việc tối ưu hóa đối với các thuật toán đơn lẻ có thể là dễ dàng (do số lượng các siêu tham số không nhiều), nhưng độ hiệu quả không cao, do điểm yếu cố hữu của mô hình.	Với nhiều mô hình stacking, việc tối ưu hóa sẽ phức tạp (do phải tối ưu hóa từng những bước nhỏ của mỗi mô hình), nhưng hiệu quả đem lại cao hơn hẳn.

Bảng 4.4: So sánh việc áp dụng các mô hình độc lập với bộ dữ liệu

2. ADASYN - PSO - NN (Faris, 2018)

Tiêu chí	**Bài báo 'A Hybrid Swarm Intelligent Neural Network Model for Customer Churn Prediction and Identifying the Influencing Factors'**	**Bài báo này**
Xử lý dữ liệu mất cân bằng	Sử dụng ADASYN để tái mẫu các lớp thiểu số, tạo thêm mẫu từ các điểm gần nhất.	Không xử lý dữ liệu mất cân bằng
Tối ưu hóa trọng số đặc trưng	Sử dụng PSO để gán trọng số cho các đặc trưng dựa trên sức mạnh dự đoán của chúng.	Không tối ưu hóa trọng số đặc trưng mà sử dụng đặc trưng nguyên bản hoặc được xử lý thủ công.
Thuật toán chính	Mạng nơ-ron ngẫu nhiên có trọng số (RWN), linh hoạt điều chỉnh số lượng nơ-ron ẩn theo yêu cầu.	Stacking với XGBoost, Logistic Regression, Decision Tree, và Naive Bayes Classifier
Tối ưu hóa mô hình	PSO giúp tối ưu hóa cấu trúc mạng nơ-ron, bao gồm việc lựa chọn số lượng nơ-ron ẩn.	Tối ưu dựa trên việc điều chỉnh các siêu tham số quan trọng của mô hình nhằm kiểm soát mô hình
Khuynh hướng tiếp cận	Data-centric: nhấn mạnh việc cân bằng và tối ưu hóa đặc trưng qua PSO.	Model-centric: sử dụng đến các mô hình học máy phức tạp, khả năng tùy biến cao và tập trung tăng cường hiệu quả mô hình

Bảng 4.5: So sánh việc áp dụng hướng tiếp cận Data-Centric với bộ dữ liệu

Chương 5

Tổng kết

Báo cáo đã tái thực hiện những gì mà tác giả đã thực hiện trong bài báo về bài toán Dự đoán khả năng ngừng sử dụng dịch vụ viễn thông của người dùng trong tương lai.

Báo cáo đã tiền hành tiền xử lý và xây dựng bộ dữ liệu đầu vào của cho các thuật toán phân lớp được sử dụng. Qua các phương pháp được sử dụng cho bài toán Dự đoán khả năng ngừng sử dụng dịch vụ của người dùng, đã cho thấy việc xử lí và làm sạch dữ liệu, cũng như lựa chọn các siêu tham số cho các mô hình thuật toán có ảnh hưởng lớn đến kết quả đầu ra của bài toán. Qua thực nghiệm cũng có thể thấy, hai thuật toán phân lớp XGBoost và Stacking đã cho ra khả năng học tập (hay kết quả đầu ra) tốt nhất so với các thuật toán phân lớp truyền thống là cây quyết định, hồi quy Logistic, và Naive Bayes classifier. Kết quả thực nghiệm trong báo cáo có phần lệch tương đối nhỏ so với trong bài báo, có thể là do kĩ thuật xử lí dữ liệu và tùy chỉnh tham số khác so với bài báo.

Trong báo cáo vào cuối kì sắp tới, bài toán sẽ tiếp tục được thử nghiệm với các kỹ thuật khác với trong bài báo, nhằm mục đích tìm ra những phương pháp tốt hơn cho xử lí bộ dữ liệu, cũng như các mô hình thuật toán cho ra được kết quả bằng hoặc tốt hơn với bài báo

Tài liệu tham khảo

Amin, A., Al-Obeidat, F., Shah, B., Adnan, A., Loo, J. and Anwar, S. (2019), 'Customer churn prediction in telecommunication industry using data certainty', *Journal of Business Research* **94**, 290–301. 8.

Borja, B., Bernardino, C., Alex, C., Ricard, G. and David, M.-M. (2013), 'The architecture of a churn prediction system based on stream mining', *Frontiers in Artificial Intelligence and Applications* **256**, 157–166. 2.

Chatterjee, S. and Simonoff, J. (2020), *Handbook of Regression Analysis with Applications in R. Logistic Regression*, Wiley, Hoboken, NJ, USA. 22.

Chen, T. and Guestrin, C. (2016), Xgboost: A scalable tree boosting system, *in* 'Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', San Francisco, CA, USA. 21.

Domingos, P. (2012), 'A few useful things to know about machine learning', *Communications of the ACM* **55**, 78–87. 18.

Edwards, R., Šúri, M., Huld, T. and Dallemand, J. (n.d.), 'Gis-based assessment of cereal straw energy resource in the european union', http://citeseerx.ist.psu.edu/viewdoc/download? Accessed: 10 February 2020.

Faris, H. (2018), 'A hybrid swarm intelligent neural network model for customer churn prediction and identifying the influencing factors', *Information* **9**, 288. 16.

García, D., Nebot, and Vellido, A. (2017), 'Intelligent data analysis approaches to churn as a business problem: A survey', *Knowledge and Information Systems* **51**, 719–774. 1.

García-Torres, M., Gómez-Vela, F., Becerra-Alonso, D., Melián-Batista, B. and Moreno-Vega, J. (2015), Feature grouping and selection on high-dimensional microarray data, *in* 'Proceedings of the 2015 International Workshop on Data Mining with Industrial Applications (DMIA)', San Lorenzo, Paraguay. 19.

Jahromi, A., Moeini, M., Akbari, I. and Akbarzadeh, A. (2010), 'A dual-step multi-algorithm approach for churn prediction in pre-paid telecommunications service providers', *Journal of Innovation and Sustainability* **1**, 2179–3565. 11.

Kotler, P. (1994), *Marketing Management: Analysis, Planning, Implementation and Control*, Prentice-Hall, London, UK. 3.

Larose, C. and Larose, D. (2019), Naïve bayes classification, *in* 'Data Science Using Python and R', John Wiley & Sons, Inc., Hoboken, NJ, USA. 24.

Larose, D. and Larose, C. (2014), *Discovering Knowledge in Data: An Introduction to Data Mining*, John Wiley & Sons, New York, NY, USA. 17.

Motoda, H. and Liu, H. (2001), Feature selection, extraction and construction, *in* 'Communication of IICM (Institute of Information and Computing Machinery)', Vol. 5, pp. 67–72. 5.

Ngai, E., Xiu, L. and Chau, D. (2009), 'Application of data mining techniques in customer relationship management: A literature review and classification', *Expert Systems with Applications* **36**, 2592–2602. 4.

Obiedat, R., Al-kasassbeh, M., Faris, H. and Harfoushi, O. (2013), 'Customer churn prediction using a hybrid genetic programming approach', *Scientific Research Essays* **8**, 1289–1295. 14.

Reichheld, F. and Sasser, W. (1990), 'Zero defections: Quality comes to services', *Harvard Business Review* **68**, 105–111. 13.

Sabbeh, S. (2018), 'Machine-learning techniques for customer retention: A comparative study', *International Journal of Advanced Computer Science and Applications* **9**. 15.

Sharma, A. and Panigrahi, D. (2013), 'A neural network based approach for predicting customer churn in cellular network services', *International Journal of Computer Applications* **27**, 26–31. 7.

Sturges, H. (1926), 'The choice of a class interval', *Journal of the American Statistical Association* **21**, 65–66. 20.

Suzuki, J. (2020), *Decision Trees*, Springer, Singapore. 23.

Ting, K. and Witten, I. (1999), 'Issues in stacked generalization', *Journal of Artificial Intelligence Research* **10**, 271–289. 25.

Umayaparvathi, V. and Iyakutti, K. (2012), 'Applications of data mining techniques in telecom churn prediction', *International Journal of Computer Applications* **42**, 5–9. 10.

Vijaya, J. and Sivasankar, E. (2019), 'An efficient system for customer churn prediction through particle swarm optimization based feature selection model with simulated annealing', *Cluster Computing* **22**, 10757–10768. 9.

Zhang, Y., Qi, J., Shu, H. and Cao, J. (2007), A hybrid knn-lr classifier and its application in customer churn prediction, *in* 'Proceedings of the 2007 IEEE International Conference on Systems, Man and Cybernetics', Montréal, QC, Canada, pp. 3265–3269. 12.

Zhou, Z. (2012), *Ensemble Methods: Foundations and Algorithms*, CRC Press, Boca Raton, FL, USA. 26.