

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC VĂN HIẾN
KHOA KỸ THUẬT – CÔNG NGHỆ**



BÁO CÁO TIỂU LUẬN
Build a Logistics Regression model for
Customer Churn

GVHD: ThS. HỒ NHỰT MINH

SVTH: PHẠM HOÀNG TÙNG

MSSV: 221A020058

LỚP : 221A0201

TP. HỒ CHÍ MINH - 2024

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC VĂN HIẾN
KHOA KỸ THUẬT – CÔNG NGHỆ**



BÁO CÁO TIỂU LUẬN
Build a Logistics Regression model for
Customer Churn

GVHD: ThS. HỒ NHỰT MINH

SVTH: PHẠM HOÀNG TÙNG

MSSV: 221A020058

LỚP : 221A0201

TP. HỒ CHÍ MINH - 2024

MỤC LỤC

LỜI CẢM ƠN	1
MỞ ĐẦU	3
Giới thiệu về Hồi quy Logistic và Phân loại	4
Xây dựng mô hình Hồi quy Logistic	9
1.Các hệ số trong hồi quy logistic (Coefficients in logistics regression)	9
2.Khái niệm về Maximum Log-Likelihood (Concept of maximum log-likelihood)	11
3.Các chỉ số đánh giá hiệu suất trong hồi quy logistic (Performance metrics like confusion metric, recall, accuracy, precision, f1-score, AUC, and ROC)	12
4. Nhập dữ liệu và thư viện cần thiết (Importing the dataset and required libraries.)	15
5.Thực hiện Phân tích dữ liệu thăm dò cơ bản (EDA).(Performing basic Exploratory Data Analysis (EDA).)	15
6.Sử dụng thư viện matplotlib và seaborn để giải thích dữ liệu và trực quan hóa nâng cao(Using python libraries such as matplotlib and seaborn for data interpretation and advanced visualizations)	17
7.Kiểm tra và làm sạch dữ liệu(Data inspection and cleaning)	18
8.Sử dụng thư viện statsmodel và sklearn để xây dựng mô hình(Using statsmodel and sklearn libraries to build the model)	21
9.Đào tạo mô hình bằng các kỹ thuật Phân loại như Hồi quy Logistics,Training a model using Classification techniques like Logistics Regression,	22
10.Chia tập dữ liệu thành huấn luyện và kiểm tra bằng sklearn.(Splitting Dataset into Train and Test using sklearn.)	23
11.Dự đoán bằng cách sử dụng mô hình đã được đào tạo. Making predictions using the trained model.	24
12.Tăng sự tin tưởng vào mô hình bằng cách sử dụng các số liệu như điểm chính xác,ma trận nhầm lẫn, khả năng thu hồi, độ chính xác và điểm f1 (Gaining confidence in the model using metrics such as accuracy score, confusion matrix, recall, precision, and f1 score)	26
13.Xử lý dữ liệu không cân bằng bằng nhiều phương pháp khác nhau.(Handling the unbalanced data using various methods.)	29

14. Thực hiện lựa chọn tính năng bằng nhiều phương pháp (Performing feature selection with multiple methods)	30
15. Lưu mô hình tốt nhất ở định dạng pickle để sử dụng trong tương lai. Saving the best model in pickle format for future use.	33
Tổng Kết	34

LỜI CẢM ƠN

Thực tế hiện nay không có sự thành công nào mà không học hỏi tìm tòi, khám phá dù ít hay nhiều, dù trực tiếp hay gián tiếp về những cái hay mới mẻ của mọi người. Trong suốt thời gian làm bài tiểu luận, em đã tìm hiểu, học hỏi và nhận được sự hỗ trợ của giảng viên hướng dẫn, các anh khóa trên. Sau cùng, trong quá trình học tập cũng như trong thời gian làm bài tiểu luận không tránh khỏi những thiếu sót, em rất mong được sự góp ý quý báu của quý Thầy/Cô cũng như bạn bè để kết quả của em được hoàn thiện hơn.

Em xin chân thành cảm ơn quý Thầy/Cô trong khoa Kỹ thuật – Công nghệ, Trường Đại học Văn Hiến đã tận tình truyền đạt kiến thức, tạo điều kiện tốt nhất cho em học tập và thực hiện làm bài tiểu luận. Với vốn kiến thức được tiếp thu trong quá trình học không chỉ là nền tảng cho quá trình thực hiện mà còn là hành trang quý báu để em bước vào đời một cách vững chắc và tự tin.

Em xin chân thành cảm ơn thầy ThS. Hồ Nhật Minh đã tận tâm hướng dẫn em để hoàn thành bài tiểu luận. Nếu không có những lời hướng dẫn, dạy bảo của thầy thì em nghĩ bài tiểu luận của em khó có thể hoàn thiện được. Một lần nữa em xin chân thành cảm ơn thầy.

TP. Hồ Chí Minh, ngày... tháng... năm...

Sinh viên thực hiện

(Ký tên và ghi rõ họ tên)

Phạm Hoàng Tùng

TP. Hồ Chí Minh, ngày... tháng... năm...

Giảng viên hướng dẫn

(Ký tên và ghi rõ họ tên)

MỞ ĐẦU

➤ Lý do chọn đề tài

Trong bối cảnh cạnh tranh gay gắt hiện nay, việc duy trì khách hàng và ngăn chặn tình trạng rời bỏ dịch vụ (Customer Churn) đã trở thành một yếu tố sống còn đối với sự thành công của nhiều doanh nghiệp. Khách hàng trung thành không chỉ giúp doanh nghiệp duy trì doanh thu ổn định mà còn góp phần xây dựng hình ảnh thương hiệu và tạo đà phát triển dài hạn. Tuy nhiên, việc khách hàng quyết định ngừng sử dụng sản phẩm hoặc dịch vụ là một hiện tượng phổ biến mà các công ty luôn phải đối mặt. Để giảm thiểu tình trạng này, doanh nghiệp cần có những công cụ dự đoán hiệu quả, cho phép họ hành động kịp thời và xây dựng các chiến lược giữ chân khách hàng mạnh mẽ.

➤ Phương pháp nghiên cứu:

Một trong những phương pháp được sử dụng rộng rãi trong việc dự đoán tỷ lệ rời đi của khách hàng là hồi quy logistic. Đây là một kỹ thuật phân loại có giám sát, đặc biệt hiệu quả khi giải quyết các vấn đề có biến mục tiêu phân loại, như trường hợp khách hàng có thể rời bỏ hay tiếp tục sử dụng dịch vụ. Hồi quy logistic cho phép doanh nghiệp mô hình hóa mối quan hệ giữa các đặc trưng cụ thể của khách hàng và xác suất mà họ sẽ chấm dứt mối quan hệ với doanh nghiệp. Thông qua việc phân tích dữ liệu về khách hàng như độ tuổi, giới tính, thời gian sử dụng dịch vụ, hoặc số lần gọi đến trung tâm hỗ trợ, mô hình hồi quy logistic có thể cung cấp dự báo chính xác về việc khách hàng có khả năng rời bỏ dịch vụ trong tương lai.

Trong nghiên cứu này, tôi sẽ xây dựng một mô hình hồi quy logistic dựa trên một tập dữ liệu bao gồm 2.000 khách hàng với 16 đặc trưng khác nhau. Các đặc trưng này bao gồm các yếu tố quan trọng như thời gian đăng ký, số phút xem nội dung hàng tuần, số lần khách hàng gọi đến dịch vụ hỗ trợ, và nhiều yếu tố khác có thể ảnh hưởng đến quyết định rời bỏ của khách hàng. Mục tiêu của nghiên cứu là phân tích dữ liệu, áp dụng phương pháp hồi quy logistic để dự đoán khả năng rời bỏ dịch vụ của từng khách hàng.

Ngoài ra, nghiên cứu này không chỉ dừng lại ở việc xây dựng mô hình dự đoán, mà còn giúp cung cấp cho doanh nghiệp các hiểu biết chiến lược. Từ những kết quả dự đoán, doanh nghiệp có thể triển khai các biện pháp cụ thể để giữ chân khách hàng có nguy cơ rời đi cao, chẳng hạn như cung cấp các chương trình khuyến mãi, cải thiện chất lượng dịch vụ hoặc cá nhân hóa trải nghiệm của khách hàng. Việc áp dụng các phương pháp tiên tiến trong phân tích dữ liệu không chỉ mang lại lợi thế cạnh tranh cho doanh nghiệp mà còn góp phần tạo dựng mối quan hệ bền vững giữa doanh nghiệp và khách hàng.

➤ Mục tiêu, nhiệm vụ:

Qua nghiên cứu này, tôi hy vọng sẽ cung cấp một cái nhìn tổng quan về cách hồi quy logistic có thể được sử dụng trong việc dự đoán tình trạng rời đi khách hàng. Đồng thời, nghiên cứu sẽ đóng góp những kiến thức thực tiễn giúp doanh nghiệp tối ưu hóa chiến lược giữ chân khách hàng, giảm thiểu rủi ro rời bỏ, và tạo nền tảng vững chắc cho sự phát triển lâu dài.

Giới thiệu về Hồi quy Logistic và Phân loại

I, Phân loại

Phân loại (Classification) là một kỹ thuật quan trọng trong học máy, được sử dụng để dự đoán hoặc gán các quan sát vào một tập hợp các lớp hoặc danh mục nhất định. Đây là một trong những nền tảng cốt lõi giúp các mô hình học máy đưa ra quyết định về việc một đối tượng cụ thể thuộc về nhóm nào dựa trên dữ liệu đầu vào. Mục tiêu của phân loại là xây dựng một mô hình hoặc hàm ánh xạ có khả năng dự đoán chính xác nhãn của một quan sát mới, dựa trên các đặc trưng có sẵn từ dữ liệu huấn luyện. Trong các bài toán phân loại, biến mục tiêu thường là một biến rời rạc, đại diện cho các nhóm hoặc lớp.

Có hai loại phân loại phổ biến:

1. Phân loại nhị phân: Đây là dạng phân loại đơn giản nhất, trong đó biến mục tiêu chỉ có hai giá trị hoặc nhãn (ví dụ: khách hàng có rời bỏ dịch vụ hay không). Đây là dạng phân loại được ứng dụng nhiều nhất trong các bài toán thực tiễn, như việc dự đoán hành vi của khách hàng hoặc phát hiện gian lận tài chính.

2. Phân loại đa lớp: Trong trường hợp này, biến mục tiêu có nhiều hơn hai lớp (ví dụ: phân loại các loại sản phẩm, dự đoán các loại bệnh khác nhau). Đây là loại phân loại thường gặp trong các ứng dụng như phân tích hình ảnh hoặc nhận dạng giọng nói.

Quy trình cơ bản trong bài toán phân loại bao gồm các bước như:

- **Thu thập dữ liệu:** Tập hợp một tập dữ liệu đầy đủ các đặc trưng cần thiết và nhãn phân loại tương ứng.
- **Tiền xử lý dữ liệu:** Đây là bước quan trọng để làm sạch và chuẩn hóa dữ liệu, đảm bảo dữ liệu phù hợp với mô hình phân loại. Bước này thường bao gồm xử lý giá trị thiếu, mã hóa dữ liệu phân loại, và chuẩn hóa giá trị số.
- **Chọn mô hình phân loại:** Các mô hình phân loại phổ biến bao gồm hồi quy logistic, cây quyết định, k-Nearest Neighbors (k-NN), và mạng nơ-ron nhân tạo. Tùy thuộc vào bài toán cụ thể mà mỗi mô hình sẽ có những ưu và nhược điểm khác nhau.
- **Huấn luyện mô hình:** Mô hình sẽ được huấn luyện dựa trên tập dữ liệu huấn luyện, nơi các đặc trưng của dữ liệu được ánh xạ với nhãn phân loại để học cách đưa ra quyết định.
- **Kiểm thử mô hình:** Sau khi mô hình đã được huấn luyện, tập dữ liệu kiểm thử sẽ được sử dụng để đánh giá hiệu suất của mô hình.
- **Đánh giá mô hình:** Hiệu suất của mô hình được đo lường thông qua các chỉ số như độ chính xác (Accuracy), Recall, Precision, và F1-score. Các chỉ số này cung cấp cái nhìn toàn diện về khả năng của mô hình trong việc phân loại chính xác các đối tượng.

Một trong những thách thức lớn của bài toán phân loại là việc đối mặt với dữ liệu mất cân bằng, khi một lớp chiếm ưu thế hơn nhiều so với các lớp khác. Điều này có thể khiến mô hình trở nên thiên lệch và không dự đoán chính xác đối với các lớp ít xuất hiện. Các phương pháp như điều chỉnh trọng số, hoặc kỹ thuật SMOTE (Synthetic Minority Over-sampling Technique) được áp dụng để giải quyết vấn đề này. Phân loại là một kỹ thuật quan trọng, với nhiều ứng dụng thực tiễn trong kinh doanh và cuộc sống. Đặc biệt, trong các bài toán dự đoán khách hàng rời bỏ, phân loại nhị phân thông qua hồi quy logistic là phương pháp hiệu quả, giúp doanh nghiệp dự đoán hành vi của khách hàng và xây dựng các chiến lược giữ chân phù hợp.

II, Hồi quy logistic

Giới thiệu về hồi quy logistic

Hồi quy logistic là một thuật toán học máy có giám sát được sử dụng phổ biến trong các bài toán phân loại, đặc biệt là các bài toán nhị phân. Thay vì dự đoán giá trị liên tục như hồi quy tuyến tính, hồi quy logistic tập trung vào việc dự đoán xác suất một sự kiện xảy ra, với kết quả đầu ra thường được biểu diễn bằng hai trạng thái hoặc lớp, chẳng hạn như "có" hoặc "không", "rời bỏ" hoặc "không rời bỏ". Điều này làm cho hồi quy logistic trở thành công cụ hữu hiệu trong các lĩnh vực yêu cầu phân loại, chẳng hạn như dự đoán khách hàng rời bỏ dịch vụ (customer churn), phát hiện gian lận tài chính, hoặc chẩn đoán y tế.

Một trong những điểm đặc biệt của hồi quy logistic là nó sử dụng hàm sigmoid (hay còn gọi là hàm logistic) để chuyển đổi giá trị đầu ra thành xác suất trong khoảng từ 0 đến 1. Cụ thể, mô hình này ước tính xác suất mà một quan sát thuộc về một lớp cụ thể. Khi xác suất vượt qua một ngưỡng (thường là 0.5), mô hình sẽ dự đoán rằng quan sát thuộc về lớp đó. Hàm sigmoid có dạng:

$$P(y = 1|X) = \frac{1}{1 + e^{-z}}$$

Trong đó, z là một kết hợp tuyến tính của các biến đầu vào và các tham số cần ước lượng. Chính nhờ hàm sigmoid mà hồi quy logistic có khả năng mô hình hóa các dữ liệu có kết quả nhị phân.

Ứng dụng của hồi quy logistic

Hồi quy logistic được áp dụng rộng rãi trong nhiều lĩnh vực khác nhau nhờ khả năng dự đoán và phân loại hiệu quả:

Dự đoán hành vi khách hàng : Một trong những ứng dụng phổ biến nhất của hồi quy logistic là trong việc dự đoán hành vi khách hàng, như dự đoán khả năng rời bỏ dịch vụ

của khách hàng. Điều này giúp các doanh nghiệp có thể xây dựng các chiến lược giữ chân khách hàng hiệu quả.

Phát hiện gian lận tài chính : Các tổ chức tài chính thường sử dụng hồi quy logistic để phát hiện các giao dịch có dấu hiệu gian lận, giúp bảo vệ doanh nghiệp và khách hàng khỏi các rủi ro về tài chính.

Chẩn đoán bệnh lý : Trong y tế, hồi quy logistic được sử dụng để dự đoán khả năng bệnh nhân mắc bệnh dựa trên các yếu tố nguy cơ và triệu chứng lâm sàng.

Ưu điểm của hồi quy logistic

Giải thích trực quan : Một trong những lợi thế lớn của hồi quy logistic là khả năng giải thích dễ hiểu về mối quan hệ giữa các đặc trưng và xác suất xảy ra sự kiện. Điều này rất hữu ích khi doanh nghiệp hoặc các tổ chức cần đưa ra quyết định dựa trên phân tích dữ liệu.

Hiệu quả trong bài toán phân loại nhị phân : Với các bài toán phân loại nhị phân, hồi quy logistic là một trong những phương pháp đơn giản và hiệu quả nhất. Nó dễ dàng triển khai và thường cho kết quả tốt với dữ liệu có quy mô vừa và nhỏ.

Không yêu cầu phân phối chuẩn : Hồi quy logistic không yêu cầu dữ liệu phải tuân theo phân phối chuẩn, giúp nó linh hoạt trong việc xử lý các tập dữ liệu khác nhau.

Hạn chế của hồi quy logistic

Tuyến tính trong các đặc trưng : Hồi quy logistic giả định rằng mối quan hệ giữa các đặc trưng đầu vào và biến mục tiêu là tuyến tính. Trong thực tế, điều này không phải lúc nào cũng đúng, và mô hình có thể không hiệu quả khi dữ liệu có mối quan hệ phi tuyến mạnh.

Không phù hợp với dữ liệu phức tạp : Đối với các bài toán phức tạp hoặc dữ liệu có nhiều chiều, hồi quy logistic có thể không đạt được hiệu quả cao như các mô hình khác như mạng nơ-ron, cây quyết định hay các mô hình phi tuyến khác.

Hồi quy logistic trong bài toán dự đoán khách hàng rời bỏ dịch vụ

Trong bối cảnh kinh doanh, hồi quy logistic thường được sử dụng để dự đoán khách hàng rời bỏ dịch vụ, một trong những vấn đề quan trọng đối với các doanh nghiệp cung cấp dịch vụ đăng ký dài hạn. Việc dự đoán chính xác liệu một khách hàng có rời bỏ dịch vụ hay không giúp doanh nghiệp có thể triển khai các chiến lược giữ chân khách hàng trước khi họ quyết định ngừng sử dụng dịch vụ.

Ví dụ, bằng cách sử dụng hồi quy logistic, một doanh nghiệp có thể phân tích các đặc trưng như thời gian sử dụng dịch vụ, số lần liên lạc với bộ phận hỗ trợ, hoặc số phút sử dụng dịch vụ hàng tuần để dự đoán xác suất khách hàng sẽ rời bỏ. Nếu xác suất rời bỏ

cao, doanh nghiệp có thể thực hiện các biện pháp như cung cấp khuyến mãi, cải thiện dịch vụ, hoặc chăm sóc khách hàng tốt hơn nhằm giữ chân khách hàng đó. Hồi quy logistic là một công cụ mạnh mẽ trong việc dự đoán các sự kiện phân loại nhị phân. Nó không chỉ đơn giản và dễ hiểu, mà còn có tính ứng dụng cao trong nhiều lĩnh vực. Trong bối cảnh doanh nghiệp, hồi quy logistic giúp các công ty không chỉ hiểu rõ hành vi của khách hàng mà còn xây dựng các chiến lược giữ chân khách hàng một cách hiệu quả, từ đó nâng cao lợi thế cạnh tranh và duy trì sự phát triển bền vững.

III, Hiểu biết về hàm logit

Hàm logit là một khái niệm quan trọng trong hồi quy logistic, được sử dụng để chuyển đổi xác suất thành một giá trị mà có thể xử lý trong mô hình hồi quy. Hàm này cho phép chúng ta biểu diễn mối quan hệ giữa xác suất và các đặc trưng đầu vào theo một cách tuyến tính. Cụ thể, hàm logit được định nghĩa như sau:

$$\text{logit}(p) = \log \left(\frac{p}{1-p} \right)$$

Trong đó:

- p là xác suất mà một quan sát thuộc về lớp mục tiêu (ví dụ: xác suất một khách hàng rời bỏ dịch vụ).
- \log là hàm logarithm tự nhiên (logarithm tự nhiên với cơ số e).

Đặc điểm của hàm logit

Biến đổi xác suất: Hàm logit biến đổi xác suất p thành giá trị vô hướng không giới hạn, có thể dao động từ $-\infty$ đến $+\infty$. Điều này rất quan trọng vì mô hình hồi quy logistic cần dự đoán một giá trị tuyến tính để tính toán và ước lượng các tham số.

Giá trị dương và âm: Khi xác suất p lớn hơn 0.5, hàm logit cho giá trị dương, điều này cho thấy rằng khả năng thuộc về lớp mục tiêu cao hơn. Ngược lại, nếu $p < 0.5$, hàm logit cho giá trị âm, biểu thị rằng xác suất thuộc về lớp không phải mục tiêu lớn hơn.

Tính liên tục và tăng dần: Hàm logit là một hàm liên tục và tăng dần, nghĩa là nếu xác suất tăng, giá trị của hàm logit cũng sẽ tăng. Điều này giúp chúng ta duy trì tính chất dự đoán khi chuyển đổi giữa xác suất và các tham số đầu vào.

Mối quan hệ giữa logit và hồi quy logistic

Trong hồi quy logistic, mối quan hệ giữa các đặc trưng đầu vào và xác suất mà một quan sát thuộc về một lớp cụ thể được mô hình hóa qua hàm logit. Cụ thể, mô hình hồi quy logistic có thể được diễn đạt như sau:

$$\text{logit}(p) = z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n$$

Trong đó:

- z là giá trị đầu ra tuyến tính từ các biến độc lập X_1, X_2, \dots, X_n .
- β_0 là hệ số chặn (intercept), và $\beta_1, \beta_2, \dots, \beta_n$ là các hệ số hồi quy tương ứng cho từng biến độc lập.

Để tính toán xác suất từ giá trị z , chúng ta sử dụng hàm sigmoid:

$$p = \frac{1}{1 + e^{-z}}$$

Hàm sigmoid giúp chuyển đổi giá trị z thành xác suất p trong khoảng từ 0 đến 1. Bằng cách này, hồi quy logistic có thể dự đoán xác suất thuộc về một trong hai lớp dựa trên các đặc trưng đầu vào.

Ứng dụng của hàm logit

Hàm logit không chỉ được sử dụng trong hồi quy logistic mà còn có ứng dụng trong nhiều lĩnh vực khác, như kinh tế học, sinh học và y học. Cụ thể, nó giúp trong việc:

- **Mô hình hóa các mối quan hệ phi tuyến:** Khi mối quan hệ giữa các biến không phải là tuyến tính, hàm logit cho phép các nhà nghiên cứu nắm bắt các xu hướng phức tạp hơn.
- **Phân tích ảnh hưởng của các yếu tố:** Trong các nghiên cứu về chính sách, hàm logit có thể được sử dụng để phân tích ảnh hưởng của các yếu tố như thu nhập, giáo dục, hoặc tuổi tác đến khả năng tham gia vào một hành động cụ thể (như bỏ phiếu).

Hàm logit là một phần quan trọng trong việc hiểu cách mà hồi quy logistic hoạt động. Nó không chỉ cho phép chúng ta chuyển đổi xác suất thành một giá trị có thể xử lý được trong mô hình mà còn giúp duy trì mối quan hệ tuyến tính giữa các đặc trưng và xác suất. Hiểu rõ về hàm logit là điều cần thiết để phát triển và áp dụng hồi quy logistic hiệu quả trong các bài toán dự đoán và phân loại.

Xây dựng mô hình Hồi quy Logistic

1. Các hệ số trong hồi quy logistic (Coefficients in logistics regression)

Trong hồi quy logistic, các hệ số (coefficients) là những giá trị xác định mức độ ảnh hưởng của từng biến độc lập (đặc trưng) đến xác suất xảy ra một sự kiện, chẳng hạn như việc khách hàng rời bỏ dịch vụ. Các hệ số này tương tự như trong hồi quy tuyến tính, nhưng trong hồi quy logistic, chúng được sử dụng để mô hình hóa mối quan hệ giữa các biến độc lập và logit (log-odds) của xác suất sự kiện xảy ra.

Hồi quy logistic có phương trình như sau:

$$\text{logit}(p) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n$$

Trong đó:

- β_0 là hệ số chặn (intercept), đại diện cho logit khi tất cả các biến độc lập X_1, X_2, \dots, X_n bằng 0.
- $\beta_1, \beta_2, \dots, \beta_n$ là các hệ số hồi quy tương ứng với các biến độc lập X_1, X_2, \dots, X_n .
- X_1, X_2, \dots, X_n là các giá trị của các đặc trưng đầu vào.

Ý nghĩa của các hệ số trong hồi quy logistic

Tác động của hệ số:

Mỗi hệ số β_i biểu thị mức độ ảnh hưởng của biến X_i đến logit của xác suất. Nếu $\beta_i > 0$, điều này cho thấy biến X_i làm tăng logit của xác suất xảy ra sự kiện (xác suất thuộc về lớp mục tiêu). Ngược lại, nếu $\beta_i < 0$, biến X_i làm giảm logit của xác suất xảy ra sự kiện.

Mối quan hệ với log-odds:

Hồi quy logistic dựa trên khái niệm log-odds, tức là logarithm tự nhiên của tỷ số giữa xác suất sự kiện xảy ra và không xảy ra. Một sự thay đổi đơn vị trong X_i dẫn đến thay đổi log-odds của sự kiện xảy ra theo một lượng bằng β_i . Cụ thể:

$$\text{log-odds} = \log \left(\frac{p}{1-p} \right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n$$

Tác động đến xác suất:

Mặc dù hệ số ảnh hưởng đến log-odds, chúng cũng có tác động đến xác suất xảy ra sự kiện. Nếu hệ số β_i dương, xác suất xảy ra sự kiện tăng khi biến độc lập X_i tăng. Ngược lại, nếu β_i âm, xác suất xảy ra sự kiện giảm khi X_i tăng.

Tỷ lệ odds (odds ratio):

Để hiểu rõ hơn về tác động của các hệ số lên xác suất, chúng ta có thể tính **tỷ lệ odds** từ các hệ số hồi quy. Tỷ lệ odds được tính bằng cách lấy e^{β_i} , trong đó e là cơ số của logarithm tự nhiên.

Tỷ lệ odds cho thấy mức thay đổi về odds (tỷ số giữa xác suất sự kiện xảy ra và không xảy ra) khi biến độc lập X_i thay đổi 1 đơn vị. Nếu:

- $e^{\beta_i} > 1$: Odds của sự kiện xảy ra tăng khi X_i tăng.
- $e^{\beta_i} < 1$: Odds của sự kiện xảy ra giảm khi X_i tăng.
- $e^{\beta_i} = 1$: Biến X_i không ảnh hưởng đến xác suất xảy ra sự kiện.

Ý nghĩa thực tiễn của các hệ số trong hồi quy logistic

● Giải thích tác động của các yếu tố:

Các hệ số giúp chúng ta hiểu rõ hơn về mối quan hệ giữa các đặc trưng và xác suất sự kiện xảy ra. Ví dụ, trong một mô hình dự đoán khách hàng rời bỏ dịch vụ, nếu hệ số của biến số "số lần gọi hỗ trợ khách hàng" là dương và có giá trị lớn, điều này cho thấy rằng càng nhiều lần khách hàng gọi hỗ trợ, khả năng họ rời bỏ dịch vụ càng cao.

● Phân tích yếu tố ảnh hưởng chính:

Khi xây dựng mô hình hồi quy logistic, các hệ số lớn và có ý nghĩa thống kê cao chỉ ra rằng các biến tương ứng có tác động lớn đến xác suất xảy ra sự kiện. Điều này giúp doanh nghiệp tập trung vào những yếu tố quan trọng nhất khi đưa ra quyết định.

● Ứng dụng trong chiến lược kinh doanh:

Dựa trên giá trị của các hệ số hồi quy, doanh nghiệp có thể thực hiện các biện pháp cụ thể để giảm tỷ lệ rời bỏ dịch vụ. Ví dụ, nếu hệ số của biến "số ngày không hoạt động" là dương và lớn, doanh nghiệp có thể khuyến khích khách hàng sử dụng dịch vụ thường xuyên hơn để giảm nguy cơ rời bỏ.

Các hệ số trong hồi quy logistic không chỉ đơn giản là các tham số toán học, mà còn cung cấp cái nhìn sâu sắc về mối quan hệ giữa các đặc trưng và xác suất xảy ra của một sự kiện. Hiểu rõ ý nghĩa của chúng giúp chúng ta phân tích, dự đoán và đưa ra các quyết định chính xác hơn trong các tình huống thực tế, như dự đoán khách hàng rời bỏ dịch vụ, phát hiện gian lận, hay phân tích rủi ro trong tài chính.

2. Khái niệm về Maximum Log-Likelihood (Concept of maximum log-likelihood)

Khái niệm về Maximum Log-Likelihood

Trong hồi quy logistic, Maximum Likelihood Estimation (MLE) là phương pháp chính được sử dụng để ước lượng các tham số của mô hình. Mục tiêu của MLE là tìm ra các giá trị tham số tối ưu sao cho xác suất quan sát được dữ liệu thực tế là cao nhất. Nói cách khác, MLE tìm ra các tham số sao cho mô hình dự đoán tốt nhất dựa trên dữ liệu thực.

Hàm Likelihood

Hàm likelihood đo lường mức độ "khả thi" của một bộ tham số cho dữ liệu quan sát. Khi áp dụng vào hồi quy logistic, nó tính toán xác suất một sự kiện xảy ra (ví dụ: khách hàng rời bỏ dịch vụ) hoặc không xảy ra, dựa trên các đặc trưng của dữ liệu. Hàm này được tính bằng cách nhân các xác suất của tất cả các quan sát trong tập dữ liệu.

Tuy nhiên, vì các xác suất thường rất nhỏ nên việc nhân các giá trị này có thể dẫn đến số liệu rất bé, gây khó khăn cho tính toán. Do đó, thay vì dùng trực tiếp hàm likelihood, người ta thường sử dụng log-likelihood, tức là lấy logarit của hàm likelihood để dễ tính toán hơn.

Hàm Log-Likelihood

Hàm log-likelihood là phiên bản logarit của hàm likelihood, giúp biến phép nhân phức tạp thành phép cộng đơn giản hơn. Mục tiêu của MLE là tìm ra các giá trị tham số làm cho hàm log-likelihood đạt giá trị cao nhất, tức là xác suất của dữ liệu quan sát được lớn nhất.

Ước lượng hợp lý tối đa (MLE)

Ước lượng hợp lý tối đa (MLE) là quá trình tìm các tham số mô hình sao cho hàm log-likelihood đạt giá trị cao nhất. Trong hồi quy logistic, điều này có nghĩa là tìm ra các tham số sao cho mô hình dự đoán tốt nhất xác suất xảy ra sự kiện dựa trên dữ liệu.

Khi thực hiện MLE, chúng ta thường sử dụng các thuật toán tối ưu hóa như Gradient Descent để tối đa hóa hàm log-likelihood và tìm ra các hệ số phù hợp cho mô hình hồi quy logistic. Quá trình này đảm bảo rằng mô hình được điều chỉnh tốt nhất với dữ liệu thực tế.

Ý nghĩa của MLE trong hồi quy logistic

Trong hồi quy logistic, việc tối đa hóa log-likelihood giúp mô hình tìm ra mối quan hệ tốt nhất giữa các biến độc lập (đặc trưng) và xác suất xảy ra của sự kiện (chẳng hạn như

khách hàng rời bỏ dịch vụ). Các hệ số tham số được ước lượng thông qua MLE cho chúng ta biết mức độ ảnh hưởng của mỗi biến đến xác suất xảy ra sự kiện.

Ưu điểm của MLE

- Chính xác : MLE giúp tìm ra tham số tối ưu cho mô hình, đảm bảo mô hình dự đoán tốt nhất cho dữ liệu.
- Linh hoạt : Phương pháp này có thể áp dụng trong nhiều loại mô hình thống kê và học máy, không chỉ giới hạn trong hồi quy logistic.

Maximum Log-Likelihood là một phương pháp ước lượng quan trọng trong hồi quy logistic. Nó giúp xác định các tham số sao cho mô hình có thể dự đoán chính xác nhất xác suất xảy ra sự kiện, từ đó tối ưu hóa sự phù hợp của mô hình với dữ liệu thực tế. Phương pháp MLE không chỉ hiệu quả mà còn là nền tảng quan trọng trong nhiều kỹ thuật phân tích dữ liệu và học máy.

3.Các chỉ số đánh giá hiệu suất trong hồi quy logistic (Performance metrics like confusion metric, recall, accuracy, precision, f1-score, AUC, and ROC)

Khi xây dựng một mô hình hồi quy logistic để dự đoán các kết quả phân loại, việc đánh giá hiệu suất của mô hình là rất quan trọng để đảm bảo rằng nó hoạt động tốt. Dưới đây là các chỉ số đánh giá hiệu suất phổ biến, bao gồm ma trận nhầm lẫn, độ chính xác, độ nhạy (recall), độ chính xác (precision), điểm F1, AUC và ROC.

➤ **Ma trận nhầm lẫn (Confusion Matrix)**

Ma trận nhầm lẫn là một công cụ trực quan dùng để đánh giá hiệu suất của một mô hình phân loại. Nó cho thấy số lượng dự đoán đúng và sai của mô hình trong các lớp khác nhau. Ma trận nhầm lẫn thường có cấu trúc 2x2 cho bài toán phân loại nhị phân:

True Positive (TP): Số lượng dự đoán đúng cho lớp Positive.

True Negative (TN): Số lượng dự đoán đúng cho lớp Negative.

False Positive (FP): Số lượng dự đoán sai cho lớp Positive (còn gọi là Type I Error).

False Negative (FN): Số lượng dự đoán sai cho lớp Negative (còn gọi là Type II Error).

➤ **Độ chính xác (Accuracy)**

Độ chính xác là tỷ lệ phần trăm dự đoán đúng của mô hình trên tổng số quan sát. Nó được tính bằng công thức:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Độ chính xác là chỉ số hữu ích, nhưng nó có thể gây hiểu nhầm khi dữ liệu không cân bằng (tức là một lớp có nhiều quan sát hơn lớp khác).

➤ Độ nhạy (Recall)

Độ nhạy, còn gọi là độ nhớ (sensitivity), đo lường khả năng của mô hình trong việc phát hiện các trường hợp Positive. Nó được tính bằng công thức:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Độ nhạy cho biết tỷ lệ các quan sát thực tế là Positive mà mô hình dự đoán đúng.

➤ Độ chính xác (Precision)

Độ chính xác đo lường tỷ lệ các dự đoán Positive mà thực sự là Positive. Nó được tính bằng công thức:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Độ chính xác thể hiện khả năng của mô hình trong việc tránh dự đoán sai các trường hợp Positive.

➤ Điểm F1 (F1-Score)

Điểm F1 là một chỉ số kết hợp giữa độ nhạy và độ chính xác, giúp cân bằng giữa hai yếu tố này. Điểm F1 có giá trị từ 0 đến 1, với 1 là tốt nhất. Nó được tính bằng công thức:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Điểm F1 là chỉ số hữu ích khi cần một đánh giá tổng thể cho mô hình, đặc biệt là khi dữ liệu không cân bằng.

➤ AUC (Area Under the Curve)

AUC, hoặc diện tích dưới đường cong, thường được sử dụng trong các bài toán phân loại nhị phân để đánh giá khả năng phân biệt giữa hai lớp. AUC đo lường khả năng của mô hình trong việc phân loại đúng các lớp. Giá trị của AUC nằm trong khoảng từ 0 đến 1:

- $AUC = 1$: Mô hình hoàn hảo, phân loại chính xác tất cả các quan sát.
- $AUC = 0.5$: Mô hình không phân biệt được, tương đương với ngẫu nhiên.
- $AUC < 0.5$: Mô hình dự đoán sai.

➤ ROC (Receiver Operating Characteristic)

ROC là một biểu đồ thể hiện mối quan hệ giữa độ nhạy (Recall) và tỷ lệ dương tính giả (False Positive Rate) cho các ngưỡng khác nhau của mô hình. Đường cong ROC cho phép đánh giá khả năng phân loại của mô hình. Một mô hình tốt sẽ có đường cong ROC gần với điểm (0,1) trên biểu đồ, cho thấy khả năng phát hiện các trường hợp Positive cao với tỷ lệ dương tính giả thấp.

- ❖ Việc sử dụng các chỉ số đánh giá hiệu suất như ma trận nhầm lẫn, độ chính xác, độ nhạy, độ chính xác, điểm F1, AUC và ROC giúp chúng ta có cái nhìn tổng quát về khả năng dự đoán của mô hình hồi quy logistic. Các chỉ số này không chỉ giúp đánh giá mô hình mà còn cung cấp thông tin cần thiết để điều chỉnh và cải thiện mô hình, nhằm nâng cao hiệu suất dự đoán trong các bài toán thực tiễn.

4. Nhập dữ liệu và thư viện cần thiết (Importing the dataset and required libraries.)

```
import pandas as pd # Thư viện xử lý dữ liệu
```

```
import numpy as np # Thư viện toán học
```

```
import seaborn as sns # Thư viện trực quan hóa dữ liệu
```

```
import matplotlib.pyplot as plt # Thư viện vẽ biểu đồ
```

```
from sklearn.model_selection import train_test_split # Tách dữ liệu
```

```
from sklearn.linear_model import LogisticRegression # Mô hình hồi quy logistic
```

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, roc_curve # Các chỉ số đánh giá
```

```
import statsmodels.api as sm # Thư viện statsmodels cho hồi quy
```

```
import pickle # Thư viện lưu trữ mô hình
```

Nhập dữ liệu vào mô hình

```
[121] # import pandas
import pandas as pd
df=pd.read_csv("/content/drive/MyDrive/hoc may/13 Build a Logistic Regression Model in Python from Scratch/Data (1)/Data/data_regression.csv")

# get the first 10 rows
df.head(10)
```

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen	mail_subscribed	weekly_mins_watched	minimum_daily_mins	maximum_daily_mins	weekly_max_night_mins
0	2015	100198	409-8743	Female	36	62	no	no	148.35	12.2	16.81	82
1	2015	100643	340-5930	Female	39	149	no	no	294.45	7.7	33.37	87
2	2015	100756	372-3750	Female	65	126	no	no	87.30	11.9	9.89	91
3	2015	101595	331-4902	Female	24	131	no	yes	321.30	9.5	36.41	102
4	2015	101653	351-8398	Female	40	191	no	no	243.00	10.9	27.54	83
8	2015	103408	413-4039	Male	61	205	no	yes	263.70	7.8	29.89	64
9	2015	103676	338-5207	Male	31	63	no	no	316.80	12.3	35.90	58
10	2015	103697	411-9554	Male	34	114	no	yes	338.70	8.4	38.39	100
11	2015	103738	392-5296	Male	34	107	no	no	201.00	7.3	22.78	79
12	2015	104025	380-6722	Female	30	84	no	no	112.95	12.3	12.80	134

5.Thực hiện Phân tích dữ liệu thăm dò cơ bản (EDA).(Performing basic Exploratory Data Analysis (EDA).)

Phân tích khám phá dữ liệu giúp hiểu rõ hơn về dữ liệu trước khi xây dựng mô hình.

```
# Hiển thị thông tin tổng quát về dữ liệu
```

```
print(data.info())
```

```
# Kiểm tra số lượng giá trị bị thiếu
```

```
print(data.isnull().sum())
```

```
# Trực quan hóa phân phối của các biến
```

```

sns.histplot(data['Age']) # Ví dụ cho biến 'Age'

plt.title('Distribution of Age')

plt.show()

# Kiểm tra mối quan hệ giữa các biến

sns.pairplot(data)

plt.show()

# Kiểm tra tương quan giữa các biến

correlation_matrix = data.corr()

sns.heatmap(correlation_matrix, annot=True)

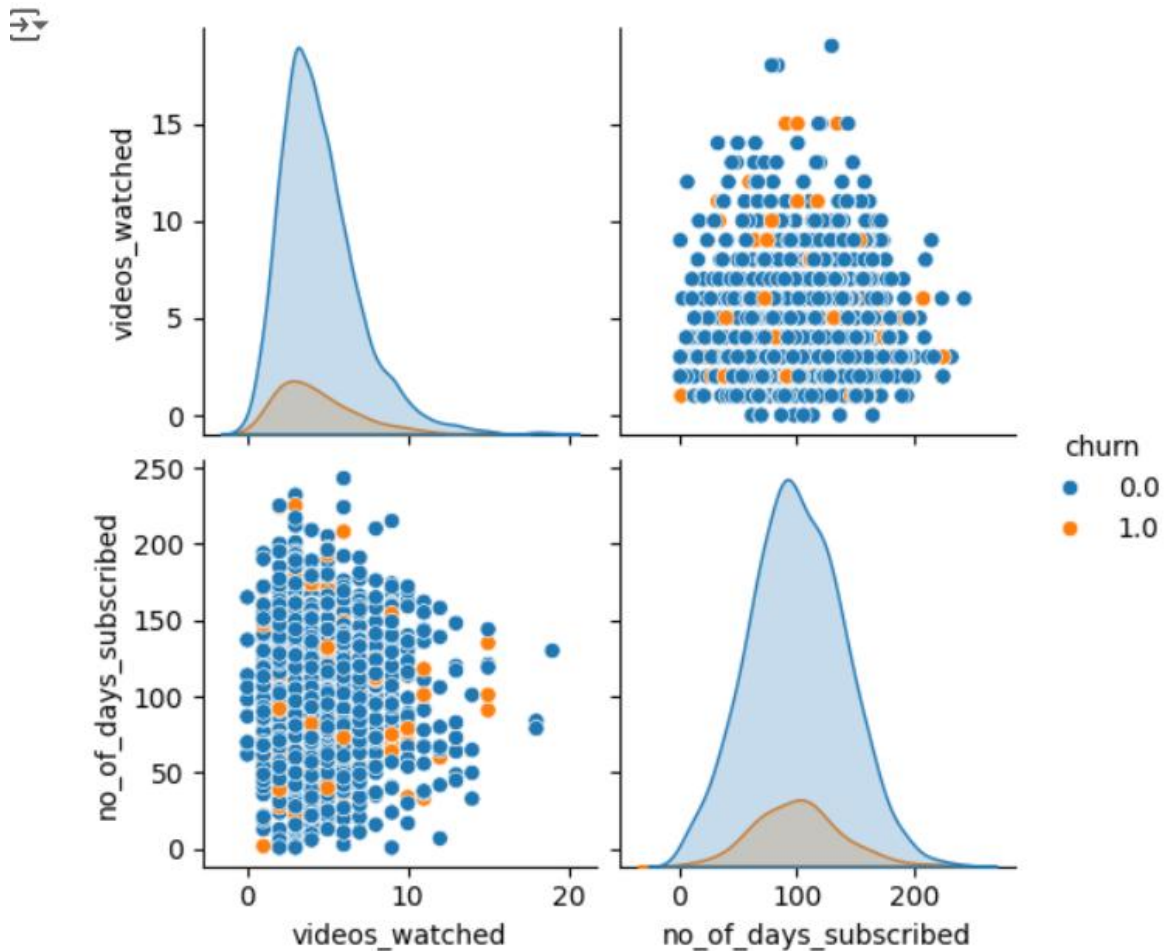
plt.title('Correlation Matrix')

plt.show()

```

Exploratory Data Analysis (EDA) là bước quan trọng giúp hiểu rõ dữ liệu trước khi xây dựng mô hình. Đầu tiên, kiểm tra kích thước, kiểu dữ liệu và xem qua vài dòng đầu để có cái nhìn tổng quan. Sau đó, kiểm tra các giá trị bị thiếu và trực quan hóa chúng bằng heatmap. Phân tích phân phối của các biến số qua biểu đồ Histogram và Boxplot giúp phát hiện ngoại lệ. Với biến phân loại, biểu đồ cột thể hiện tần suất xuất hiện của từng giá trị. Tiếp theo, mối quan hệ giữa các biến được đánh giá qua ma trận tương quan. Cuối cùng, phân tích mối quan hệ giữa biến độc lập và biến mục tiêu giúp hiểu tác động của các biến đến kết quả dự đoán. EDA giúp phát hiện vấn đề và chuẩn bị dữ liệu tốt hơn cho mô hình.

6. Sử dụng thư viện matplotlib và seaborn để giải thích dữ liệu và trực quan hóa nâng cao (Using python libraries such as matplotlib and seaborn for data interpretation and advanced visualizations)



Biểu đồ phân tán giữa hai biến

```
sns.scatterplot(x='Age', y='Weekly mins watched', hue='Churn', data=data)
```

```
plt.title('Age vs Weekly Mins Watched by Churn Status')
```

```
plt.show()
```

function for creating plots for selective columns only

```
def selected_diagnostic(df, class_col, cols_to_eval):
```

```
    import seaborn as sns
```

```
    cols_to_eval.append(class_col)
```

```
    X = df[cols_to_eval] # only selective columns
```

```
    sns.pairplot(X, hue = class_col) # plot
```

Đoạn mã này định nghĩa một **hàm** có tên là `selected_diagnostic` dùng để tạo các biểu đồ phân tán (scatter plots) cho các cột được chọn trong một DataFrame. Cụ thể, nó sử dụng thư viện `seaborn` để trực quan hóa mối quan hệ giữa các biến trong dữ liệu.

Giải thích chi tiết:

Đầu vào của hàm:

1. `df`: DataFrame chứa dữ liệu mà bạn muốn trực quan hóa.
2. `class_col`: Tên của cột phân loại (nhãn) mà bạn muốn sử dụng làm biến phân loại, tức là dùng để phân loại các nhóm khác nhau (thường là biến mục tiêu).
3. `cols_to_eval`: Danh sách các cột bạn muốn đánh giá bằng biểu đồ.

Các bước bên trong hàm:

1. **Import seaborn**: Đảm bảo rằng thư viện `seaborn` được nhập để sử dụng các chức năng vẽ biểu đồ.
2. **Thêm cột phân loại** (`class_col`) vào danh sách các cột được đánh giá (`cols_to_eval`) để đảm bảo rằng cột này cũng được sử dụng trong việc trực quan hóa.
3. **Chọn cột dữ liệu**: Tạo một DataFrame con chỉ chứa các cột bạn muốn đánh giá (bao gồm cả cột phân loại).
4. **Sử dụng pairplot**: Hàm `sns.pairplot` vẽ các biểu đồ phân tán giữa các cặp cột khác nhau trong DataFrame, sử dụng biến phân loại để xác định màu sắc của các điểm dữ liệu.

7. Kiểm tra và làm sạch dữ liệu (Data inspection and cleaning)

Trước khi xây dựng mô hình, việc làm sạch dữ liệu là rất cần thiết. Chúng ta cần kiểm tra xem có giá trị nào bị thiếu, loại bỏ các cột không cần thiết và xử lý các giá trị bị thiếu để đảm bảo rằng dữ liệu được sử dụng là chính xác và đáng tin cậy.

```
# check for the missing values and dataframes
```

```
def inspection(dataframe):
```

```
    import pandas as pd
```

```
    import seaborn as sns
```

```
    print("Types of the variables we are working with:")
```

```
    print(dataframe.dtypes) # dtypes
```

```
    print("Total Samples with missing values:")
```

```
    print(df.isnull().any(axis=1).sum()) # null values
```

```
    print("Total Missing Values per Variable")
```

```
print(df.isnull().sum())
```

```
print("Map of missing values")
```

```
sns.heatmap(dataframe.isnull())
```

Hàm `inspection()` trong đoạn mã trên được sử dụng để kiểm tra các giá trị bị thiếu và kiểu dữ liệu của các cột trong một DataFrame. Dưới đây là chi tiết từng phần của hàm:

```
print(dataframe.dtypes):
```

In ra kiểu dữ liệu của từng cột trong DataFrame (ví dụ: số nguyên, chuỗi ký tự, số thực). Điều này giúp bạn hiểu rõ các loại dữ liệu mà bạn đang làm việc để xác định có cần thay đổi hoặc xử lý thêm hay không.

```
print(df.isnull().any(axis=1).sum()):
```

Kiểm tra và đếm tổng số dòng có ít nhất một giá trị bị thiếu.

`isnull()` sẽ trả về True nếu có giá trị bị thiếu (NaN), và `any(axis=1)` sẽ kiểm tra từng dòng. Cuối cùng `.sum()` tổng hợp số dòng có giá trị bị thiếu.

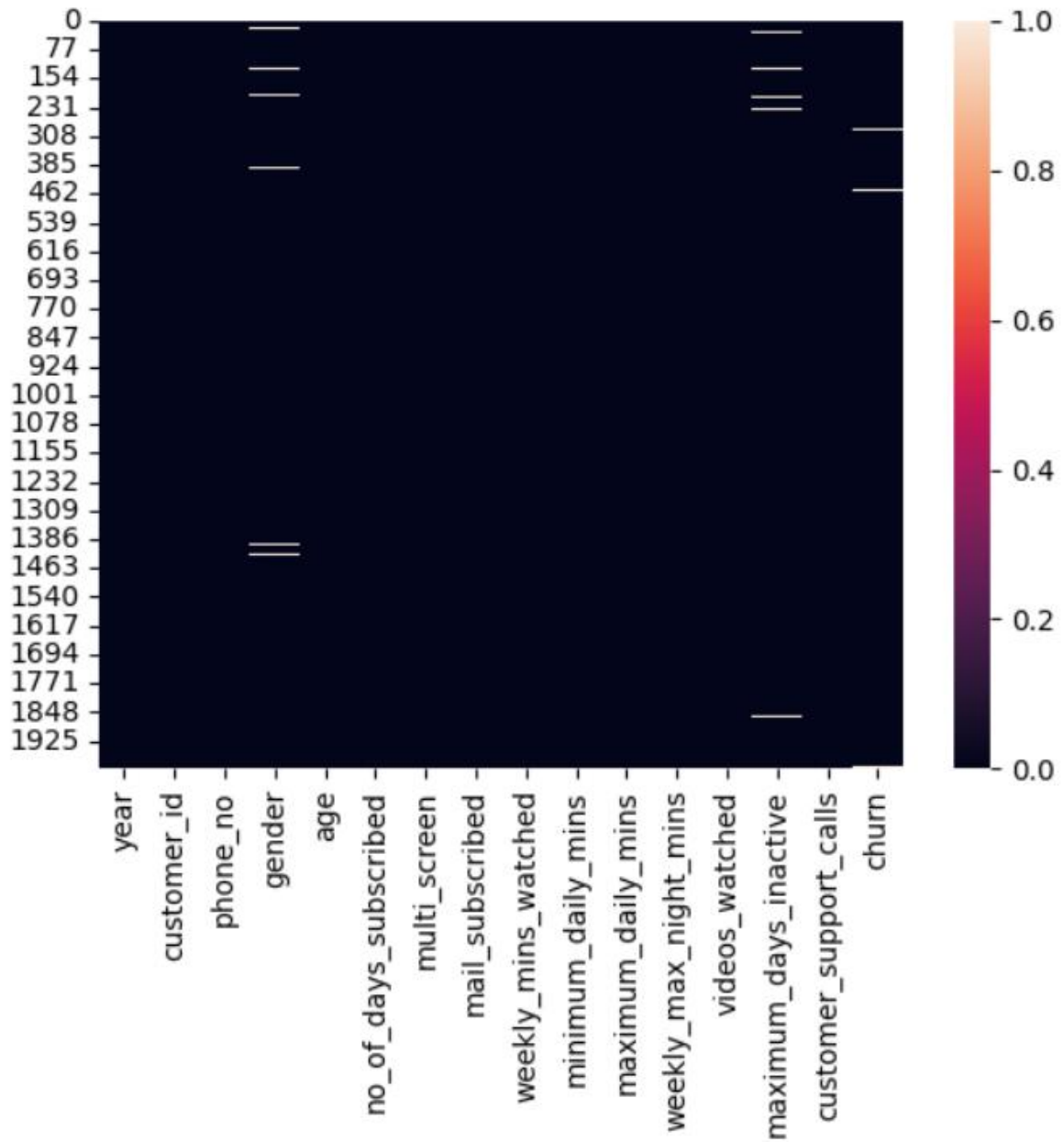
```
print(df.isnull().sum()):
```

Hiển thị tổng số giá trị bị thiếu cho từng cột. Điều này giúp xác định biến nào có nhiều giá trị bị thiếu để quyết định phương pháp xử lý phù hợp (ví dụ: loại bỏ hoặc thay thế giá trị bị thiếu).

```
sns.heatmap(dataframe.isnull()):
```

Sử dụng thư viện seaborn để tạo biểu đồ dạng heatmap, trực quan hóa vị trí các giá trị bị thiếu trong DataFrame. Những ô màu trên biểu đồ sẽ cho thấy rõ phần nào của dữ liệu bị thiếu.

map of missing values



8.Sử dụng thư viện statsmodel và sklearn để xây dựng mô hình(Using statsmodel and sklearn libraries to build the model)

```
# Chọn biến độc lập (X) và biến phụ thuộc (y)
X = data.drop('Churn', axis=1) y = data['Churn']

# Thêm hằng số vào mô hình hồi quy logistic với statsmodels
X = sm.add_constant(X) # Thêm hằng số cho hồi quy
model = sm.Logit(y, X).fit() # Xây dựng mô hình hồi quy logistic
print(model.summary()) # Hiển thị tóm tắt mô hình

def prepare_model(df,class_col,cols_to_exclude):
    ## Split in training and test set

    from sklearn.model_selection import train_test_split

    import numpy as np

    ##Selecting only the numerical columns and excluding the columns we specified in the
    function

    cols=df.select_dtypes(include=np.number).columns.tolist() X=df[cols]

    X = X[X.columns.difference([class_col])]

    X = X[X.columns.difference(cols_to_exclude)]

    ##Selecting y as a column

    y=df[class_col]

    global X_train, X_test, y_train, y_test #This allow us to do call these variables outside
    this function

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0) #
    perform train test split

    def run_model(X_train,X_test,y_train,y_test):

        from sklearn.linear_model import LogisticRegression

        from sklearn.metrics import roc_auc_score,classification_report

        global logreg #Defines the logistic model as a global model that can be used outside of
        this function

        ##Fitting the logistic regression

        logreg = LogisticRegression(random_state = 13) logreg.fit(X_train, y_train) # fit the
        model

        ##Predicting y values
```

```
global y_pred #Defines the Y_Pred as a global variable that can be used outside of this function
```

```
y_pred = logreg.predict(X_test) # make predictions on th test data
```

```
logit_roc_auc = roc_auc_score(y_test, logreg.predict(X_test))
```

```
print(classification_report(y_test, y_pred)) # check for classification report
```

```
print("The area under the curve is: %0.2f"%logit_roc_auc) # check for AUC
```

statsmodels giúp bạn hiểu rõ hơn về các thông số thống kê của mô hình (các hệ số, giá trị p, độ phù hợp).

sklearn dễ sử dụng và cung cấp các công cụ mạnh mẽ để đánh giá mô hình, phù hợp cho các quy trình machine learning thực tiễn như phân chia dữ liệu và tối ưu hóa.

9.Đào tạo mô hình bằng các kỹ thuật Phân loại như Hồi quy Logistics, Training a model using Classification techniques like Logistics Regression,

```
def logistic_regression(df,class_col,cols_to_exclude):
```

```
import statsmodels.api as sm
```

```
import numpy as np
```

```
cols=df.select_dtypes(include=np.number).columns.tolist()
```

```
X=df[cols]
```

```
X = X[X.columns.difference([class_col])]
```

```
X = X[X.columns.difference(cols_to_exclude)] # unwanted columns
```

```
## Scaling variables
```

```
##from sklearn import preprocessing
```

```
##scaler = preprocessing.StandardScaler().fit(X)
```

```
##X_scaled = scaler.transform(X)
```

```
#X_Scale = scaler.transform(X)
```

```
y=df[class_col] # the target variable
```

```
logit_model=sm.Logit(y,X)
```

```
result=logit_model.fit() # fit the model
```

```
print(result.summary2()) # check for summary
```

Hàm logistic_regression() xây dựng một mô hình hồi quy logistic từ dữ liệu số, loại bỏ các cột không cần thiết, sau đó huấn luyện mô hình. Cuối cùng, hàm sẽ in ra bản tóm

tất của mô hình, giúp bạn đánh giá hiệu suất và độ tin cậy của các biến dự đoán trong việc xác định biến mục tiêu (phân loại).

Optimization terminated successfully.

Current function value: 0.336585

Iterations 7

Results: Logit

Model:	Logit	Method:	MLE
Dependent Variable:	churn	Pseudo R-squared:	0.137
Date:	2024-10-17 16:48	AIC:	1315.1404
No. Observations:	1918	BIC:	1381.8488
Df Model:	11	Log-Likelihood:	-645.57
Df Residuals:	1906	LL-Null:	-748.02
Converged:	1.0000	LLR p-value:	7.1751e-38
No. Iterations:	7.0000	Scale:	1.0000

	Coef.	Std.Err.	z	P> z	[0.025	0.975]
age	-0.0208	0.0068	-3.0739	0.0021	-0.0340	-0.0075
customer_support_calls	0.4246	0.0505	8.4030	0.0000	0.3256	0.5237
gender_code	-0.2144	0.1446	-1.4824	0.1382	-0.4979	0.0691
mail_subscribed_code	-0.7529	0.1798	-4.1873	0.0000	-1.1053	-0.4005
maximum_daily_mins	-3.7125	25.2522	-0.1470	0.8831	-53.2058	45.7809
maximum_days_inactive	-0.7870	0.2473	-3.1828	0.0015	-1.2716	-0.3024
minimum_daily_mins	0.2075	0.0727	2.8555	0.0043	0.0651	0.3499
multi_screen_code	1.9511	0.1831	10.6562	0.0000	1.5923	2.3100
no_of_days_subscribed	-0.0045	0.0018	-2.5572	0.0106	-0.0080	-0.0011
videos_watched	-0.0948	0.0317	-2.9954	0.0027	-0.1569	-0.0328
weekly_max_night_mins	-0.0169	0.0032	-5.3119	0.0000	-0.0231	-0.0107
weekly_mins_watched	0.4248	2.8619	0.1484	0.8820	-5.1844	6.0340

10. Chia tập dữ liệu thành huấn luyện và kiểm tra bằng sklearn.(Splitting Dataset into Train and Test using sklearn.)

From sklearn.model_selection import train_test_split

Bước 1: Chuẩn bị dữ liệu

X = df[['feature1', 'feature2', 'feature3']] # Các biến độc lập (đặc trưng)

y = df['target'] # Biến mục tiêu (cần dự đoán)

Bước 2: Chia dữ liệu thành tập huấn luyện và kiểm thử

70% dữ liệu cho huấn luyện và 30% dữ liệu cho kiểm thử

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

In ra kích thước của các tập dữ liệu sau khi chia

print("Kích thước tập huấn luyện:", X_train.shape, y_train.shape)

print("Kích thước tập kiểm thử:", X_test.shape, y_test.shape)

- Tập huấn luyện: Dùng để huấn luyện mô hình, giúp mô hình học mối quan hệ giữa các biến đầu vào và biến mục tiêu.
- Tập kiểm thử: Được sử dụng để đánh giá mô hình trên dữ liệu mà nó chưa từng thấy trước đây. Điều này giúp xác định mức độ tổng quát hóa của mô hình và tránh overfitting (quá khớp).
- Việc chia dữ liệu thành các tập riêng biệt giúp bạn đảm bảo mô hình được kiểm tra đúng cách và cung cấp cái nhìn chính xác về hiệu suất thực tế khi áp dụng vào dữ liệu mới.

11. Dự đoán bằng cách sử dụng mô hình đã được đào tạo. Making predictions using the trained model.

```
y_pred = logreg.predict(X_test) # Dự đoán trên tập kiểm thử
```

```
[95] # save the model using pickle function
import pickle
pickle.dump(logreg, open('model1.pkl', 'wb'))
```

```
[116] # load the saved model
      model = pickle.load(open('model1.pkl', 'rb'))
```

```
# make predictions on the test data
model.predict(X_test)
```

[illegible]

Sau khi huấn luyện mô hình, bước tiếp theo là sử dụng mô hình để dự đoán và đánh giá hiệu suất. Dưới đây là quy trình tổng kết:

1. Dự đoán nhãn : Sử dụng `predict()` để dự đoán các lớp (0 hoặc 1) của dữ liệu đầu vào, như tập kiểm thử hoặc tập dữ liệu mới.

2. Dự đoán xác suất : Sử dụng ``predict_proba()`` để tính toán xác suất mỗi mẫu thuộc về từng lớp, cho phép phân tích sâu hơn về sự tự tin của mô hình.

3. Đánh giá hiệu suất :

- Độ chính xác (accuracy) : Đánh giá mô hình bằng cách so sánh dự đoán với kết quả thực tế.

- Ma trận nhầm lẫn : Giúp kiểm tra chi tiết số lượng dự đoán đúng và sai ở mỗi lớp, hỗ trợ phân tích độ chính xác của mô hình.

Sau khi huấn luyện mô hình, dự đoán và đánh giá kết quả là bước quan trọng để xác định hiệu quả của mô hình và mức độ phù hợp của nó đối với bài toán dự đoán.

12. Tăng sự tin tưởng vào mô hình bằng cách sử dụng các số liệu như điểm chính xác, ma trận nhầm lẫn, khả năng thu hồi, độ chính xác và điểm f1 (Gaining confidence in the model using metrics such as accuracy score, confusion matrix, recall, precision, and f1 score)

Các chỉ số đánh giá mô hình

1. Độ chính xác (Accuracy Score) :

- Tỷ lệ mẫu dự đoán đúng trên tổng số mẫu.
- Công thức : $\text{Độ chính xác} = \frac{\text{Số mẫu đúng}}{\text{Tổng số mẫu}}$

2. Ma trận nhầm lẫn (Confusion Matrix) :

- Là bảng thể hiện số lượng mẫu dự đoán đúng và sai cho từng lớp.

Gồm:

- TP (True Positives) : Số mẫu dương được dự đoán đúng.
- TN (True Negatives) : Số mẫu âm được dự đoán đúng.
- FP (False Positives) : Số mẫu âm bị dự đoán nhầm là dương.
- FN (False Negatives) : Số mẫu dương bị dự đoán nhầm là âm.

3. Độ chính xác (Precision) :

- Tỷ lệ mẫu dương được dự đoán đúng so với tổng số mẫu được dự đoán là dương.
- Công thức : $\text{Precision} = \frac{TP}{TP+FP}$

4. Độ nhạy (Recall) :

- Tỷ lệ mẫu dương được dự đoán đúng so với tất cả mẫu dương thực tế.

- Công thức : $\text{Recall} = \frac{TP}{TP+FN}$

5. F1 Score :

- Là trung bình hài hòa giữa độ chính xác và độ nhạy. Thích hợp khi bạn cần cân bằng giữa độ chính xác và độ nhạy.

- Công thức : $F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

```python

```
import numpy as np
```

```
import pandas as pd
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.ensemble import RandomForestClassifier
```

```

from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from imblearn.over_sampling import SMOTE

Tạo dữ liệu giả (để minh họa)
X = np.random.rand(1000, 20) # 1000 mẫu, 20 đặc trưng
y = np.array([0] * 900 + [1] * 100) # 900 mẫu lớp 0, 100 mẫu lớp 1

Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

Cân bằng dữ liệu bằng SMOTE
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X_train, y_train)

Huấn luyện mô hình Random Forest
model = RandomForestClassifier(random_state=42)
model.fit(X_resampled, y_resampled)

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

Tính toán và in ra các chỉ số
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

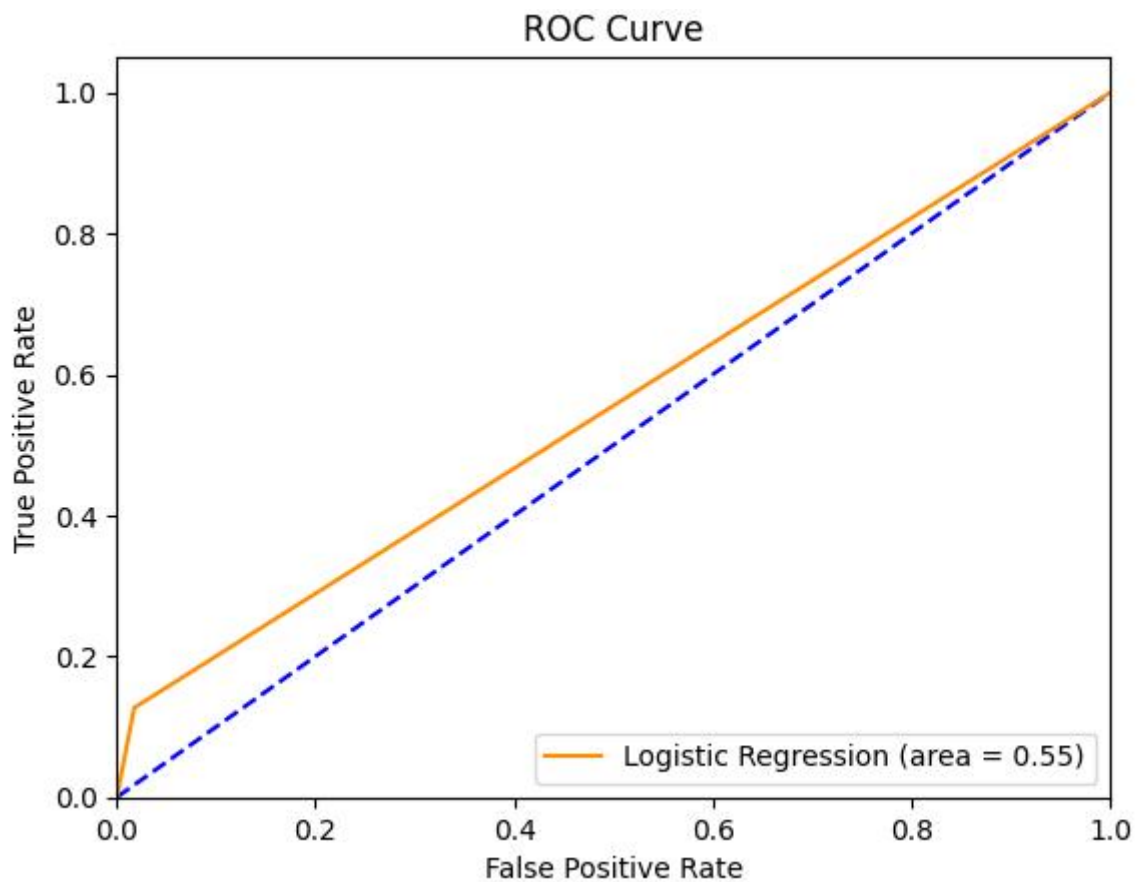
print(f'Độ chính xác: {accuracy:.2f}')
print("Ma trận nhầm lẫn:")
print(conf_matrix)
print("Báo cáo phân loại:")
print(class_report)

```

1. Tạo dữ liệu giả : Tạo ra một tập dữ liệu ngẫu nhiên để sử dụng cho ví dụ này.
2. Chia dữ liệu : Dữ liệu được chia thành tập huấn luyện (80%) và tập kiểm tra (20%).
3. Cân bằng dữ liệu : Sử dụng SMOTE để tạo ra các mẫu giả cho lớp thiểu số (lớp 1), giúp cân bằng số lượng mẫu giữa các lớp.
4. Huấn luyện mô hình : Sử dụng mô hình Random Forest để huấn luyện trên tập huấn luyện đã được cân bằng.
5. Dự đoán : Mô hình đưa ra dự đoán trên tập kiểm tra.

## 6. Tính toán các chỉ số :

- Độ chính xác : Tính toán tỷ lệ mẫu dự đoán đúng.
- Ma trận nhầm lẫn : Hiển thị số lượng mẫu đúng và sai cho từng lớp.
- Báo cáo phân loại : Cung cấp thông tin chi tiết về precision, recall và F1 score.
- Độ chính xác : Thể hiện hiệu suất tổng thể của mô hình. Ví dụ, nếu độ chính xác là 0.85, điều đó có nghĩa là 85% mẫu được dự đoán đúng.
- Ma trận nhầm lẫn : Cho biết mô hình đã dự đoán đúng bao nhiêu mẫu cho từng lớp và số mẫu bị dự đoán sai.
- Precision và Recall : Giúp hiểu rõ hơn về khả năng dự đoán mẫu dương của mô hình.
  - Precision cao cho thấy ít mẫu âm bị dự đoán nhầm là dương.
  - Recall cao cho thấy hầu hết mẫu dương thực tế được phát hiện.
- F1 Score : Là một chỉ số duy nhất kết hợp giữa precision và recall, hữu ích khi dữ liệu không cân bằng.



### 13.Xử lý dữ liệu không cân bằng bằng nhiều phương pháp khác nhau.(Handling the unbalanced data using various methods.)

Tăng cường dữ liệu (Data Augmentation): Tạo mẫu mới từ lớp thiểu số (ví dụ: SMOTE, ADASYN).

Giảm số lượng mẫu (Under-sampling): Giảm mẫu từ lớp chiếm ưu thế (ví dụ: Random Under-sampling, Cluster-based Under-sampling).

Kỹ thuật cân bằng lớp (Class Weighting): Gán trọng số cho lớp thiểu số trong hàm mất mát.

Sử dụng mô hình học sâu (Deep Learning): Sử dụng các mô hình có khả năng tự động điều chỉnh với dữ liệu không cân bằng.

Chọn mô hình thích hợp: Sử dụng các mô hình như cây quyết định, Random Forest hoặc Gradient Boosting.

Kỹ thuật ensemble: Kết hợp nhiều mô hình để cải thiện dự đoán (ví dụ: Bagging, Boosting).

Phân tích đặc trưng (Feature Analysis): Tối ưu hóa các đặc trưng đầu vào.

Đánh giá mô hình: Sử dụng các thước đo phù hợp như AUC-ROC, độ chính xác, độ nhạy.

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
from imblearn.over_sampling import SMOTE
Giả định dữ liệu là một DataFrame
df = pd.read_csv('duong_dan_toi_du_lieu.csv')
Tạo dữ liệu giả
X = np.random.rand(1000, 20) # 1000 mẫu, 20 đặc trưng
y = np.array([0] * 900 + [1] * 100) # 900 mẫu lớp 0, 100 mẫu lớp 1
Chia dữ liệu thành tập huấn luyện và kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
Áp dụng SMOTE
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X_train, y_train)

Huấn luyện mô hình Random Forest
model = RandomForestClassifier(random_state=42)
model.fit(X_resampled, y_resampled)

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

Đánh giá mô hình
print(classification_report(y_test, y_pred))
```

## 14. Thực hiện lựa chọn tính năng bằng nhiều phương pháp (Performing feature selection with multiple methods)

### Tổng quan về Chọn Lọc Đặc Trưng

Chọn lọc đặc trưng là bước quan trọng trong quy trình học máy, nhằm chọn ra một tập hợp các đặc trưng (features) có liên quan nhất để xây dựng mô hình. Việc này có thể cải thiện hiệu suất của mô hình, giảm tình trạng quá khớp (overfitting), và rút ngắn thời gian huấn luyện. Dưới đây là các phương pháp chọn lọc đặc trưng phổ biến cùng với ví dụ mã Python cho từng phương pháp.

### Phương pháp Lọc (Filter Methods)

Các phương pháp lọc đánh giá tính liên quan của các đặc trưng bằng cách xác định mối tương quan với biến mục tiêu, độc lập với các thuật toán học máy.

- Ví dụ: Hệ số tương quan

```
`python
import pandas as pd

Tạo dữ liệu giả
data = pd.DataFrame({
 'feature1': [1, 2, 3, 4, 5],
 'feature2': [5, 4, 3, 2, 1],
 'target': [1, 1, 0, 0, 1]
})
```

# Tính toán hệ số tương quan

```
correlation = data.corr()
```

```
print(correlation['target'].sort_values(ascending=False))
```

Phương pháp Bọc (Wrapper Methods)

Các phương pháp bọc đánh giá các tập hợp đặc trưng dựa trên hiệu suất của mô hình. Chúng sử dụng mô hình dự đoán để đánh giá các kết hợp khác nhau của các đặc trưng.

- Ví dụ: Loại bỏ Đặc Trưng Đệ Quy (Recursive Feature Elimination - RFE)

```
```python
```

```
from sklearn.feature_selection import RFE
```

```
from sklearn.linear_model import LogisticRegression
```

```
from sklearn.datasets import make_classification
```

```
# Tạo dữ liệu
```

```
X, y = make_classification(n_samples=100, n_features=10, n_informative=5,
random_state=42)
```

```
# Khởi tạo mô hình
```

```
model = LogisticRegression()
```

```
# Khởi động RFE
```

```
rfe = RFE(model, 5) # Chọn 5 đặc trưng quan trọng nhất
```

```
fit = rfe.fit(X, y)
```

```
# In ra các đặc trưng đã chọn
```

```
print("Các đặc trưng đã chọn: ", fit.support_)
```

```
print("Vị trí các đặc trưng đã chọn: ", fit.ranking_)
```

Phương pháp Nhúng (Embedded Methods)

Các phương pháp nhúng thực hiện việc chọn lọc đặc trưng trong quá trình huấn luyện mô hình. Những phương pháp này thường hiệu quả hơn so với các phương pháp bọc.

- Ví dụ: Hồi quy Lasso

```
```python
```

```
from sklearn.linear_model import Lasso
```

```
from sklearn.datasets import make_regression
```

```
Tạo dữ liệu
```

```
X, y = make_regression(n_samples=100, n_features=10, noise=0.1, random_state=42)
```

# Khởi tạo và huấn luyện mô hình Lasso

```
lasso = Lasso(alpha=0.1)
```

```
lasso.fit(X, y)
```

# In ra các trọng số và xác định các đặc trưng quan trọng

```
print("Trọng số của các đặc trưng: ", lasso.coef_)
```

```
important_features = [i for i in range(len(lasso.coef_)) if lasso.coef_[i] != 0]
```

```
print("Các đặc trưng quan trọng: ", important_features)
```

### Phương pháp Dựa trên Cây (Tree-Based Methods)

Các phương pháp dựa trên cây như Random Forest cung cấp điểm quan trọng cho từng đặc trưng, điều này có thể hữu ích cho việc chọn lọc đặc trưng.

- Ví dụ: Độ Quan Trọng của Các Đặc Trưng trong Random Forest

```
from sklearn.ensemble import RandomForestClassifier
```

# Tạo dữ liệu

```
X, y = make_classification(n_samples=100, n_features=10, n_informative=5,
random_state=42)
```

# Khởi tạo và huấn luyện mô hình Random Forest

```
rf = RandomForestClassifier()
```

```
rf.fit(X, y)
```

# Lấy độ quan trọng của các đặc trưng

```
importances = rf.feature_importances_
```

# In ra độ quan trọng

```
print("Độ quan trọng của các đặc trưng: ", importances)
```

```
important_features = [i for i in range(len(importances)) if importances[i] > 0.1] # Chọn
các đặc trưng có độ quan trọng > 0.1
```

```
print("Các đặc trưng quan trọng: ", important_features)
```

### Phương pháp Kết hợp (Hybrid Methods)

Phương pháp kết hợp kết hợp ưu điểm của các phương pháp lọc và bọc để nâng cao hiệu quả chọn lọc đặc trưng.

- Ví dụ: Sử dụng Phương Pháp Lọc để Chọn Đặc Trưng Trước, Sau Đó Sử Dụng RFE

```
from sklearn.feature_selection import SelectKBest, chi2
```

# Tạo dữ liệu

```
X, y = make_classification(n_samples=100, n_features=10, n_informative=5,
random_state=42)
```

# Lựa chọn 5 đặc trưng tốt nhất dựa trên thống kê chi-squared

```
X_new = SelectKBest(chi2, k=5).fit_transform(X, y)
```

# Sử dụng RFE trên tập dữ liệu đã giảm số lượng đặc trưng

```
rfe = RFE(model, 3) # Chọn 3 đặc trưng quan trọng nhất
```

```
fit = rfe.fit(X_new, y)
```

```
print("Các đặc trưng đã chọn sau khi RFE: ", fit.support_)
```

Việc chọn lọc đặc trưng là bước quan trọng trong quy trình phát triển mô hình học máy. Bằng cách sử dụng các phương pháp như lọc, bọc, nhúng, và dựa trên cây, bạn có thể xác định các đặc trưng quan trọng nhất cho mô hình của mình. Điều này không chỉ cải thiện hiệu suất của mô hình mà còn giảm thiểu độ phức tạp và thời gian xử lý. Nếu bạn cần thêm thông tin hoặc có câu hỏi, đừng ngần ngại hỏi nhé!

## 15. Lưu mô hình tốt nhất ở định dạng pickle để sử dụng trong tương lai. Saving the best model in pickle format for future use.

Khi phát triển một mô hình học máy, việc lưu trữ mô hình tốt nhất là rất quan trọng để có thể sử dụng lại mà không cần phải huấn luyện lại. Định dạng pickle cho phép bạn lưu trữ đối tượng Python, bao gồm cả mô hình học máy, vào tệp và tải lại sau này. Điều này giúp tiết kiệm thời gian và tài nguyên trong quá trình phát triển.

# save the model using pickle function

```
import pickle
```

```
pickle.dump(logreg, open('model1.pkl', 'wb'))
```

# load the saved model

```
model = pickle.load(open('model1.pkl', 'rb'))
```

# make predictions on the test data

```
model.predict(X_test)
```

## Tổng Kết

Tầm quan trọng của việc duy trì khách hàng

- Chi phí thấp hơn : Giữ chân khách hàng hiện tại thường ít tốn kém hơn so với việc thu hút khách hàng mới.
- Tăng doanh thu : Khách hàng trung thành có khả năng chi tiêu nhiều hơn và thường giới thiệu cho bạn bè và người thân.
- Phản hồi tích cực : Khách hàng trung thành thường sẽ cung cấp phản hồi và đánh giá tốt, góp phần xây dựng thương hiệu mạnh mẽ.

. Nguyên nhân chính dẫn đến Customer Churn

- Chất lượng dịch vụ kém : Sự thiếu sót trong cung cấp dịch vụ hoặc sản phẩm không đáp ứng mong đợi của khách hàng.
- Giá cả : Giá cả không hợp lý hoặc không cạnh tranh so với đối thủ.
- Thiếu sự tương tác : Khách hàng cảm thấy bị bỏ rơi hoặc không được quan tâm.
- Sự phát triển của đối thủ : Các công ty khác có thể cung cấp sản phẩm/dịch vụ tốt hơn hoặc các ưu đãi hấp dẫn hơn.

Chiến lược ngăn chặn Customer Churn

- Cải thiện chất lượng sản phẩm/dịch vụ : Đảm bảo rằng sản phẩm hoặc dịch vụ luôn đáp ứng hoặc vượt quá mong đợi của khách hàng.
- Chương trình chăm sóc khách hàng : Thiết lập một hệ thống chăm sóc khách hàng chuyên nghiệp, lắng nghe và phản hồi kịp thời các thắc mắc, khiếu nại của khách hàng.
- Cá nhân hóa trải nghiệm khách hàng : Sử dụng dữ liệu để hiểu và phục vụ nhu cầu của khách hàng một cách tốt nhất, từ đó tạo ra trải nghiệm cá nhân hóa.
- Phân tích dữ liệu và dự đoán rủi ro : Sử dụng các công cụ phân tích dữ liệu để nhận diện các dấu hiệu cho thấy khách hàng có thể rời bỏ dịch vụ, từ đó có biện pháp can thiệp kịp thời.
- Cung cấp các ưu đãi và khuyến mãi : Đưa ra các chương trình khuyến mãi hoặc ưu đãi cho khách hàng trung thành, nhằm tạo động lực giữ chân họ.
- Tạo cộng đồng và sự kết nối : Xây dựng một cộng đồng xung quanh thương hiệu, nơi khách hàng có thể giao lưu và kết nối với nhau.

Theo dõi và đánh giá

- Sử dụng chỉ số đo lường : Theo dõi các chỉ số như tỷ lệ rời bỏ (churn rate), tỷ lệ giữ chân khách hàng (retention rate), và mức độ hài lòng của khách hàng (NPS) để đánh giá hiệu quả của các chiến lược.
- Khảo sát khách hàng : Thực hiện khảo sát định kỳ để lấy ý kiến phản hồi từ khách hàng về sản phẩm, dịch vụ và trải nghiệm của họ.

## Kết luận

Trong bối cảnh cạnh tranh gay gắt hiện nay, việc duy trì khách hàng và ngăn chặn tình trạng rời bỏ dịch vụ (Customer Churn) đã trở thành một yếu tố sống còn đối với các doanh nghiệp. Khách hàng rời bỏ không chỉ làm giảm doanh thu mà còn ảnh hưởng đến hình ảnh thương hiệu. Một trong những nguyên nhân chính dẫn đến tình trạng này là sự không hài lòng với chất lượng sản phẩm hoặc dịch vụ, cũng như sự thiếu sự quan tâm từ phía doanh nghiệp. Để giữ chân khách hàng, các công ty cần cải thiện chất lượng dịch vụ, thiết lập chương trình chăm sóc khách hàng hiệu quả, và cá nhân hóa trải nghiệm của họ. Bên cạnh đó, việc phân tích dữ liệu khách hàng để dự đoán những khách hàng có nguy cơ rời bỏ và đưa ra các ưu đãi hấp dẫn cũng là những chiến lược quan trọng. Thực hiện các biện pháp này không chỉ giúp giảm tỷ lệ rời bỏ mà còn gia tăng sự trung thành của khách hàng, từ đó thúc đẩy doanh thu và xây dựng thương hiệu bền vững trong thị trường cạnh tranh.