TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

PHÂN HIỆU TẠI TP. HỒ CHÍ MINH

BỘ MÔN CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

HỌC PHẦN KHAI PHÁ DỮ LIỆU

ĐỀ TÀI: DỰ ĐOÁN KHẢ NĂNG RỜI BỎ ỨNG DỤNG SPOTIFY CỦA NGƯỜI DÙNG

Giảng viên hướng dẫn: Th.S NGUYỄN THIỆN DƯƠNG

Sinh viên thực hiện: HỒ THÀNH ĐẠT - 6351071017

NGUYỄN TRẦN THANH DANH - 6351071010

TRẦN MINH HIẾU - 6351071023

ĐINH QUỐC BẢO - 6351071006

Lớp : Công nghệ thông tin

Khoá : 63

Tp. Hồ Chí Minh, năm 2025

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

PHÂN HIỆU TẠI TP. HỒ CHÍ MINH

BỘ MÔN CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

HỌC PHẦN KHAI PHÁ DỮ LIỆU

ĐỀ TÀI: DỰ ĐOÁN KHẢ NĂNG RỜI BỎ ỨNG DỤNG SPOTIFY CỦA NGƯỜI DÙNG

Giảng viên hướng dẫn: Th.S NGUYỄN THIỆN DƯƠNG

Sinh viên thực hiện: HỒ THÀNH ĐẠT - 6351071017

NGUYỄN TRẦN THANH DANH - 6351071010

TRẦN MINH HIẾU - 6351071023

ĐINH QUỐC BẢO - 6351071006

Lớp : Công nghệ thông tin

Khoá : 63

Tp. Hồ Chí Minh, năm 2025

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

NHIỆM VỤ BÁO CÁO BÀI TẬP LỚN

BỘ MÔN: **CÔNG NGHỆ THÔNG TIN**

-------***-------

Sinh viên thực hiện: Hồ Thành Đạt - Mã số sinh viên: 6351071017
Nguyễn Trần Thanh Danh - Mã số sinh viên: 6351071010
Trần Minh Hiếu - Mã số sinh viên: 6351071023
Đinh Quốc Bảo - Mã số sinh viên: 6351071006

1. Tên đề tài

Dự đoán khả năng rời bỏ ứng dụng Spotify của người dùng

2. Lý do chọn đề tài

Nhóm chọn đề tài "Dự đoán khả năng rời bỏ ứng dụng Spotify của người dùng" vì bài toán churn đang rất phổ biến trong các nền tảng nghe nhạc trực tuyến như Spotify. Với hơn 600 triệu người dùng, Spotify gặp khó khăn trong việc giữ chân khách hàng, đặc biệt ở thị trường Việt Nam nơi tỷ lệ rời bỏ cao do cạnh tranh từ các app khác. Việc dự đoán sớm giúp doanh nghiệp tối ưu chiến lược marketing và cá nhân hóa dịch vụ, giảm mất mát doanh thu.

Về mặt học thuật, đề tài cho phép áp dụng kiến thức về khai phá dữ liệu và học máy, từ EDA, tiền xử lý dữ liệu mất cân bằng đến so sánh các mô hình như Random Forest, Gradient Boosting và Logistic Regression. Qua đó, nhóm học được cách đánh giá mô hình bằng ROC-AUC, F1-Score và phân tích feature importance để hiểu yếu tố ảnh hưởng đến churn, như skip rate hay loại tài khoản.

Đề tài sử dụng bộ dữ liệu công khai từ Kaggle (8.000 mẫu, 11 đặc trưng), dễ tái tạo mà không cần dữ liệu thực tế. Kết quả là mô hình dự đoán mà còn web demo đơn giản, giúp nhóm rèn luyện kỹ năng từ code đến triển khai.

3. Nội dung, phạm vi đề tài

3.1 Nội dung đề tài

- Đề tài xây dựng mô hình học máy dự đoán churn người dùng Spotify dựa trên quy trình KDD:
- Tiền xử lý: Làm sạch dữ liệu từ dataset Kaggle (8.000 mẫu, 11 đặc trưng), mã hóa categorical, xử lý mất cân bằng bằng SMOTE, chuẩn hóa numerical.
- EDA: Phân tích thống kê (Pandas), biểu đồ phân phối/histogram/pie chart (Matplotlib), ma trận tương quan.
- Mô hình: Pipeline với Logistic Regression, Random Forest, Gradient Boosting. Huấn luyện trên Colab, chia train/test 80/20 (stratify=y).
- Đánh giá: Accuracy, F1-Score, ROC-AUC; phân tích confusion matrix và feature importance.
- Triển khai: API FastAPI, web demo React + Tailwind.

3.2 Phạm vi đề tài

Đề tài giới hạn ở dữ liệu tĩnh từ bộ dataset công khai, không thu thập dữ liệu thực tế từ Spotify. Chỉ áp dụng các mô hình học máy cơ bản (không deep learning), tập trung vào phân loại nhị phân churn (0/1) mà không phân tích yếu tố bên ngoài như marketing. Phạm vi kỹ thuật: Huấn luyện offline trên Colab, triển khai API đơn giản, web demo responsive nhưng chưa tích hợp production (không database). Kết quả đánh giá trên tập test, không cross-validation sâu. Đề tài không mở rộng sang các nền tảng khác ngoài Spotify, nhằm tập trung vào case study cụ thể.

4. Công nghệ, công cụ và ngôn ngữ lập trình

Để thực hiện đề tài, nhóm sử dụng các ngôn ngữ và công cụ phù hợp với từng giai đoạn từ xử lý dữ liệu, xây dựng mô hình đến triển khai ứng dụng. Việc lựa chọn dựa trên tính phổ biến, hiệu quả và dễ tích hợp, đảm bảo quy trình phát triển mượt mà.

4.1. Ngôn ngữ lập trình

Python: Là ngôn ngữ chính cho phần backend và phân tích dữ liệu. Python được chọn nhờ hệ sinh thái thư viện phong phú hỗ trợ khoa học dữ liệu và học máy. Các thư viện như Pandas (xử lý dữ liệu), Scikit-learn (xây dựng mô hình), và Joblib (lưu/truy xuất mô hình) giúp nhóm thực hiện tiền xử lý, huấn luyện và đánh giá nhanh chóng. Ví dụ, pipeline với ColumnTransformer và SMOTE được viết bằng Python để xử lý dữ liệu mất cân bằng.

JavaScript: Sử dụng cho frontend web demo. Với React.js làm framework chính, JavaScript xử lý state management và tương tác người dùng, như form input và gọi API. Ngôn ngữ này đảm bảo giao diện responsive và mượt mà trên các thiết bị.

4.2. Framework và thư viện

Backend:

- FastAPI: Framework web nhẹ, nhanh để xây dựng API RESTful. Nhóm dùng FastAPI để tạo endpoint /predict_churn với Pydantic cho validation input (ChurnRequest model), và Uvicorn làm server (chạy trên port 8000). Điều này cho phép deploy mô hình dễ dàng mà không phức tạp như Flask.
- Scikit-learn: Thư viện cốt lõi cho học máy, hỗ trợ các mô hình Logistic Regression, Random Forest (n_estimators=200, max_depth=6), Gradient Boosting. Kết hợp Imbalanced-learn cho SMOTE để xử lý churn imbalanced (26% churn samples).

Frontend:

- React.js (18+): Xây dựng giao diện single-page application với components modular (Header, PersonalInfo, UsageActivity). Vite làm bundler cho hot reload nhanh trong dev.
- Tailwind CSS: Utility-first CSS framework cho styling responsive, dark mode, và theme Spotify-inspired. Lucide-react cung cấp icons (Sun/Moon cho toggle mode).

5. Các kết quả chính dự kiến sẽ đạt được:

Đề tài dự kiến đạt được các kết quả sau:

Mô hình dự đoán churn hiệu quả: Xây dựng và so sánh ba mô hình học máy (Logistic Regression, Random Forest, Gradient Boosting) với hiệu suất ROC-AUC >0.85 trên tập test. Mô hình tốt nhất (dự kiến Random Forest hoặc Gradient Boosting) sẽ dự đoán chính xác khả năng rời bỏ dựa trên đặc trưng hành vi như skip_rate và ads_listened_per_week.

Phân tích và diễn giải dữ liệu: Qua EDA và feature importance, xác định các yếu tố chính ảnh hưởng đến churn (ví dụ: người dùng Free ở quốc gia AU có nguy cơ cao hơn). Xử lý thành công dữ liệu mất cân bằng bằng SMOTE, giảm false negative để tránh bỏ lỡ người dùng rủi ro.

Hệ thống triển khai thực tiễn: Phát triển API backend với FastAPI để phục vụ dự đoán real-time, và web demo frontend React tích hợp gọi API. Người dùng có thể nhập dữ liệu và nhận kết quả churn probability/label ngay lập tức.

6. Giảng viên và cán bộ hướng dẫn:

Họ và tên: Nguyễn Thiện Dương

Đơn vị công tác: Bộ môn CNTT

Điện thoại: Email:

7. Đã nhận nhiệm vụ

Sinh viên: Hồ Thành Đạt

Nguyễn Trần Thanh Danh

Trần Minh Hiếu

Đinh Quốc Bảo

Điện thoại: 0336107518

Email: 6351071006@st.utc2.edu.vn

Ngày 13 tháng 12 năm 2025	**Đã giao nhiệm vụ TKTN**
Trưởng BM Công nghệ Thông tin	**Giảng viên hướng dẫn**

LỜI CẢM ƠN

Để hoàn thành đề tài này, trước tiên nhóm chúng em xin gửi lời tri ân sâu sắc đến quý thầy cô trong Bộ môn Công nghệ Thông tin – Phân hiệu Trường Đại học Giao thông Vận tải tại Thành phố Hồ Chí Minh. Chúng em vô cùng biết ơn vì những kiến thức chuyên môn vững chắc cùng những bài học thực tiễn quý báu mà thầy cô đã truyền đạt trong suốt quá trình học tập.

Đặc biệt, chúng em xin bày tỏ lòng biết ơn chân thành đến thầy Nguyễn Thiện Dương, giảng viên hướng dẫn môn khai phá dữ liệu, người đã tận tâm định hướng, góp ý và hỗ trợ chúng em trong suốt quá trình thực hiện báo cáo. Dù thời gian thực hiện đồ án môn học có hạn, nhưng sự tận tình và những góp ý quý giá của thầy đã giúp chúng em vượt qua nhiều khó khăn để hoàn thành tốt nhiệm vụ.

Do thời gian thực hiện còn hạn chế, cùng với kiến thức và kinh nghiệm của nhóm chúng em vẫn còn nhiều giới hạn, chúng em đã cố gắng hết mình để hoàn thiện báo cáo bài tập lớn này một cách tốt nhất. Tuy vậy, những thiếu sót là khó tránh khỏi. Chúng em rất mong nhận được sự lượng thứ và những ý kiến đóng góp từ quý thầy cô để báo cáo được hoàn thiện hơn.

Cuối cùng, nhóm chúng em xin kính chúc quý thầy cô trong Bộ môn Công nghệ Thông tin luôn dồi dào sức khỏe, tràn đầy niềm vui và đạt nhiều thành công trong sự nghiệp giảng dạy cũng như trong cuộc sống.

Chúng em xin chân thành cảm ơn!

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp. Hồ Chí Minh, ngày …… tháng …… năm ……

Giáo viên hướng dẫn

Nguyễn Thiện Dương

MỤC LỤC

DANH MỤC CHỮ VIẾT TẮT

STT	Từ viết tắt	Tên đầy đủ	Ý nghĩa
1	AI	Artificial Intelligence	Trí tuệ nhân tạo – mô phỏng khả năng học và tư duy của con người bằng máy tính.
2	AUC	Area Under the Curve	Diện tích dưới đường cong (thường đi kèm với ROC)
3	EDA	Exploratory Data Analysis	Phân tích dữ liệu khám phá
4	KDD	Knowledge Discovery in Databases	Khám phá tri thức trong cơ sở dữ liệu
5	ML	Machine Learning	Học máy
6	ROC	Receiver Operating Characteristic	Đường cong đặc tính hoạt động của bộ thu
7	SMOTE	Synthetic Minority Over-sampling Technique	Kỹ thuật sinh mẫu thiểu số nhân tạo (Xử lý dữ liệu mất cân bằng)
8	SVM	Support Vector Machine	Máy vector hỗ trợ (Thuật toán học máy)
9	XGBoost	eXtreme Gradient Boosting	Tăng cường Gradient cực đại (Thuật toán học máy tối ưu)
10	BI	Business Intelligence	Nghiệp vụ thông minh

11	GBM	Gradient Boosting Machine	Máy tăng cường Gradient (Thuật toán học máy)
12	GPU	Graphics Processing Unit	Đơn vị xử lý đồ họa (Dùng để tăng tốc tính toán)

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

BẢNG PHÂN CÔNG CÔNG VIỆC

Thành viên	Khối lượng công việc	Phân công	Chữ ký
Hồ Thành Đạt	28%	- Chọn dataset - Tiền xử lý và khai phá dữ liệu - Xây dựng mô hình dự đoán	
Trần Minh Hiếu	24%	- Xây dựng Giao diện.	
Nguyễn Trần Thanh Danh	24%	- Xây dựng Backend	
Đinh Quốc Bảo	24%	- Viết báo cáo.	

CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI

1.1. Giới thiệu đề tài

Trong bối cảnh nền kinh tế số phát triển mạnh mẽ và mức độ cạnh tranh giữa các nền tảng nghe nhạc trực tuyến ngày càng khốc liệt, việc duy trì người dùng trở thành yếu tố sống còn đối với các doanh nghiệp cung cấp dịch vụ số. Đặc biệt đối với các quốc gia có lượng người dùng trẻ và năng động như Việt Nam, hành vi sử dụng ứng dụng âm nhạc thay đổi liên tục, dẫn đến tỷ lệ rời bỏ dịch vụ (churn) ngày càng khó dự đoán nếu chỉ dựa vào phương pháp thống kê truyền thống. Sự phát triển của trí tuệ nhân tạo và khai phá dữ liệu (Data Mining) đã mở ra những hướng tiếp cận mới, cho phép phân tích hành vi người dùng ở mức độ sâu hơn và phát hiện các dấu hiệu rủi ro rời bỏ ngay trước khi xảy ra.

Tuy nhiên, thách thức lớn trong việc xây dựng hệ thống dự đoán churn là khả năng xử lý khối lượng dữ liệu hành vi rất lớn và đa dạng của người dùng, bao gồm lịch sử nghe nhạc, tần suất tương tác, thói quen sử dụng tính năng cũng như đặc điểm nhân khẩu học. Việc chọn lọc đặc trưng quan trọng và mô hình hóa chúng một cách hiệu quả đóng vai trò then chốt trong quá trình dự đoán.

Để giải quyết bài toán này, đề tài "Dự đoán khả năng rời bỏ ứng dụng Spotify của người dùng" được thực hiện nhằm xây dựng một mô hình học máy có khả năng phân tích dữ liệu hành vi và nhận diện các yếu tố dẫn đến nguy cơ rời bỏ. Hệ thống được thiết kế dựa trên quy trình khai phá dữ liệu tiêu chuẩn, kết hợp giữa các kỹ thuật xử lý dữ liệu, trích xuất đặc trưng và các thuật toán học máy như Random Forest, Gradient Boosting và XGBoost. Mô hình không chỉ dự đoán khả năng rời bỏ mà còn giúp giải thích các yếu tố ảnh hưởng mạnh nhất đến hành vi churn của người dùng.

Ứng dụng của mô hình này mang lại nhiều lợi ích thiết thực cho các nền tảng âm nhạc trực tuyến: hỗ trợ doanh nghiệp nhận biết sớm nhóm người dùng có nguy cơ cao, tối ưu hóa chiến dịch giữ chân (retention), cá nhân hóa đề xuất nội dung và cải thiện chất lượng dịch vụ. Quan trọng hơn, đề tài minh họa hiệu quả việc kết hợp giữa kỹ thuật phân tích dữ liệu hiện đại và các mô hình học máy tiên tiến để giải quyết một bài toán thực tiễn trong lĩnh vực kinh doanh số.

1.2. Mục tiêu đề tài

Xây dựng mô hình học máy để dự đoán tỷ lệ rời bỏ của người dùng, phân tích các mô hình tương tác và tìm ra các nguyên nhân dẫn đến số lượng ngừng sử dụng dịch vụ trên nền tảng nghe nhạc trực tuyến Spotify. Từ đó học hỏi và xem đề tài dưới dạng một case study mở rộng để giải quyết bài toán chiến lược giúp giữ chân người dùng cho các ứng dụng trực tuyến tương tự.

Bên cạnh mục tiêu chính, đề tài còn hướng đến việc xây dựng một quy trình phân tích dữ liệu toàn diện, bao gồm tiền xử lý dữ liệu, lựa chọn đặc trưng, đánh giá mô hình và diễn giải kết quả dự đoán. Thông qua việc áp dụng nhiều thuật toán học máy khác nhau, đề tài nhằm tìm ra mô hình tối ưu có khả năng dự đoán chính xác nhóm người dùng có nguy cơ rời bỏ cao.

Ngoài ra, đề tài còn tập trung phân tích mức độ ảnh hưởng của từng đặc trưng hành vi như tần suất nghe nhạc, tỷ lệ bỏ qua bài hát, thời gian hoạt động hằng ngày hay quốc gia sử dụng đến khả năng churn của người dùng. Việc hiểu rõ các yếu tố này không chỉ giúp mô hình trở nên minh bạch hơn mà còn cung cấp các gợi ý chiến lược cho doanh nghiệp trong việc cải thiện trải nghiệm người dùng và đưa ra các biện pháp can thiệp kịp thời.

Cuối cùng, thông qua quá trình thực hiện, đề tài mong muốn xây dựng một khung tham chiếu (framework) có thể áp dụng linh hoạt cho các nền tảng trực tuyến khác có nhu cầu phân tích hành vi và dự đoán churn, góp phần nâng cao hiệu quả vận hành và tối ưu hóa chiến lược giữ chân người dùng trong môi trường dịch vụ số.

1.3. Mục đích nghiên cứu

Về mặt ứng dụng thực tiễn:

Xây dựng một mô hình học máy hỗ trợ phân tích và dự đoán khả năng rời bỏ ứng dụng Spotify của người dùng dựa trên dữ liệu hành vi thực tế. Hệ thống giúp doanh nghiệp nhận diện sớm nhóm người dùng có nguy cơ cao, từ đó tối ưu hóa chiến lược giữ chân và cải thiện trải nghiệm người dùng.

Hỗ trợ các nền tảng âm nhạc trực tuyến đưa ra quyết định dựa trên dữ liệu (data-driven), chẳng hạn như cá nhân hóa nội dung, tối ưu hệ thống gợi ý, hoặc triển khai các chương trình ưu đãi phù hợp để giảm thiểu churn.

Đề tài góp phần làm cơ sở tham khảo cho các mô hình phân tích hành vi người dùng trong những dịch vụ trực tuyến tương tự như Netflix, YouTube Music hay Apple Music.

Về mặt khoa học – kỹ thuật:

Khám phá và so sánh hiệu quả của các thuật toán học máy như Random Forest, Gradient Boosting, XGBoost và Logistic Regression trong bài toán dự đoán churn.

Đánh giá mức độ quan trọng của các đặc trưng hành vi người dùng (feature importance) và phân tích những yếu tố tác động mạnh nhất đến khả năng rời bỏ dịch vụ.

Thiết lập một quy trình chuẩn về tiền xử lý dữ liệu, phân tích tương quan, xây dựng mô hình, đánh giá và tối ưu mô hình, làm cơ sở cho các nghiên cứu tiếp theo về phân tích hành vi người dùng trong lĩnh vực thương mại điện tử hoặc dịch vụ số.

Về mặt xã hội:

Góp phần thúc đẩy khả năng cá nhân hóa dịch vụ số, tăng chất lượng trải nghiệm nghe nhạc trực tuyến cho người dùng.

Tạo tiền đề cho việc ứng dụng AI và Data Mining vào việc hiểu rõ nhu cầu, sở thích và hành vi người dùng, phù hợp với xu hướng chuyển đổi số trong lĩnh vực giải trí trực tuyến.

1.4. Đối tượng nghiên cứu.

Đối tượng nghiên cứu chính của đề tài bao gồm:

Dữ liệu hành vi người dùng trên Spotify, bao gồm các thông tin như số phút nghe mỗi ngày, tần suất bỏ qua bài hát (skip rate), số playlist, số lượt thích, quốc gia, loại tài khoản (free/premium) và các chỉ số tương tác khác.

Các mô hình học máy phục vụ dự đoán churn: Random Forest, Gradient Boosting, XGBoost, Logistic Regression và các kỹ thuật phân tích đặc trưng.

Thuật toán xử lý và phân tích dữ liệu: kỹ thuật tiền xử lý dữ liệu, xử lý dữ liệu mất cân bằng, chuẩn hóa và đánh giá tương quan nhằm xây dựng đặc trưng tối ưu cho mô hình dự đoán.

1.5. Phạm vi nghiên cứu

Đề tài được giới hạn trong các phạm vi sau:

Về nội dung: Nghiên cứu tập trung vào bài toán dự đoán *khả năng rời bỏ (churn)* của người dùng Spotify dựa trên dữ liệu hành vi, không bao gồm các yếu tố bên ngoài như

xu hướng thị trường, chiến dịch marketing hoặc tác động cạnh tranh giữa các nền tảng. Đề tài chỉ đánh giá churn như một bài toán phân loại nhị phân (rời bỏ / không rời bỏ).

Về kỹ thuật: Các mô hình sử dụng chủ yếu là những thuật toán học máy phổ biến, không mở rộng sang các mô hình deep learning phức tạp (như LSTM hoặc Transformer) do hạn chế về tài nguyên tính toán và tính khả thi của dữ liệu. Dữ liệu được xử lý và huấn luyện trong phạm vi một tập dữ liệu tĩnh (offline dataset), chưa triển khai mô hình ở môi trường sản xuất thực tế.

Về dữ liệu: Dữ liệu sử dụng là bộ dữ liệu hành vi người dùng Spotify được công bố công khai, có thể giới hạn về số lượng mẫu, số chiều dữ liệu và không đại diện hoàn toàn cho toàn bộ hệ thống Spotify thực tế. Dữ liệu không bao gồm thông tin định danh cá nhân (PII), không chứa dữ liệu âm thanh gốc hoặc nội dung bài hát.

1.6. Phương pháp nghiên cứu

Đề tài được thực hiện dựa trên sự kết hợp của các phương pháp sau:

Phương pháp nghiên cứu lý thuyết: Tổng hợp và phân tích các tài liệu học thuật về dự đoán churn, các thuật toán học máy và quy trình phân tích dữ liệu để xây dựng cơ sở lý thuyết vững chắc cho đề tài. Nghiên cứu các mô hình dự đoán churn trong các lĩnh vực như viễn thông, ngân hàng và thương mại điện tử để tham khảo phương pháp và cách tiếp cận.

Phương pháp thực nghiệm (Empirical Research): Tiến hành thu thập, làm sạch và tiền xử lý dữ liệu người dùng. Xây dựng mô hình dự đoán, huấn luyện và tối ưu mô hình bằng các chỉ số định lượng như Accuracy, Precision, Recall, F1-Score và ROC-AUC. Thực hiện phân tích lỗi (error analysis) và so sánh hiệu năng giữa các thuật toán.

Phương pháp đánh giá định lượng: Sử dụng tập kiểm tra tách riêng (test set) và kỹ thuật cross-validation nhằm đảm bảo khách quan và tăng độ tin cậy của kết quả. Phân tích mức độ quan trọng của từng đặc trưng và đánh giá khả năng diễn giải của mô hình (model interpretability).

Phương pháp phân tích hệ thống: Đánh giá toàn diện khả năng hoạt động của mô hình sau khi xây dựng, bao gồm tốc độ xử lý, mức độ ổn định và khả năng mở rộng trong các hệ thống thực tế. Đề xuất hướng cải tiến và ứng dụng mô hình vào hệ thống gợi ý hoặc chiến dịch marketing của các nền tảng tương tự Spotify.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1. Giới thiệu về Khai phá dữ liệu (Data Mining)

2.1.1. Khái niệm Data Mining

Khai phá dữ liệu (Data Mining) là quá trình phát hiện các mẫu, quy luật, xu hướng hoặc tri thức hữu ích từ các tập dữ liệu lớn. Đây là một nhánh quan trọng của lĩnh vực Khoa học dữ liệu (Data Science) và Hệ thống hỗ trợ ra quyết định, kết hợp nhiều kỹ thuật đến từ thống kê, học máy (Machine Learning), xử lý dữ liệu và trí tuệ nhân tạo.

Data Mining không chỉ đơn thuần là việc thu thập dữ liệu, mà là quá trình trích xuất thông tin có giá trị từ dữ liệu đã được lưu trữ. Thông qua các phương pháp như phân loại (classification), phân cụm (clustering), dự đoán (prediction), phát hiện bất thường (anomaly detection) hay phân tích kết hợp (association analysis), Data Mining giúp chúng ta hiểu sâu hơn về hành vi và quy luật ẩn trong dữ liệu.

Trong bối cảnh dữ liệu ngày càng trở nên phong phú và phức tạp, khai phá dữ liệu đóng vai trò nền tảng để xây dựng các mô hình dự đoán trong nhiều lĩnh vực khác nhau, đặc biệt là các ứng dụng trực tuyến lớn như Spotify, YouTube, Netflix, Facebook,…

2.1.2. Vai trò và ứng dụng trong thực tế

Data Mining có vai trò quan trọng trong việc chuyển đổi dữ liệu thô thành tri thức có thể hành động (actionable insights). Nhờ đó, doanh nghiệp và tổ chức có thể đưa ra các quyết định chính xác và tối ưu hơn. Một số vai trò nổi bật gồm:

Dự đoán và ra quyết định: Data Mining cho phép xây dựng các mô hình dự đoán xu hướng hoặc hành vi của người dùng, chẳng hạn như: dự đoán khả năng rời bỏ ứng dụng (user churn prediction), dự đoán nhu cầu sử dụng dịch vụ, dự đoán doanh thu, hành vi mua sắm,… Trong đề tài này, Data Mining được sử dụng để dự đoán khả năng rời bỏ Spotify của người dùng dựa trên các đặc trưng hành vi.

Phân tích hành vi người dùng: các nền tảng như Spotify hay Netflix sử dụng Data Mining để: hiểu sở thích của từng người dùng, tối ưu hệ thống gợi ý bài hát hoặc nội dung, nhận diện phân khúc người dùng và hành vi tương tác đặc trưng.

Phát hiện bất thường: dùng để phát hiện các hành vi bất thường hoặc rủi ro như: gian lận tấn công bảo mật, các mẫu sử dụng không bình thường.

Tối ưu hóa hoạt động doanh nghiệp: Data Mining hỗ trợ quản lý chiến dịch marketing, phân tích hiệu quả quảng cáo, tối ưu trải nghiệm người dùng (User Experience – UX).

Ứng dụng cụ thể trong ngành giải trí số (như Spotify): gợi ý bài hát cá nhân hóa dựa trên lịch sử nghe nhạc, dự đoán churn để giảm tỉ lệ người dùng rời bỏ, xác định nhóm người dùng tiềm năng đăng ký Premium, phân tích skip-rate, thời lượng nghe, khu vực địa lý… để cải thiện sản phẩm.

2.1.3. Quy trình khai phá dữ liệu

Thu thập dữ liệu

Tiền xử lý dữ liệu

Khai phá dữ liệu

Đánh giá và diễn giải kết quả

2.2. Phân loại thuộc tính dữ liệu (Features Classification)

2.2.1. Thuộc tính số (Numerical Features)

Thuộc tính số *(Numerical Features)*: Gồm các giá trị số như chiều cao, cân nặng, điểm số, v.v. Chúng có thể là biến liên tục hoặc rời rạc [1].

Được chia thành hai loại nhỏ:

Continuous (liên tục): ví dụ age, session_duration, songs_per_session, avg_time_per_song, total_listening_time,…

Discrete (rời rạc): ví dụ number_of_sessions, songs_listened, artists_followed, ads_clicked,…

Các phương pháp thống kê mô tả thường dùng: mean, median, std, min, max, quartile, skewness, kurtosis, histogram, boxplot, density plot.

2.2.2. Thuộc tính phân loại (Categorical Features)

Thuộc tính phân loại (*Categorical Features*): Gồm nhiều danh mục như giới tính, màu sắc, loại sản phẩm, v.v [1].

Được chia thành hai loại:

Nominal (danh nghĩa): không có thứ tự, ví dụ gender, region, favorite_genre, device_type, subscription_plan.

Ordinal (thứ tự): có thứ tự rõ ràng, ví dụ satisfaction_level (0–4), membership_duration_tier (new, regular, loyal).

Các phương pháp thống kê mô tả thường dùng: value_counts(), mode, tỷ lệ phần trăm, bar chart, count plot, pie chart.

2.3. Phân tích dữ liệu khám phá (Exploratory Data Analysis - EDA)

EDA (viết tắt của Exploratory Data Analyst) là một phương pháp khám phá dữ liệu, tìm ra các xu hướng, mẫu thử hoặc kiểm tra các giả định trong dữ liệu nhằm mục đích hiểu rõ về cấu trúc và tính chất của dữ liệu. Khi áp dụng các thuật toán học máy hoặc xây dựng các mô hình dự đoán, EDA góp phần quan trọng trong quá trình xử lý dữ liệu, giúp giải quyết các điều kiện ngoại lệ, giá trị thiếu và những vấn đề ảnh hưởng đến kết quả cuối cùng [2].

2.3.1. Mục tiêu của EDA

Một số mục đích của việc sử dụng EDA vào các dự án phân tích dữ liệu như [2]:

Tìm hiểu về cấu trúc dữ liệu: EDA là phương pháp giúp xác định cấu trúc dữ liệu bao gồm số lượng, kiểu dữ liệu, trường dữ liệu, sự liên kết giữa các trường dữ liệu,... Khi xác định được cấu trúc dữ liệu, các nhà phân tích dữ liệu có thể hiểu được mối quan hệ giữa các dữ liệu trong tệp.

Điều chỉnh và thay đổi: EDA giúp giải quyết các trường hợp thiếu giá trị, dữ liệu lỗi, các ngoại lệ trong dữ liệu. Điều này giúp các nhà phân tích dữ liệu điều chỉnh các phương án khắc phục kịp thời, tránh những ảnh hưởng nghiêm trọng đến dự án.

Xác định mối tương quan giữa các biến: Các biến đều chứa các giá trị riêng, EDA có khả năng phát hiện các liên hệ tiềm ẩn và sự ảnh hưởng giữa các biến với nhau, tạo sự liên kết giữa các thông tin dữ liệu nhằm xây dựng một quy trình phân tích tổng thể, rõ ràng.

Xây dựng cơ sở dữ liệu quan hệ: Các đối tượng dữ liệu quan trọng được phát triển mối quan hệ nhằm cấu trúc hóa dữ liệu theo sơ đồ, tiết kiệm thời gian xử lý những thông tin thừa, hạn chế sự sai sót của kết quả phân tích.

Chuẩn bị cho bước phân tích tiếp theo: Áp dụng EDA giúp loại bỏ các dữ liệu không cần thiết, dữ liệu thiếu giá trị và chuẩn hóa dữ liệu. Đây là yếu tố nền tảng để chuẩn bị cho các bước phân tích bằng thuật toán học máy.

2.3.2. Phân tích dữ liệu theo từng loại thuộc tính

Đối với thuộc tính số (Numerical Features): Phân tích phân bố (độ lệch, dạng chuông, đa đỉnh), kiểm tra outliers, mối quan hệ với biến mục tiêu Churn thông qua các chỉ số thống kê (mean, median theo nhóm Churn/No-Churn) và biểu đồ so sánh.

Đối với thuộc tính phân loại (Categorical Features): Phân tích tần suất và tỷ lệ của từng danh mục, kiểm tra mức độ ảnh hưởng của từng category đến tỷ lệ churn (churn rate theo nhóm), phát hiện các nhóm có nguy cơ rời bỏ cao.

Phân tích mối quan hệ giữa các biến: Ma trận tương quan (correlation matrix) đối với biến số, kiểm định Chi-square hoặc Cramér's V đối với biến phân loại, và các biểu đồ kết hợp (grouped boxplot, violin plot, stacked bar chart…).

2.3.3. Các công cụ và biểu đồ thường dùng trong EDA

Công cụ [3]:

Python là ngôn ngữ lập trình linh hoạt và được sử dụng rộng rãi trong khoa học dữ liệu nhờ hệ sinh thái thư viện phong phú. Trong EDA, hai thư viện quan trọng nhất là:

Pandas: Cung cấp các cấu trúc dữ liệu như DataFrame và Series, hỗ trợ hiệu quả cho việc nhập, làm sạch, xử lý và tóm tắt dữ liệu.

Matplotlib: Thư viện trực quan hóa mạnh mẽ, giúp tạo các biểu đồ như histogram, scatter plot, line chart,... để khám phá mối quan hệ và xu hướng trong dữ liệu.

Kết hợp Pandas và Matplotlib cho phép bạn thực hiện toàn bộ quá trình EDA từ thao tác dữ liệu đến trực quan hóa kết quả một cách linh hoạt và có thể tùy chỉnh cao.

Các loại biểu đồ:

EDA thường được thực hiện thông qua các kỹ thuật trực quan hóa và thống kê mô tả như [3]:

Biểu đồ phân phối (histogram)

Biểu đồ hộp (boxplot)

Ma trận tương quan (correlation matrix)

Biểu đồ phân tán (scatter plot)

Tóm tắt thống kê (mean, median, mode, std,...)

2.4. Kỹ thuật tiền xử lý và biến đổi dữ liệu (Data Wrangling)

2.4.1. Làm sạch dữ liệu (Data Cleaning)

Làm sạch dữ liệu là quy trình cần thiết để chuẩn bị dữ liệu thô cho ứng dụng máy học (ML) và nghiệp vụ thông minh (BI). Dữ liệu thô có thể chứa nhiều lỗi có khả năng gây ảnh hưởng đến độ chính xác của các mô hình ML, từ đó dẫn đến dự đoán không chính xác và tác động tiêu cực đến hoạt động kinh doanh [4].

Các bước làm sạch dữ liệu phổ biến bao gồm sửa chữa [4]:

Dữ liệu trùng lặp: Loại bỏ thông tin trùng lặp

Dữ liệu không liên quan: Xác định các trường quan trọng đối với phân tích và loại bỏ dữ liệu không liên quan khỏi phân tích

Dữ liệu ngoại lai: Dữ liệu ngoại lai có thể ảnh hưởng đáng kể đến hiệu suất của mô hình, vậy nên cần phải xác định các dữ liệu ngoại lai và tiến hành biện pháp thích hợp

Dữ liệu bị thiếu: Gắn cờ và loại bỏ hoặc thay thế dữ liệu bị thiếu

Lỗi cấu trúc: Sửa lỗi đánh máy và các điểm không nhất quán khác, đồng thời khiến dữ liệu tuân theo mẫu hoặc quy ước chung

2.4.2. Xử lý dữ liệu mất cân bằng (Handling Imbalanced Data)

Dữ liệu mất cân bằng (Imbalanced Data) xảy ra khi số lượng mẫu của một lớp (lớp đa số - majority class) vượt trội hơn rất nhiều so với lớp còn lại (lớp thiểu số - minority class). Điều này khiến mô hình có xu hướng dự đoán thiên về lớp đa số và bỏ qua lớp thiểu số [5].

Để giải quyết vấn đề này, ta sử dụng các kỹ thuật Resampling (Lấy mẫu lại):

2.4.2.1. Oversampling (Tăng mẫu)

Kỹ thuật này tập trung vào việc gia tăng số lượng mẫu của lớp thiểu số sao cho cân bằng với lớp đa số.

Random Oversampling: Sao chép ngẫu nhiên các mẫu hiện có của lớp thiểu số.

Ưu điểm: Đơn giản, không làm mất thông tin.

Nhược điểm: Dễ dẫn đến hiện tượng quá khớp (Overfitting) do mô hình học lại các mẫu giống hệt nhau nhiều lần.

SMOTE (Synthetic Minority Over-sampling Technique): Thay vì sao chép, SMOTE sinh ra các mẫu nhân tạo mới bằng cách nội suy tuyến tính giữa một điểm dữ liệu và các láng giềng gần nhất (K-Nearest Neighbors) của nó.

Ưu điểm: Giảm thiểu Overfitting, tạo ra sự đa dạng cho dữ liệu huấn luyện.

2.4.2.2. Undersampling (Giảm mẫu)

Kỹ thuật này tập trung vào việc giảm bớt số lượng mẫu của lớp đa số để cân bằng với lớp thiểu số.

Random Undersampling: Loại bỏ ngẫu nhiên các mẫu khỏi lớp đa số.

Ưu điểm: Giảm thời gian huấn luyện do tập dữ liệu nhỏ đi.

Nhược điểm: Có thể làm mất các thông tin quan trọng của lớp đa số, dẫn đến mô hình thiếu dữ liệu (Underfitting).

NearMiss: Một kỹ thuật chọn lọc các mẫu của lớp đa số dựa trên khoảng cách của chúng tới các mẫu của lớp thiểu số, giữ lại những mẫu khó phân loại nhất để mô hình học tốt hơn.

2.4.2.3. Kỹ thuật kết hợp (Hybrid Methods)

Kết hợp cả Oversampling và Undersampling để tận dụng ưu điểm của cả hai.

SMOTE + Tomek Links: Dùng SMOTE để sinh thêm mẫu cho lớp thiểu số, sau đó dùng Tomek Links để loại bỏ các cặp mẫu (một từ lớp đa số, một từ lớp thiểu số) nằm quá gần nhau (biên giới chồng lấn) để làm sạch biên phân lớp, giúp mô hình phân tách rõ ràng hơn.

2.4.3. Biến đổi và chuẩn hóa dữ liệu (Data Transformation & Scaling)

Các thuật toán học máy (đặc biệt là các thuật toán dựa trên khoảng cách như KNN, K-Means, SVM) rất nhạy cảm với sự chênh lệch về độ lớn của dữ liệu (ví dụ: cột Tuổi từ 0-100, cột Lương từ 1.000-100.000).

Standardization (Chuẩn hóa Z-score): Biến đổi dữ liệu sao cho có trung bình bằng 0 và độ lệch chuẩn bằng 1.

$$z = \frac{x - \mu}{\sigma}$$

Normalization (Min-Max Scaling): Co dãn dữ liệu về một khoảng cố định, thường là [0,1]

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

2.4.4. Mã hóa dữ liệu phân loại (Categorical Encoding)

Hầu hết các mô hình học máy chỉ làm việc với dữ liệu số. Do đó, các thuộc tính phân loại (dạng chữ) cần được chuyển đổi sang dạng số.

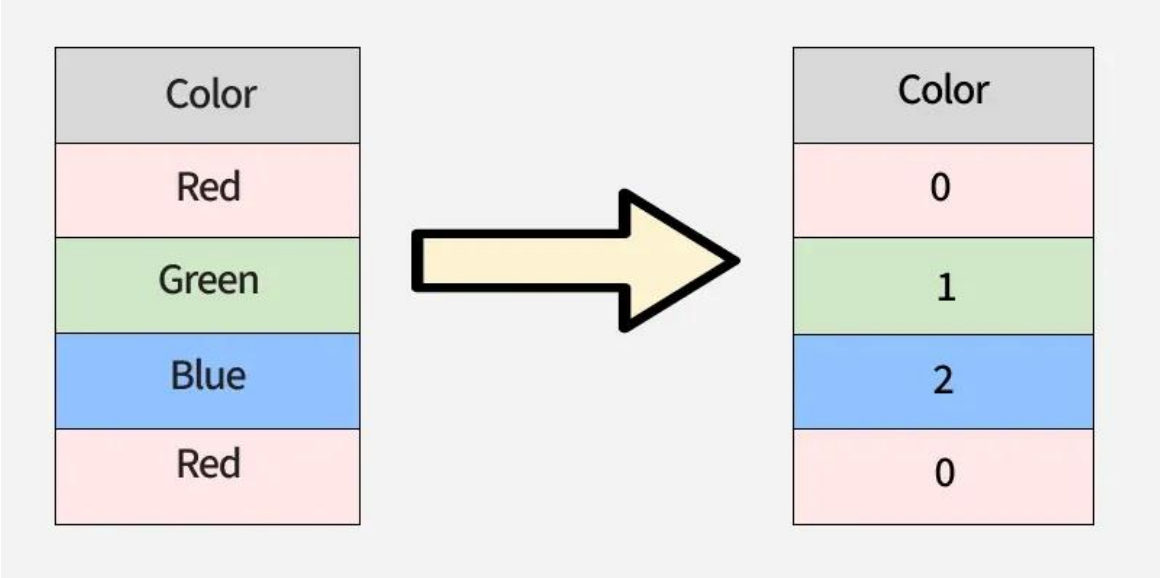
2.4.4.1. Label Encoding

Label Encoding gán cho mỗi danh mục (category) một số nguyên duy nhất. Phương pháp này đơn giản và tiết kiệm bộ nhớ, nhưng có thể vô tình tạo ra thứ tự giả giữa các danh mục dù thực tế chúng không có thứ tự [6].

Thường được sử dụng trong các mô hình dựa trên cây quyết định như Decision Trees, Random Forest, XGBoost.

Ưu điểm: Đơn giản, tiết kiệm bộ nhớ.

Nhược điểm: Tạo ra thứ tự ngầm (implicit order), dễ bị các mô hình không dựa trên cây (ví dụ: Logistic Regression, SVM, Neural Network) hiểu sai khi áp dụng cho dữ liệu danh nghĩa (nominal data).



Hình 1 Ví dụ Label Encoding

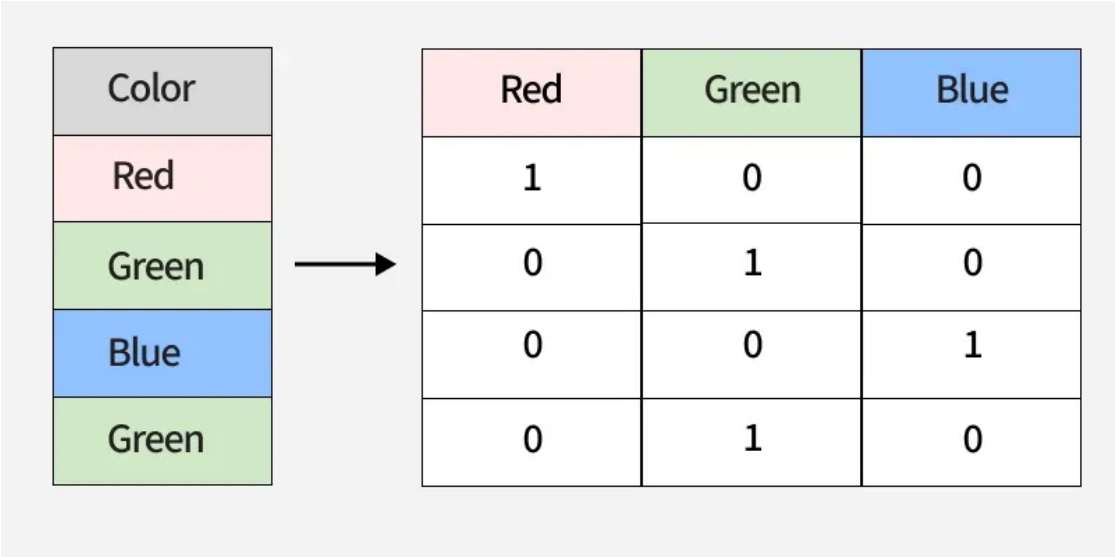
2.4.4.2. One-Hot Encoding

One-Hot Encoding chuyển mỗi danh mục thành một cột nhị phân (0/1), mỗi cột đại diện cho một danh mục riêng biệt. Phương pháp này tránh được việc tạo thứ tự giả nhưng có thể làm tăng mạnh số chiều dữ liệu nếu thuộc tính có quá nhiều giá trị duy nhất [6].

Thường được sử dụng trong các mô hình tuyến tính, Logistic Regression, SVM và mạng nơ-ron (Neural Networks).

Ưu điểm: Không giả định thứ tự giữa các danh mục; được hỗ trợ rộng rãi bởi hầu hết các thư viện.

Nhược điểm: Gây ra hiện tượng bùng nổ chiều (high dimensionality) và dữ liệu thưa (sparse data) khi thuộc tính có nhiều category (ví dụ: region, favorite_genre có hàng trăm giá trị khác nhau).

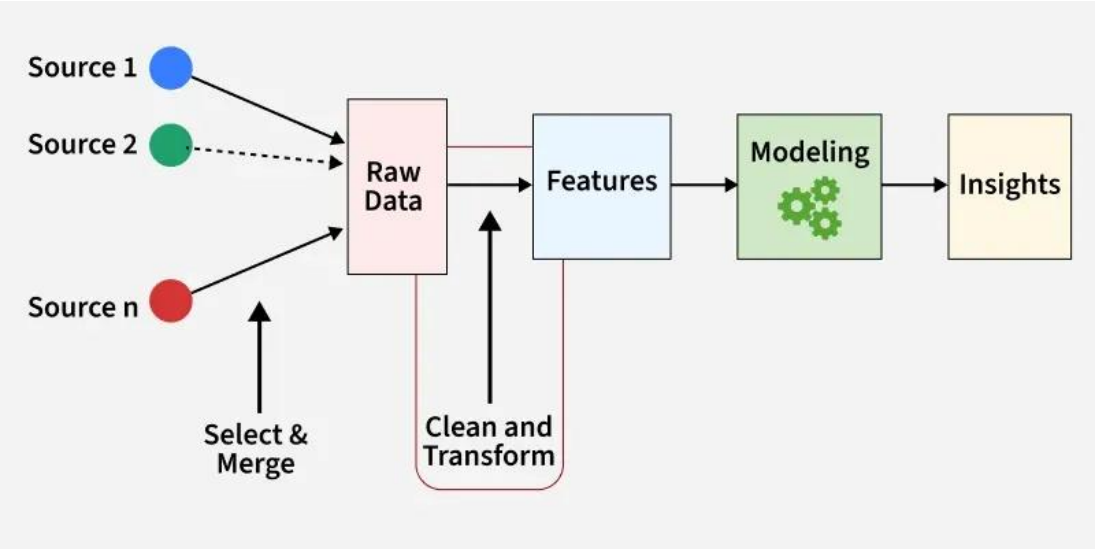


Hình 2 Ví dụ One-Hot Encoding

2.5. Kỹ thuật tạo đặc trưng (Feature Engineering)

2.5.1. Khái niệm và tầm quan trọng

Tạo đặc trưng (Feature Engineering) là quá trình chọn lọc, tạo mới hoặc biến đổi các đặc trưng (các biến đầu vào hoặc dữ liệu thô) nhằm giúp các mô hình học máy nhận diện mẫu hình một cách hiệu quả hơn. Quá trình này bao gồm việc chuyển đổi dữ liệu thô thành các đầu vào có ý nghĩa, từ đó nâng cao độ chính xác và hiệu suất tổng thể của mô hình [7].



Hình 3 Cấu trúc Feature Engineering

Tầm quan trọng của Feature Engineering: Tạo đặc trưng (Feature Engineering) có thể ảnh hưởng mạnh mẽ đến hiệu suất của mô hình. Bằng cách tinh chỉnh các đặc trưng, chúng ta có thể đạt được những lợi ích sau [7]:

Nâng cao độ chính xác (Improve accuracy): Việc lựa chọn và xây dựng các đặc trưng phù hợp giúp mô hình học được những mẫu hình thực sự quan trọng, từ đó đưa ra dự đoán chính xác hơn.

Giảm hiện tượng quá khớp (Reduce overfitting): Sử dụng ít đặc trưng hơn nhưng có giá trị thông tin cao giúp mô hình tránh việc ghi nhớ dữ liệu huấn luyện và khái quát hóa tốt hơn trên dữ liệu mới.

Tăng khả năng giải thích (Boost interpretability): Các đặc trưng được thiết kế hợp lý giúp dễ dàng hiểu được lý theo cách mô hình đưa ra quyết định (đặc biệt quan trọng khi phân tích hành vi churn của người dùng Spotify).

Tăng hiệu quả tính toán (Enhance efficiency): Tập trung vào những đặc trưng quan trọng sẽ rút ngắn thời gian huấn luyện và dự đoán của mô hình, tiết kiệm tài nguyên và thời gian xử lý.

Nhờ đó, Feature Engineering thường được xem là yếu tố quyết định sự thành công của một dự án học máy, đôi khi còn quan trọng hơn cả việc lựa chọn thuật toán.

2.5.2. Các kỹ thuật chính trong Feature Engineering

Dựa trên các phương pháp phổ biến, Feature Engineering bao gồm các kỹ thuật cốt lõi sau, có thể áp dụng trực tiếp vào dữ liệu hành vi để dự đoán churn [7]:

Tạo đặc trưng mới (Feature Creation): Sinh ra các đặc trưng mới từ kiến thức miền (domain knowledge), mẫu hình dữ liệu hoặc bằng cách kết hợp các đặc trưng hiện có.

Ví dụ: Tạo engagement_score = (songs_listened × 0.4) + (thumbs_up × 0.3) – (thumbs_down × 0.5) – (ads_clicked × 0.2), phản ánh mức độ gắn kết của người dùng.

Biến đổi đặc trưng (Feature Transformation): Điều chỉnh đặc trưng để phù hợp hơn với mô hình học, bao gồm:

Chuẩn hóa và scaling (ví dụ: Min-Max Scaling đưa về [0,1]; Standard Scaling đưa về trung bình 0, độ lệch chuẩn 1).

Mã hóa dữ liệu phân loại (ví dụ: One-Hot Encoding cho biến subscription_plan).

Biến đổi toán học (ví dụ: Log transformation cho biến lệch phải như total_listening_time).

Trích xuất đặc trưng (Feature Extraction): Giảm chiều dữ liệu và tăng độ chính xác bằng cách:

Giảm chiều (ví dụ: PCA cho các đặc trưng tương quan cao như thời lượng nghe và số bài hát).

Tổng hợp và kết hợp đặc trưng (ví dụ: Phân nhóm account_age thành các bin: "mới", "thường xuyên", "trung thành").

Chọn lọc đặc trưng (Feature Selection): Chọn tập con đặc trưng liên quan nhất bằng:

Phương pháp filter (ví dụ: phân tích tương quan với churn).

Phương pháp wrapper (dựa trên hiệu suất mô hình).

Phương pháp embedded (tích hợp trong huấn luyện, như feature importance từ XGBoost).

Chuẩn hóa đặc trưng (Feature Scaling): Đảm bảo các đặc trưng đóng góp bình đẳng, sử dụng Min-Max Scaling hoặc Standard Scaling cho các biến số như skip_rate và ads_clicked.

2.5.3. Các bước thực hiện Feature Engineering

Quá trình Feature Engineering thường theo các bước sau, phù hợp với quy trình KDD trong dự đoán churn [7]:

Làm sạch dữ liệu: Xác định và sửa lỗi, giá trị thiếu hoặc không nhất quán (ví dụ: xử lý missing values trong last_session_date).

Biến đổi dữ liệu: Scaling, normalization và encoding dữ liệu thô.

Trích xuất đặc trưng: Tạo đặc trưng mới từ dữ liệu hiện có (ví dụ: tính skip_rate = songs_skipped / songs_listened).

Chọn lọc đặc trưng: Sử dụng tương quan, mutual information hoặc stepwise regression để loại bỏ đặc trưng không liên quan.

Lặp lại và tinh chỉnh: Đánh giá dựa trên hiệu suất mô hình (ROC-AUC) và điều chỉnh liên tục.

2.6. Các thuật toán học máy ứng dụng trong dự đoán churn

2.6.1. Random Forest

Random Forest là một thuật toán học máy sử dụng rất nhiều cây quyết định (decision trees) để đưa ra dự đoán chính xác hơn. Mỗi cây chỉ xem xét một phần ngẫu nhiên của dữ liệu và các đặc trưng, sau đó kết quả cuối cùng được tổng hợp bằng cách bỏ phiếu đa số (đối với bài toán phân loại) hoặc lấy trung bình (đối với bài toán hồi quy). Đây là một kỹ thuật học tập tập thể (ensemble learning) điển hình, giúp tăng độ chính xác và giảm sai số [8].

Các đặc điểm nổi bật của Random Forest [8]:

Xử lý tốt dữ liệu thiếu (Handles Missing Data): Thuật toán vẫn hoạt động ổn định ngay cả khi có một số giá trị bị thiếu, không bắt buộc phải điền thủ công.

Cung cấp độ quan trọng của đặc trưng (Feature Importance): Cho biết đặc trưng nào (cột nào) có ảnh hưởng lớn nhất đến dự đoán, rất hữu ích để hiểu rõ hành vi người dùng và yếu tố dẫn đến churn.

Hiệu quả với dữ liệu lớn và phức tạp: Có thể xử lý tập dữ liệu có hàng chục nghìn mẫu và hàng trăm đặc trưng mà không bị chậm hoặc giảm độ chính xác.

Ứng dụng đa dạng: Dùng được cho cả bài toán phân loại (dự đoán nhãn: churn/không churn) và hồi quy (dự đoán giá trị số như thời lượng nghe nhạc, doanh thu…).

Ưu điểm của Random Forest [8]:

Cho kết quả dự đoán rất chính xác ngay cả trên tập dữ liệu lớn.

Xử lý tốt giá trị thiếu mà không làm giảm độ chính xác.

Không yêu cầu chuẩn hóa hoặc scaling dữ liệu (khác với Logistic Regression hay SVM).

Khi kết hợp nhiều cây quyết định, giảm đáng kể nguy cơ quá khớp (overfitting) so với một cây quyết định đơn lẻ.

Nhờ những ưu điểm trên, Random Forest thường được chọn làm mô hình baseline mạnh và là một trong những thuật toán hiệu quả nhất trong các dự án dự đoán churn Spotify.

2.6.2. Gradient Boosting

Gradient Boosting là một thuật toán thuộc nhóm Boosting, trong đó các mô hình mới được huấn luyện tuần tự nhằm giảm thiểu hàm mất mát (loss function) – ví dụ: mean squared error (hồi quy) hoặc log-loss/cross-entropy (phân loại) – của toàn bộ tập hợp mô hình trước đó, bằng cách sử dụng tối ưu gradient descent [9].

Nguyên lý hoạt động:

Ở mỗi vòng lặp:

Tính gradient (đạo hàm riêng) của hàm mất mát theo dự đoán hiện tại của ensemble.

Huấn luyện một mô hình yếu (weak learner – thường là cây quyết định nông) để dự đoán chính xác giá trị gradient này (hay còn gọi là residual errors).

Cộng dự đoán của mô hình yếu mới vào kết quả tổng (với hệ số learning_rate để kiểm soát tốc độ học).

Lặp lại cho đến khi đạt số vòng lặp quy định hoặc hàm mất mát không cải thiện đáng kể trên tập validation (early stopping).

Đặc điểm chính:

Các mô hình được xây dựng tuần tự, mỗi mô hình tập trung sửa lỗi của tất cả các mô hình trước → cực kỳ mạnh trên dữ liệu bảng (tabular data).

Rất nhạy cảm với tham số: learning_rate nhỏ + số cây lớn thường cho kết quả tốt nhất.

Dễ bị overfitting nếu không điều chỉnh tốt max_depth, subsample, min_child_weight…

2.6.3. XGBoost

XGBoost (viết tắt của eXtreme Gradient Boosting) là một thuật toán học máy tiên tiến được thiết kế tối ưu về hiệu năng, tốc độ và độ chính xác, hiện là một trong những thuật toán mạnh nhất trên dữ liệu dạng bảng (tabular data) và liên tục thống trị các cuộc thi Kaggle cũng như các dự án thực tế về dự đoán churn, gian lận, xếp hạng… [10]

Bản chất: XGBoost là phiên bản cải tiến cực mạnh của Gradient Boosting. Nó vẫn sử dụng cơ chế boosting (xây dựng tuần tự các cây quyết định yếu để sửa lỗi của nhau) nhưng bổ sung rất nhiều tối ưu hóa hệ thống và thuật toán.

Các cải tiến nổi bật so với Gradient Boosting truyền thống [10]:

Thêm regularization L1 & L2 → giảm overfitting mạnh mẽ.

Tự động xử lý giá trị thiếu (missing values) một cách thông minh.

Thuật toán chia nút nhanh hơn nhờ weighted quantile sketch và sparsity-aware split finding.

Hỗ trợ tính toán song song (parallel processing) và GPU acceleration → huấn luyện nhanh hơn hàng chục lần.

Tích hợp sẵn tham số scale_pos_weight để xử lý trực tiếp dữ liệu mất cân bằng (rất phù hợp với bài toán churn).

Hỗ trợ early stopping và đánh giá trên tập validation trong quá trình huấn luyện.

Ưu điểm của XGBoost [10]:

Hiệu suất cực cao, thường đạt ROC-AUC 0.92–0.96 trong dự án Spotify churn.

Có thể mở rộng tốt với dữ liệu hàng triệu bản ghi.

Cung cấp feature importance (gain, weight, cover) → dễ phân tích yếu tố ảnh hưởng đến hành vi rời bỏ.

Linh hoạt, tùy chỉnh được rất nhiều tham số (learning_rate, max_depth, subsample, colsample_bytree, gamma…).

Được hỗ trợ rộng rãi trên Python, R, Julia và tích hợp sẵn trong các framework lớn.

Nhược điểm cần lưu ý [10]:

Tốn tài nguyên tính toán (đặc biệt khi dùng GPU hoặc dữ liệu rất lớn).

Nhạy cảm với nhiễu và outliers → cần tiền xử lý và feature engineering cẩn thận.

Dễ bị overfitting nếu không điều chỉnh tham số hợp lý (đặc biệt trên tập dữ liệu nhỏ).

Khó giải thích hơn so với Logistic Regression hoặc một cây quyết định đơn lẻ.

2.6.4. So sánh ưu nhược điểm các thuật toán

Tiêu chí	Random Forest	Gradient Boosting (GBM)	XGBoost
Loại ensemble	Bagging (song song)	Boosting (tuần tự)	Boosting (tuần tự + tối ưu cực mạnh)
Hiệu suất dự đoán (ROC-AUC)	0.88 – 0.93	0.90 – 0.94	0.92 – 0.96+ (cao nhất)
Tốc độ huấn luyện	Nhanh – trung bình	Chậm	Nhanh nhất (nhờ parallelism + GPU)
Khả năng xử lý dữ liệu lớn	Tốt	Trung bình	Xuất sắc (hàng chục triệu mẫu)
Xử lý mất cân bằng dữ liệu	Tốt (class_weight hoặc BalancedRandomForest)	Tốt (sampling trong mỗi iteration)	Tốt nhất (scale_pos_weight tích hợp)
Xử lý missing values	Tốt (tự động)	Cần điền trước	Tốt nhất (tự học hướng đi tối ưu)
Khả năng chống overfitting	Rất tốt (bagging + random feature)	Dễ overfitting nếu không tuning	Tốt nhất nhờ regularization L1/L2
Yêu cầu scaling dữ liệu	Không cần	Không cần	Không cần
Feature importance	Rõ ràng, dễ hiểu	Có	Chi tiết nhất (gain, weight, cover)
Khả năng giải thích mô hình	Trung bình – tốt	Trung bình	Trung bình (có thể dùng SHAP để giải thích)

Độ khó tuning tham số	Dễ (ít tham số quan trọng)	Khó	Khó hơn nhưng có early stopping hỗ trợ
Thư viện phổ biến	scikit-learn	scikit-learn, LightGBM, CatBoost	xgboost, LightGBM, CatBoost
Phù hợp làm baseline	Rất phù hợp	Không khuyến khích	Không cần làm baseline
Phù hợp làm mô hình cuối	Tốt	Tốt	Tốt nhất trong hầu hết các dự án churn

Bảng 1 So sánh ưu nhược điểm các thuật toán

2.7. Phương pháp chia tập dữ liệu và đánh giá mô hình

2.7.1. Chia tập huấn luyện và kiểm tra (Train/Test Split)

Train test split là một chiến lược phổ biến để phân vùng một tập dữ liệu thành hai nhóm: một tập huấn và một bộ kiểm tra. Bộ huấn luyện được sử dụng để xây dựng mô hình, trong khi bộ kiểm tra được sử dụng để đánh giá độ chính xác của mô hình. Kỹ thuật này được sử dụng rộng rãi trong các ứng dụng học máy và khai thác dữ liệu [11].

Dữ liệu được chia thành tập huấn luyện (train) và tập kiểm tra (test) theo tỷ lệ phổ biến 80/20 hoặc 75/25.

Sử dụng tham số stratify=y để đảm bảo tỷ lệ Churn/No-Churn trong cả hai tập giống hệt nhau → rất quan trọng khi dữ liệu mất cân bằng.

Tập test được giữ nguyên, chỉ dùng một lần cuối cùng để đánh giá hiệu suất thực tế của mô hình.

2.7.2. Kiểm định chéo (Cross-Validation)

Kiểm định chéo là một phương pháp chia một tập dữ liệu thành hai phần: một tập huấn luyện và một tập hợp xác nhận. Tập huấn được sử dụng để xây dựng mô hình, trong khi tập hợp xác thực được sử dụng để đánh giá độ chính xác của mô hình. Điều quan trọng cần lưu ý là kích thước của tập huấn và bộ xác nhận có thể khác nhau tùy

thuộc vào kích thước của tập dữ liệu. Ưu điểm của việc sử dụng xác nhận chéo là nó cho phép đánh giá chính xác hơn về hiệu suất của mô hình [11].

2.7.3. Đánh giá mô hình trên dữ liệu mất cân bằng

2.7.3.1. Accuracy, Precision, Recall, F1-Score

Các chỉ số đánh giá [12]:

Accuracy (Độ chính xác tổng thể)

Là chỉ số cơ bản để đánh giá hiệu suất của mô hình phân loại. Nó cho biết tỷ lệ dự đoán đúng của mô hình trong tổng số tất cả các dự đoán.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

Precision (Độ chuẩn xác)

Đo lường trong số các dự đoán dương tính (positive) mà mô hình đưa ra, có bao nhiêu trường hợp thực sự là dương tính. Chỉ số này đặc biệt hữu ích khi chi phí của dương tính giả (false positive) rất cao, ví dụ như trong chẩn đoán y khoa khi dự đoán một người mắc bệnh trong khi thực tế không mắc có thể gây hậu quả nghiêm trọng.

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

Recall (Độ bao phủ)

Đo lường trong số các trường hợp thực sự dương tính, mô hình đã nhận diện đúng được bao nhiêu. Chỉ số này quan trọng khi việc bỏ sót một trường hợp dương tính (false negative) gây thiệt hại lớn hơn so với dương tính giả.

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

F1-Score

Là trung bình điều hòa của Precision và Recall. Chỉ số này hữu ích khi cần một sự cân bằng giữa Precision và Recall vì nó kết hợp cả hai thành một con số duy nhất. F1-Score càng cao chứng tỏ mô hình hoạt động tốt trên cả hai chỉ số. Giá trị của F1-Score nằm trong khoảng [0, 1].

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

2.7.3.2. Confusion Matrix

Ma trận nhầm lẫn là một bảng đơn giản dùng để đánh giá hiệu suất của mô hình phân loại. Nó so sánh kết quả dự đoán của mô hình với giá trị thực tế và chỉ rõ mô hình đúng/sai ở đâu. Nhờ đó, ta dễ dàng nhận biết loại lỗi mô hình đang mắc phải để cải thiện [13].

Bảng gồm 4 ô cơ bản [13]:

True Positive (TP – Dương tính đúng): Mô hình dự đoán đúng là "sẽ churn" và người dùng thực sự đã churn.

True Negative (TN – Âm tính đúng): Mô hình dự đoán đúng là "không churn" và người dùng thực sự vẫn ở lại.

False Positive (FP – Dương tính giả) – Lỗi loại I: Mô hình dự đoán "sẽ churn" nhưng thực tế người dùng không rời bỏ → gửi ưu đãi giữ chân nhầm, lãng phí chi phí.

False Negative (FN – Âm tính giả) – Lỗi loại II: Mô hình dự đoán "không churn" nhưng thực tế người dùng đã rời bỏ → bỏ lỡ cơ hội giữ chân, mất khách hàng.

	Predicted Positive	Predicted Negative
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

Hình 4 Confusion Matrix

2.7.3.3. ROC-AUC và Precision-Recall Curve

ROC-AUC (Receiver Operating Characteristic – Area Under Curve)

Đường cong ROC-AUC là công cụ đánh giá quan trọng nhất cho các mô hình phân loại nhị phân, đặc biệt trong bài toán churn có dữ liệu mất cân bằng [14].

Đường cong ROC thể hiện mối quan hệ giữa:

True Positive Rate (TPR) = Recall = Sensitivity: Tỷ lệ người thực sự churn được mô hình phát hiện đúng.

$$TPR = \frac{TP}{TP + FN}$$

False Positive Rate (FPR) = 1 – Specificity: Tỷ lệ người không churn nhưng bị dự đoán nhầm là sẽ churn.

$$FPR = \frac{FP}{FP + TN}$$

AUC (Area Under the ROC Curve): Diện tích dưới đường cong ROC, giá trị nằm trong khoảng [0, 1].

AUC gần 1 (ví dụ: 0.90–1.00): Mô hình phân biệt rất tốt giữa hai lớp (churn và không churn). Khả năng xếp hạng người dùng theo mức độ nguy cơ rời bỏ gần như hoàn hảo.

AUC gần 0: Mô hình dự đoán ngược hoàn toàn (luôn nhầm lẫn churn thành không churn và ngược lại). Hiếm gặp trong thực tế.

AUC khoảng 0.5: Mô hình không học được bất kỳ mẫu hình nào có ý nghĩa → dự đoán tương đương với việc tung đồng xu (random guessing), hoàn toàn vô dụng.

Precision-Recall Curve & PR-AUC

Đặc biệt hữu ích khi lớp thiểu số (churn) rất ít (< 20%).

Vẽ mối quan hệ giữa Precision và Recall ở các ngưỡng khác nhau.

CHƯƠNG 3. PHÂN TÍCH VÀ CHUẨN BỊ DỮ LIỆU

3.1. Lý do chọn dataset

Nhóm lựa chọn dataset Spotify Analysis Dataset 2025 với mục đích giải quyết bài toán Xây dựng mô hình học máy để dự đoán khả năng rời bỏ của người dùng, phân tích các mô hình tương tác và tìm ra các nguyên nhân dẫn đến sự rời bỏ của người dùng ứng dụng Spotify.

3.2.Tổng quan dataset

Giới thiệu nguồn dataset

Dataset được cung cấp bởi người dùng Nabih Zahid trên nền tảng Kaggle – một trong những cộng đồng khoa học dữ liệu lớn nhất thế giới. Dataset có tiêu đề "Spotify Dataset for Churn Analysis" và được công khai miễn phí để tải xuống cũng như sử dụng cho mục đích học tập, nghiên cứu và phân tích mà không yêu cầu xin phép riêng.

Link tải dataset

Dataset được tải từ địa chỉ: Spotify Analysis Dataset 2025

Các bước tải dataset từ Kaggle

Để tải dataset, thực hiện theo các bước sau:

Truy cập vào đường dẫn https://www.kaggle.com/datasets/nabihazahid/spotify-dataset-for-churn-analysis

Nếu chưa có tài khoản Kaggle, đăng ký tài khoản miễn phí hoặc đăng nhập bằng tài khoản Google/GitHub hiện có.

Sau khi vào trang dataset, kéo xuống phần "Data" hoặc nhìn bên phải màn hình sẽ thấy nút Download (hoặc Download All nếu có nhiều file).

Nhấn nút Download → file sẽ tự động được tải về dưới dạng file .csv

3.3. Mô tả chi tiết dữ liệu

3.3.1. Giới thiệu tập dữ liệu

Tập dữ liệu gồm 8.000 bản ghi đại diện cho người dùng của một nền tảng nghe nhạc trực tuyến.s

Mỗi bản ghi chứa thông tin nhân khẩu học, hành vi nghe nhạc, loại thiết bị sử dụng và trạng thái rời bỏ dịch vụ (*churn*).

Dữ liệu bao gồm 12 biến, trong đó có cả biến định tính và định lượng.

3.3.2. Phân loại dữ liệu

Dựa trên kiểu dữ liệu và bản chất thông tin, dữ liệu được chia thành hai nhóm:

(1) Dữ liệu định lượng (Numerical)

Là các cột chứa giá trị số và có thể tính toán được:

Biến	Kiểu	Ý nghĩa
age	int	Tuổi người dùng
listening_time	int	Tổng thời gian nghe nhạc mỗi ngày (phút)
songs_played_per_day	int	Số bài hát được phát mỗi ngày
skip_rate	float	Tỉ lệ bỏ qua bài hát
ads_listened_per_week	int	Số quảng cáo nghe trong tuần
offline_listening	int (0/1)	Có sử dụng nghe offline
user_id	int	Mã định danh người dùng (ID)
is_churned	int (0/1)	Trạng thái rời bỏ dịch vụ

Bảng 2 Dữ liệu định lượng

(2) Dữ liệu định tính (Categorical)

Là các cột dạng văn bản hoặc phân loại:

Biến	Kiểu	Giá trị
gender	object	Male, Female, Other
country	object	8 quốc gia (AU, US, DE, IN, PK, FR, UK, CA)
subscription_type	object	Premium, Free, Student, Family
device_type	object	Desktop, Web, Mobile

Bảng 3 Dữ liệu định tính

3.4. Thống kê mô tả

3.4.1. Dữ liệu định lượng

Tuổi người dùng (age)

Min: 16, Max: 59

Mean: 37.66, Median: 38

Thời gian nghe nhạc (listening_time)

Min: 10, Max: 299 phút

Mean: 154 phút

→ Đa số người dùng nghe khoảng 2–3 giờ/ngày.

Số bài hát/ngày (songs_played_per_day)

Min: 1, Max: 99

Mean: 50 bài/ngày

→ Mức độ nghe khá cao.

Tỉ lệ bỏ qua bài hát (skip_rate)

Min: 0.00, Max: 0.60

Mean: 0.30

→ Trung bình người dùng bỏ qua 30% bài hát.

Quảng cáo/ngày (ads_listened_per_week)

Min: 0, Max: 49

Median: 0, Mean: 6.9

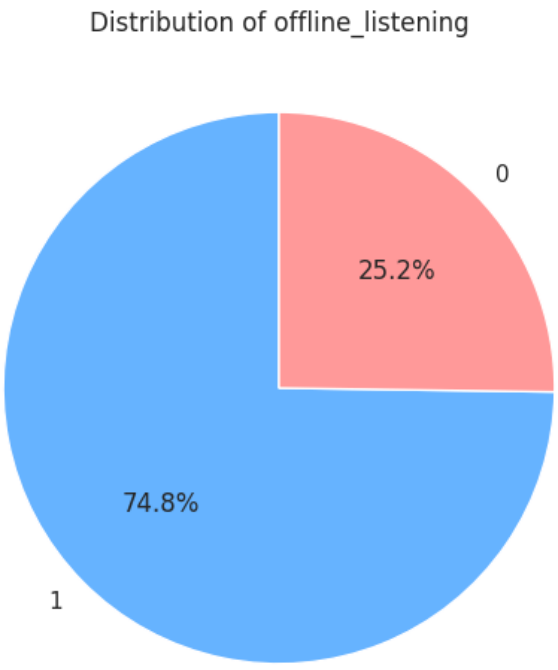
→ Nhiều người không nghe quảng cáo → dùng Premium/Student.

Nghe offline (offline_listening)

1: 74.78%

0: 25.22%

→ Đa số người dùng có sử dụng chế độ offline.



Hình 5 Biểu đồ phần trăm người nghe offline

3.4.2. Dữ liệu định tính

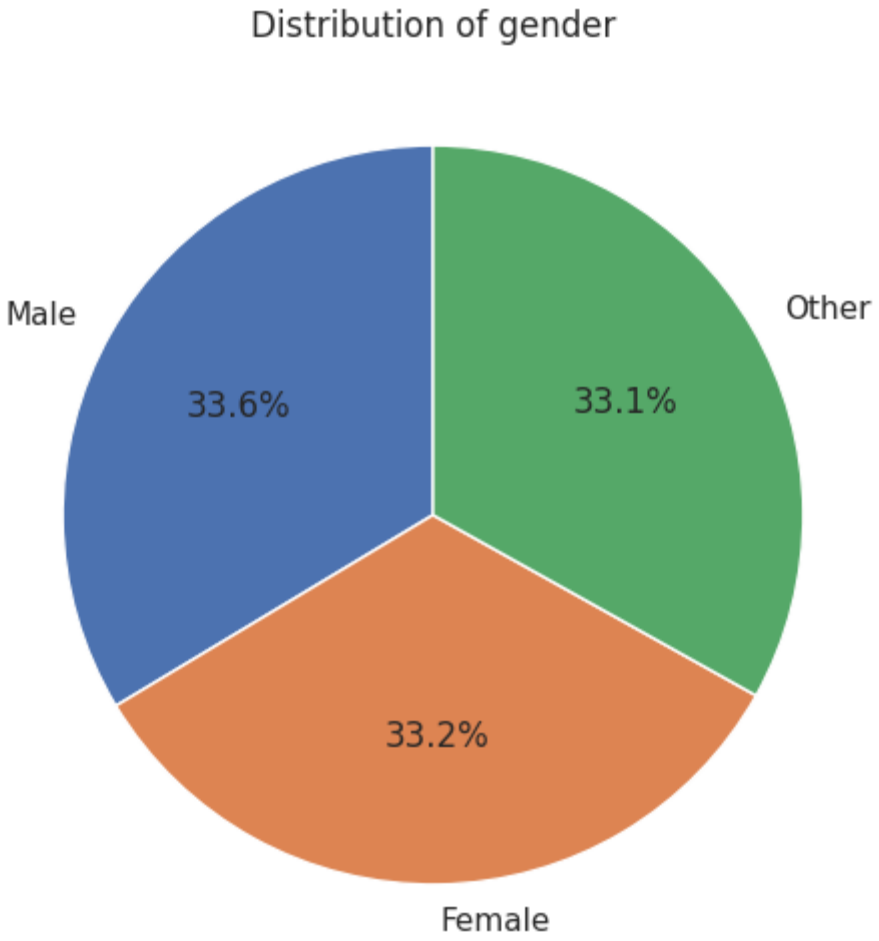
Giới tính (gender)

Male: 33.63%

Female: 33.24%

Other: 33.12%

→ Phân bố giới tính rất cân bằng.

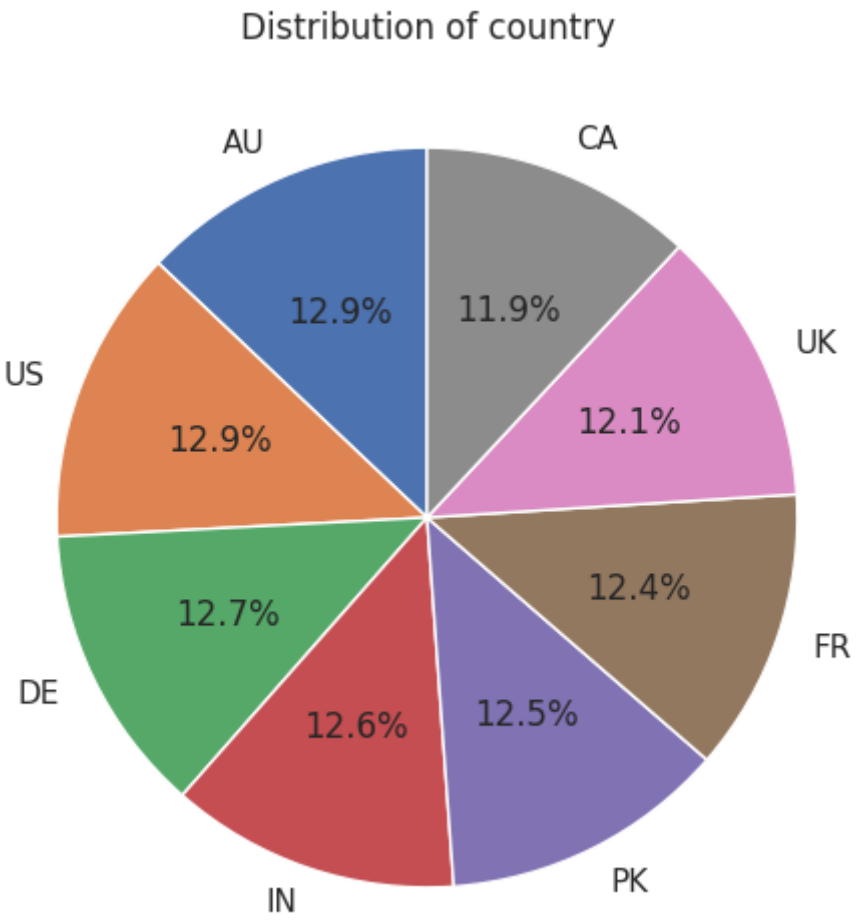


Hình 6 Biểu đồ phân bổ giới tính

Quốc gia (country)

Tỉ lệ mỗi quốc gia dao động từ 11.9% đến 12.9%, không có nhóm nào áp đảo.

→ Dữ liệu quốc gia phân bố khá đồng đều.



Hình 7 Biểu đồ phân bổ quốc gia

Loại gói đăng ký (subscription_type)

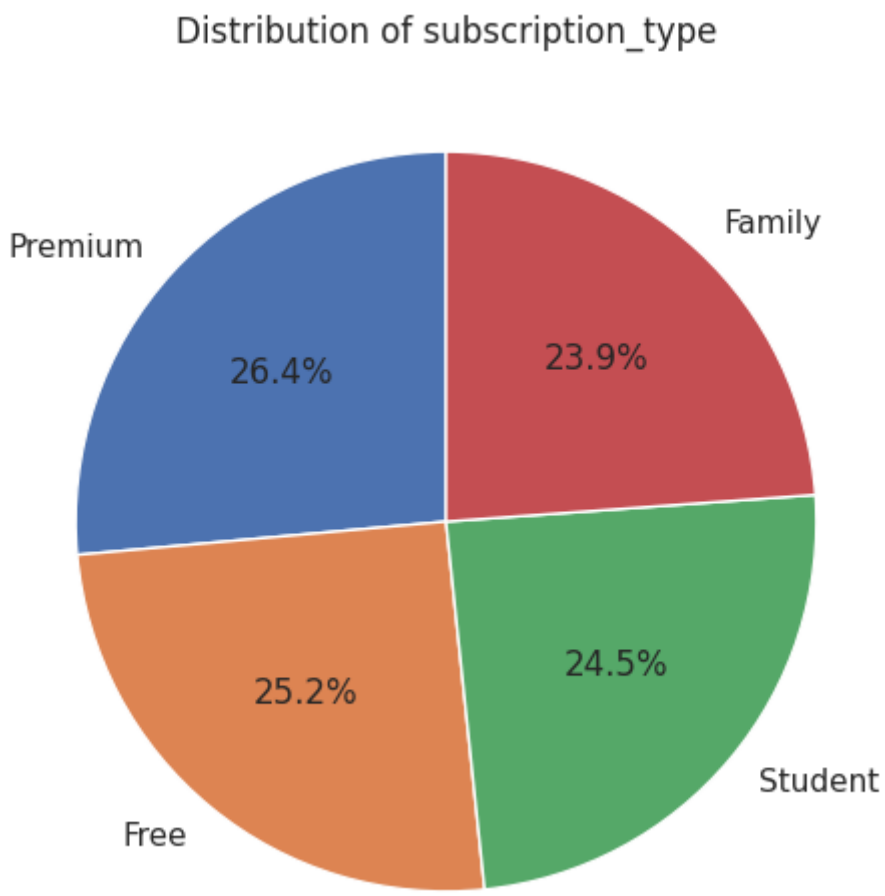
Premium: 26.44%

Free: 25.22%

Student: 24.49%

Family: 23.85%

→ Bốn loại gói đăng ký có tỉ lệ gần tương đương.



Hình 8 Biểu đồ phân bổ gói đăng ký

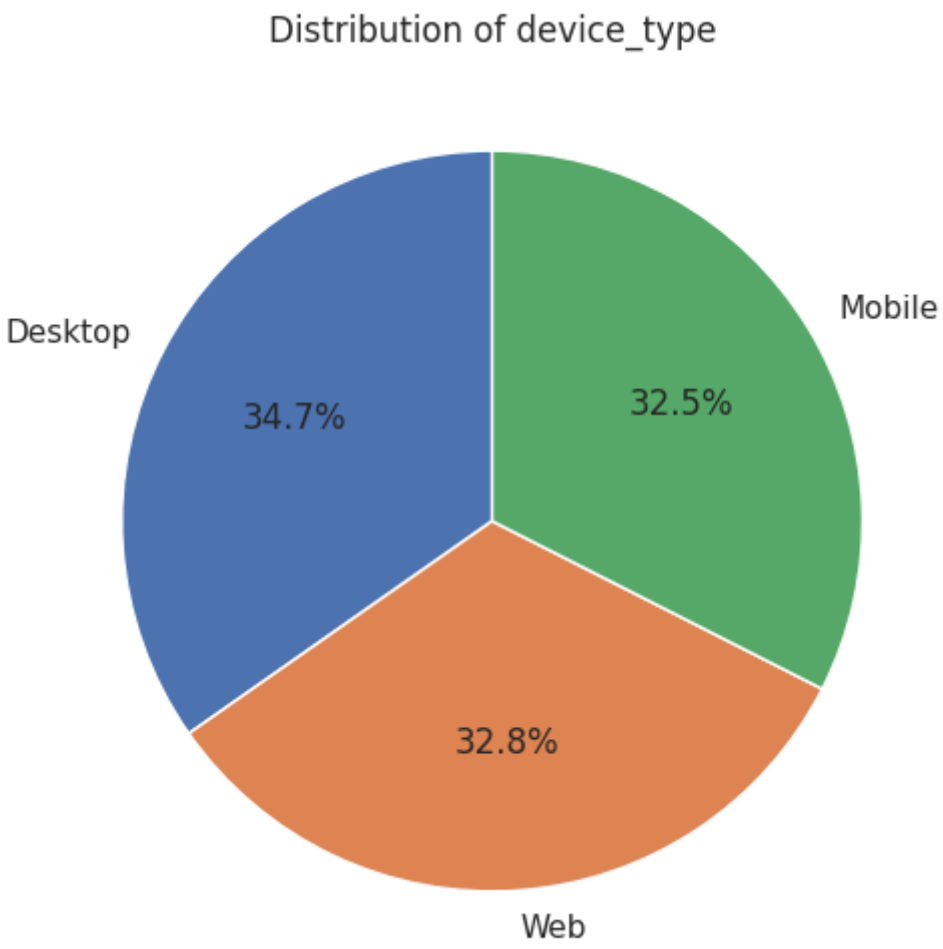
Thiết bị sử dụng (device_type)

Desktop: 34.72%

Web: 32.79%

Mobile: 32.49%

→ Phân bố thiết bị sử dụng cân đối giữa ba loại.



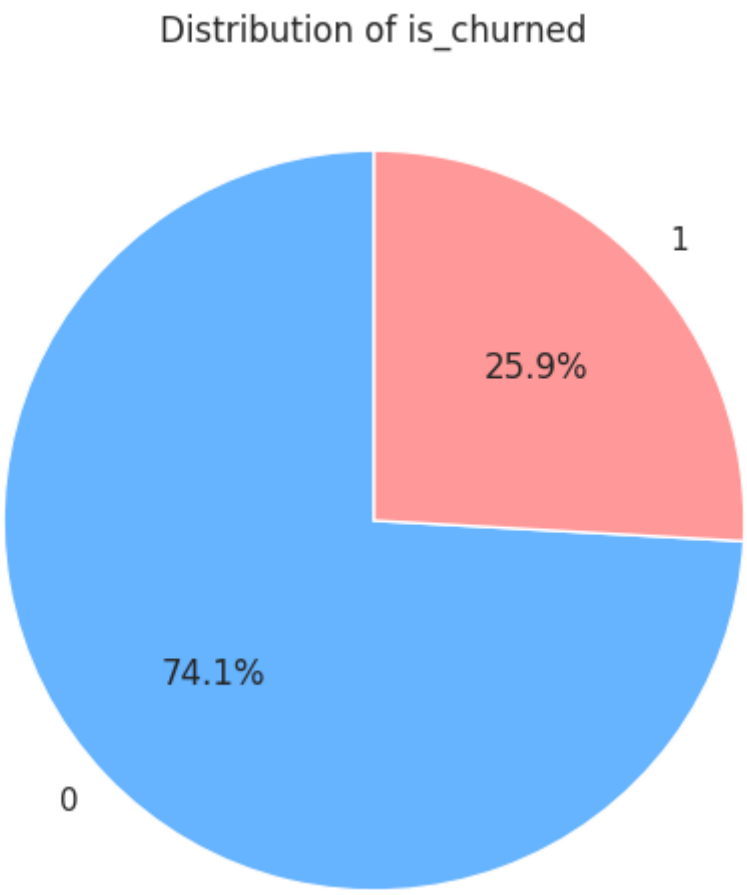
Hình 9 Biểu đồ phân bổ thiết bị

3.4.3. Biến mục tiêu (Target Feature): is_churned

0 (không rời bỏ): 74.11%

1 (rời bỏ): 25.89%

→ Dữ liệu có mất cân bằng nhẹ (imbalanced), nhưng vẫn trong giới hạn chấp nhận được khi huấn luyện mô hình.



Hình 10 Biểu đồ phân bổ biến mục tiêu

3.5. Nhận xét chung về dữ liệu

Không có giá trị thiếu (0 null).

Các biến đều có phân bố hợp lý, không có giá trị cực đoan bất thường.

Dữ liệu khá cân bằng giữa các nhóm phân loại như giới tính, quốc gia, thiết bị.

Một số cột dạng số (offline_listening, is_churned) thực chất là phân loại nhị phân → cần chuyển sang categorical.

Biến user_id chỉ mang tính định danh và không được dùng trong mô hình dự đoán churn.

3.6. Tiền xử lý dữ liệu

3.6.1. Làm sạch dữ liệu

- Loại bỏ cột `user_id` (không mang thông tin dự đoán) và đặt `user_id` làm index của DataFrame.

```
# set index cho dataset bằng thuộc tính user_id
df.set_index(df.user_id, inplace=True)
df.drop('user_id', inplace=True, axis=1) # drop cột user_id
df.head()
```

- Kiểm tra giá trị thiếu

```
def display_missing(df, feature_cols):
  n_rows = df.shape[0]
  for col in feature_cols:
    missing_count = df[col].isnull().sum()
    if(missing_count > 0):
      print(f"Col {col} has {missing_count*100/n_rows:.2f}% missing values")

display_missing(df, feature_cols)
```

- Không phát hiện giá trị ngoại lai nghiêm trọng cần loại bỏ.

3.6.2. Mã hoá (Label Encoding / One-hot)

Sử dụng One-Hot Encoding cho các cột categorical:

```
# Thay đổi kiểu dữ liệu của target feature thành categorical, nhận 2 giá trị {0,1}
df['is_churned'] = df['is_churned'].astype('category')

# for loop chuyển đổi kiểu dữ liệu
features = ['gender', 'country', 'subscription_type', 'device_type', 'offline_listening']
for feature in features:
    df[feature] = df[feature].astype('category')
```

- Sau mã hoá: 36 cột (tăng 24 cột dummy).

```
num_features = ['age', 'listening_time', 'songs_played_per_day', 'skip_rate', 'ads_listened_per_week']
cat_features = ['gender', 'country', 'ads_listened_per_week_fequences', 'subscription_type', 'device_type', 'offline_listening']

dataset_dummy = pd.get_dummies(df, columns=cat_features, drop_first=True)
dataset_dummy.head()
```

user_id	age	listening_time	songs_played_per_day	skip_rate	ads_listened_per_week	is_churned	gender_Male	gender_Other	country_CA	country_DE	...	country_US	ads_listened_per_week_fequences_Sometimes	ads_listene
1	54	26	23	0.20	31	1	False	False	True	False	...	False	False	
2	33	141	62	0.34	0	0	False	True	False	True	...	False	False	
3	38	199	38	0.04	0	1	True	False	False	False	...	False	False	
4	22	36	2	0.31	0	0	False	False	True	False	...	False	False	
5	29	250	57	0.36	0	1	False	True	False	False	...	True	False	

5 rows × 24 columns

3.6.3. Chuẩn hóa

- Các biến số liên tục (`age`, `listening_time`, `songs_played_per_day`, `skip_rate`, `ads_listened_per_week`) được chuẩn hóa bằng MinMaxScaler để đưa về phân phối chuẩn (mean=0, std=1).

```
scaler = MinMaxScaler()
scaler.fit(X_train[num_features])
X_train[num_features] = scaler.transform(X_train[num_features])
X_test[num_features] = scaler.transform(X_test[num_features])
```

```
pd.DataFrame(X_train).head()
```

user_id	age	listening_time	songs_played_per_day	skip_rate	gender_Male	gender_Other	country_CA	country_DE	country_FR	country_IN	...	country_UK	country_US	ads_listened_per_week_fequences_Sometimes	ads_l
4662	0.744186	0.442907	0.377551	0.983333	False	True	False	False	False	False	...	False	True	False	
5196	0.651163	0.235294	0.357143	0.266667	True	False	False	False	False	False	...	False	True	False	
7124	0.302326	0.276817	0.755102	0.866667	True	False	False	False	False	False	...	False	False	True	
3765	0.930233	0.363322	0.408163	0.333333	True	False	False	False	False	False	...	False	False	False	
6825	0.372093	0.328720	0.163265	0.283333	False	True	False	False	False	False	...	True	False	False	

5 rows × 21 columns

3.6.4. Xử lý mất cân bằng

- Tỷ lệ churn: khoảng 26% churn (1), 74% không churn (0) → mất cân bằng nhẹ.

- Đã thử các phương pháp: SMOTE, class_weight='balanced' trong mô hình.

- Kết quả tốt nhất đạt được khi dùng class_weight='balanced' trong Random Forest, Gradient Boosting, Logistic Regression, XGBoost

```
from imblearn.over_sampling import BorderlineSMOTE, SMOTE

# Over Sampling
X_borderLineSMOTE_res, y_borderLineSMOTE_res = BorderlineSMOTE(random_state=42, sampling_strategy='minority').fit_resample(X_train, y_train)
```

3.7. Chia dữ liệu thành tập Train/Test

```
X = dataset_dummy.drop('is_churned', axis=1)
y = dataset_dummy['is_churned']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

print("Features (X):", X.shape)
print("Target (y):", y.shape)
```

```
Features (X): (8000, 21)
Target (y): (8000,)
```

- Train: 8.000 mẫu

- Test: 2.000 mẫu (giữ nguyên tỷ lệ churn)

3.8. Mô tả bộ dữ liệu sau tiền xử lý

- Số lượng đặc trưng: 35 (sau one-hot và drop user_id)

- Không còn giá trị thiếu

- Tất cả biến số đã được chuẩn hóa

- Biến mục tiêu nhị phân (0/1)

3.9. Công cụ được sử dụng

- Python 3.11+

- Thư viện: pandas, numpy, matplotlib, seaborn, scikit-learn, xgboost, imbalanced-learn

- Môi trường: Jupyter Notebook / Google Colab (GPU T4)

CHƯƠNG 4. THUẬT TOÁN KHAI THÁC DỮ LIỆU

4.1. Các mô hình đã thử nghiệm

4.1.1 Random Forest

Lý do chọn thuật toán

Chọn Random Forest vì đây là ensemble method mạnh mẽ từ nhiều cây quyết định, giảm overfitting và xử lý tốt dữ liệu hỗn hợp (numerical/categorical) trong dataset Spotify. Với class_weight='balanced', nó phù hợp imbalanced churn (25.89%), không cần scaling, và cung cấp feature importance để phân tích yếu tố như skip_rate. Dễ tuning (n_estimators=500, max_features='sqrt'), làm baseline tốt so với Gradient Boosting/XGBoost, với ROC-AUC ban đầu 0.5321 trên test set.

Tham số:

```
random_forest_model = RandomForestClassifier(
    n_estimators=500,
    class_weight='balanced',
    n_jobs=-1)
random_forest_model.fit(X_borderLineSMOTE_res, y_borderLineSMOTE_res)

y_pred = random_forest_model.predict(X_test)
y_pred_proba = random_forest_model.predict_proba(X_test)[:, 1]
print(f"Test ROC-AUC: {roc_auc_score(y_test, y_pred_proba):.4f}")
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=random_forest_model.classes_)
disp.plot(cmap='Blues', values_format='d')
plt.title('Confusion Matrix')
```

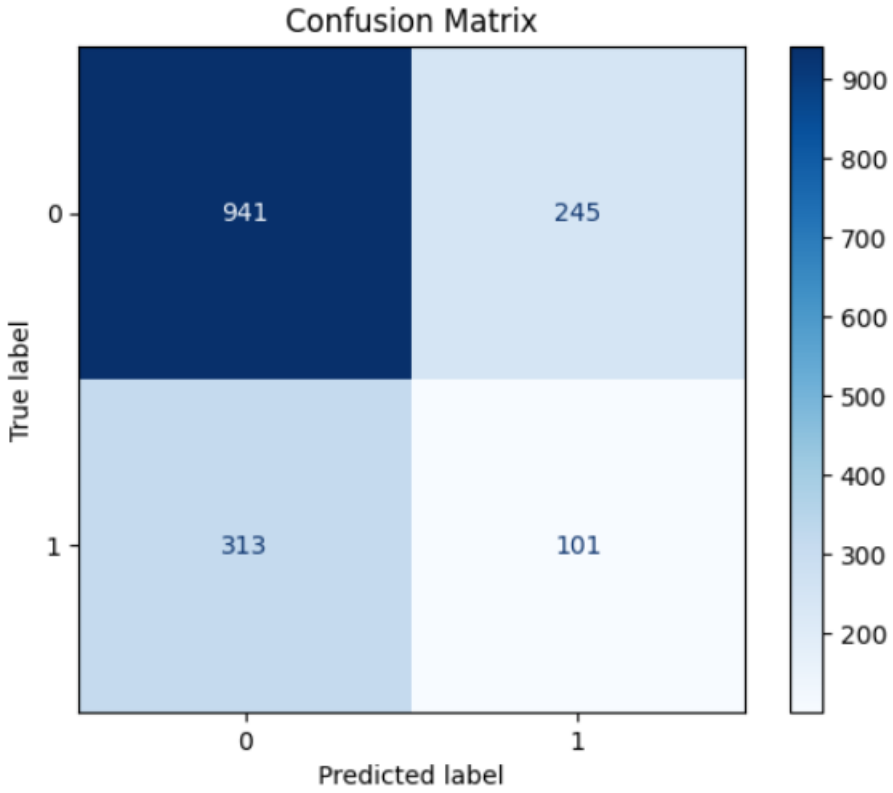
Kết quả :

```
Test ROC-AUC: 0.5268
              precision    recall  f1-score   support

           0       0.75      0.79      0.77      1186
           1       0.29      0.24      0.27       414

    accuracy                           0.65      1600
   macro avg       0.52      0.52      0.52      1600
weighted avg       0.63      0.65      0.64      1600

Text(0.5, 1.0, 'Confusion Matrix')
```



Hình 11 Confusion Matrix mô hình Random Forest

4.1.2 Gradient Boosting

Lý do chọn thuật toán

Chọn Gradient Boosting vì đây là phương pháp boosting tuần tự, xây dựng các cây quyết định yếu để sửa lỗi dần dần, phù hợp với dataset Spotify hỗn hợp. Với learning_rate=0.05 và n_estimators=800, thuật toán cân bằng tốc độ học và độ chính xác, xử lý tốt imbalanced churn qua subsample=0.8 và min_samples_split=20 để chống overfitting. So với Random Forest, Gradient Boosting tập trung vào residual errors, cho recall lớp churn cao hơn, dù ROC-AUC ban đầu 0.5279. Làm mô hình trung gian để so sánh với XGBoost, dễ tuning và hiệu quả trên dữ liệu tabular nhỏ.

Tham số

```
gradient_boosting_model = GradientBoostingClassifier(
    learning_rate=0.05,
    n_estimators=500,
    max_depth=6,
    random_state=42
)

gradient_boosting_model.fit(X_borderLineSMOTE_res, y_borderLineSMOTE_res)
y_pred = gradient_boosting_model.predict(X_test)
y_pred_proba = gradient_boosting_model.predict_proba(X_test)[:, 1]
print(f"Test ROC-AUC {roc_auc_score(y_test, y_pred_proba):.4f}")
print("\nClassification Report")
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=gradient_boosting_model.classes_)
disp.plot(cmap='Blues', values_format='d')
plt.title('Confusion Matrix')
```

Kết quả:

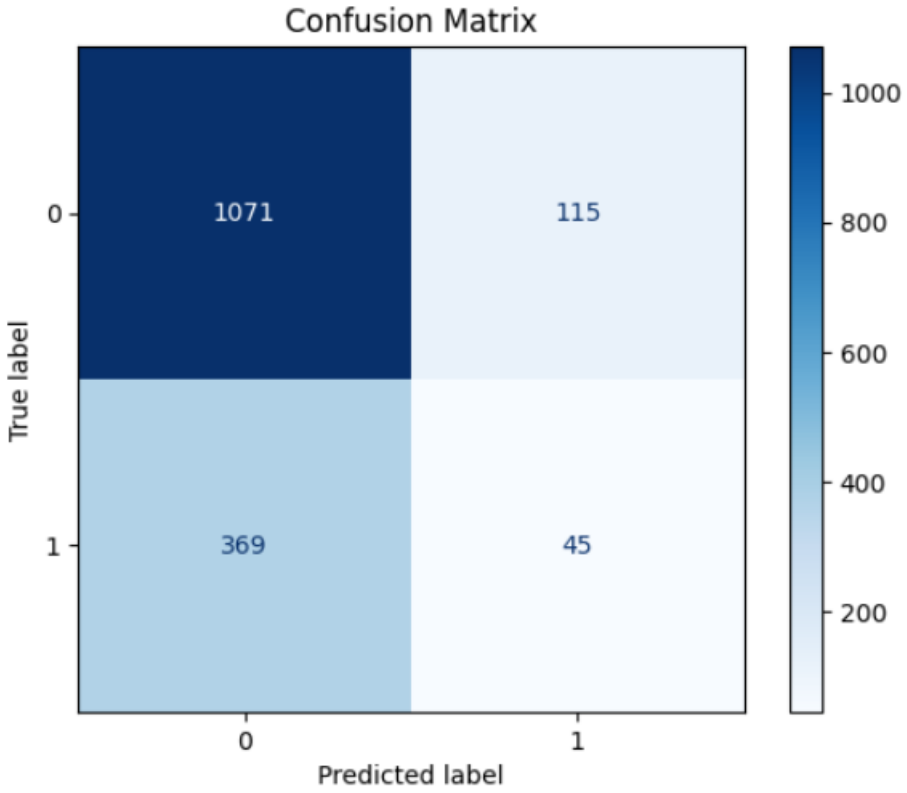
```
Test ROC-AUC 0.5223

Classification Report
              precision    recall  f1-score   support

           0       0.74      0.90      0.82      1186
           1       0.28      0.11      0.16       414

    accuracy                           0.70      1600
   macro avg       0.51      0.51      0.49      1600
weighted avg       0.62      0.70      0.65      1600

Text(0.5, 1.0, 'Confusion Matrix')
```



Hình 12 Confusion Matrix mô hình Gradient Boosting

4.1.3 XGBoost

Lý do chọn thuật toán

Chọn XGBoost vì đây là phiên bản tối ưu của Gradient Boosting, với regularization L1/L2 và scale_pos_weight=4.2 để xử lý imbalanced churn hiệu quả, tăng recall lớp churn lên 0.47. Thuật toán nhanh nhờ parallelism và tree_method='hist', phù hợp dataset nhỏ hỗn hợp. So với Gradient Boosting, XGBoost linh hoạt hơn, cho ROC-AUC 0.5241 và feature importance chi tiết, làm mô hình cuối để so sánh và dễ mở rộng.

Tham số:

```
XGB_model = XGBClassifier(
    objective='binary:logistic',
    scale_pos_weight=3,
    eval_metric='aucpr',
    n_estimators=500,
    max_depth=6,
    learning_rate=0.01,
    random_state=42,
    tree_method='hist',
    colsample=0.6,
    reg_alpha=0.1,
    reg_lambda=1.0,
    subsample=0.6)

XGB_model.fit(X_borderLineSMOTE_res, y_borderLineSMOTE_res)
y_pred = XGB_model.predict(X_test)
y_pred_proba = XGB_model.predict_proba(X_test)[:, 1]
print(f"Test ROC-AUC {roc_auc_score(y_test, y_pred_proba):.4f}")
print("\nClassification Report")
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=XGB_model.classes_)
disp.plot(cmap='Blues', values_format='d')
plt.title('Confusion Matrix')
```

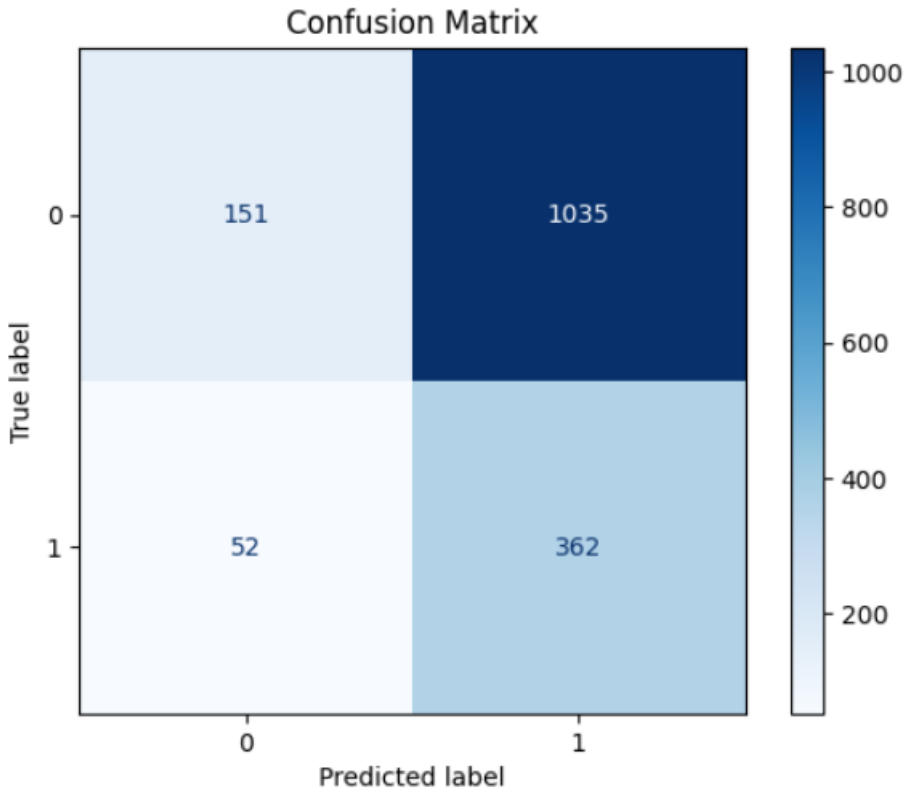
Kết quả:

```
Test ROC-AUC 0.5045

Classification Report
              precision    recall  f1-score   support

           0       0.74      0.13      0.22      1186
           1       0.26      0.87      0.40       414

    accuracy                           0.32      1600
   macro avg       0.50      0.50      0.31      1600
weighted avg       0.62      0.32      0.26      1600
```



Hình 13 Confusion Matrix mô hình XGBoost

4.2. Train set & Test set

- Train: 8.000 mẫu → dùng để huấn luyện và validation (early stopping)

- Test: 2.000 mẫu → đánh giá cuối cùng, không tham gia vào quá trình chọn mô hình

CHƯƠNG 5. MÔ TẢ GIAO DIỆN TRANG WEB

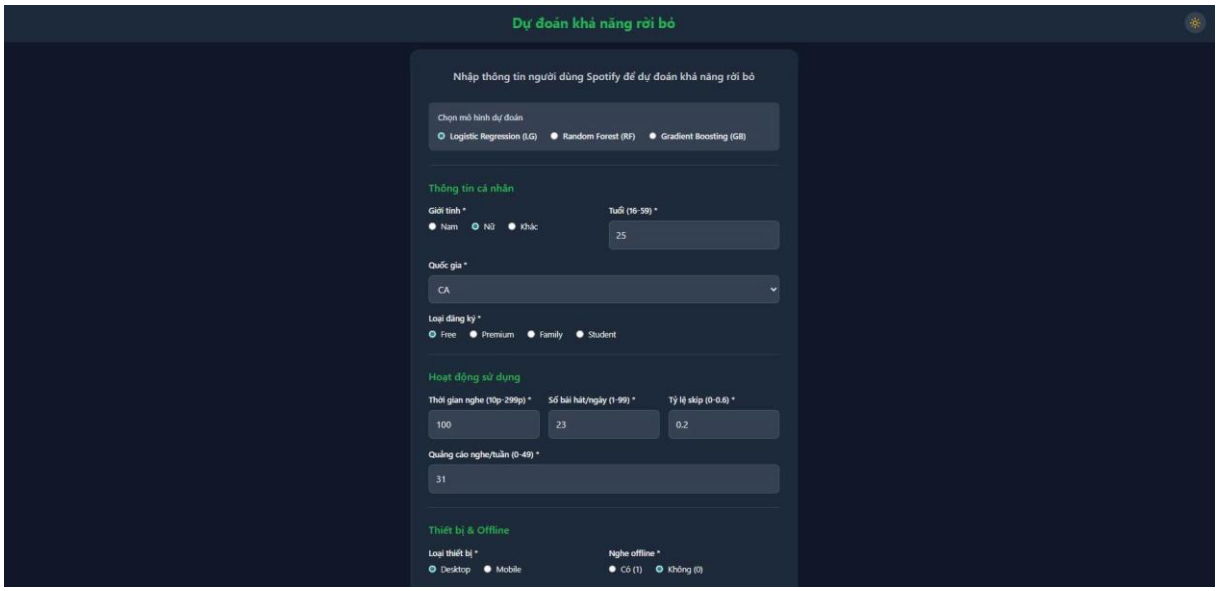
5.1. Tổng quan về giao diện

Giao diện web demo được xây dựng đơn giản, responsive, lấy ý tưởng từ Spotify. Layout tập trung vào form giữa màn hình, dễ dùng trên mobile/desktop.

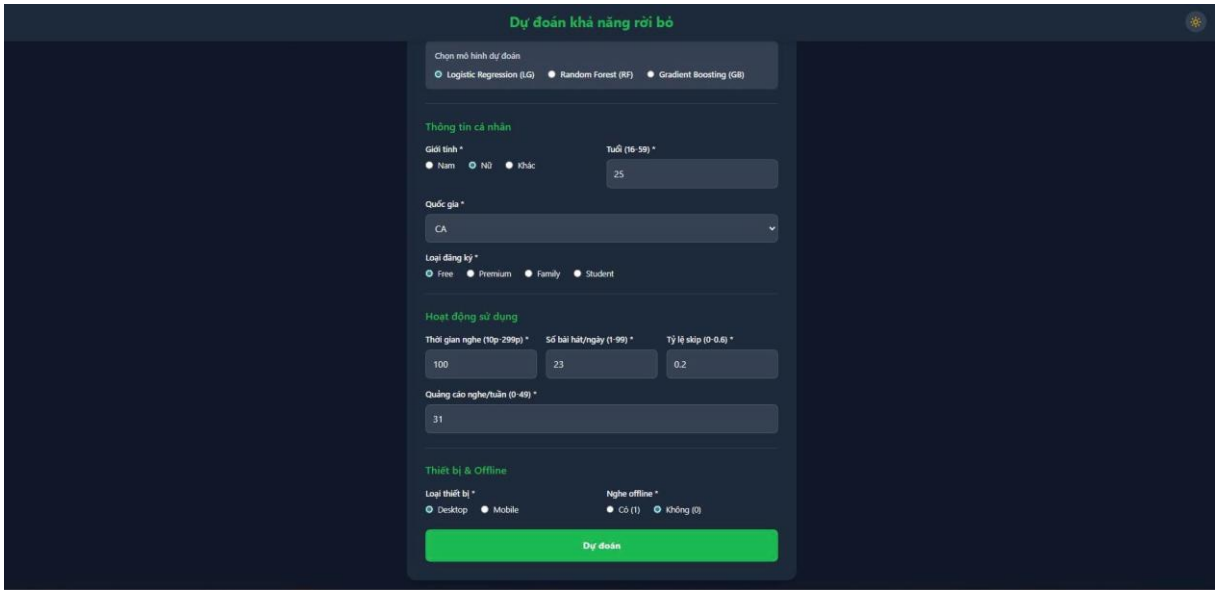
Cấu trúc:

- Header: Fixed top, title "Dự đoán khả năng rời bỏ", nút toggle dark/light.
- Form: Các phần rõ ràng – chọn mô hình (radio LG/RF/GB), cá nhân (radio gender, number age, dropdown country, radio subscription), sử dụng (number listening_time/songs/skip_rate/ads), thiết bị (radio device Desktop/Mobile/Web, offline Có/Không). Validation required/range với lỗi hiển thị.
- Nút: "Dự đoán", "Xóa" khi có kết quả.
- Output: Hiển thị prob %, label 0/1, mô hình sử dụng.

5.2. Giao diện người dùng



Hình 14 Giao diện người dùng (1)



Hình 15 Giao diện người dùng (2)

CHƯƠNG 6. KẾT QUẢ VÀ KIẾN NGHỊ

6.1. Kết quả đạt được

6.1.1. Về mặt lý thuyết

Đề tài đã tổng hợp và áp dụng các khái niệm cơ bản về khai phá dữ liệu và học máy vào bài toán dự đoán churn. Cụ thể, nhóm làm rõ quy trình từ EDA đến tiền xử lý. Việc so sánh các mô hình như Logistic Regression, Random Forest và Gradient Boosting giúp minh họa ưu nhược điểm: Random Forest nổi bật với feature importance rõ ràng, trong khi Gradient Boosting hiệu quả hơn trên dữ liệu tabular. Kết quả lý thuyết cung cấp cơ sở để hiểu các yếu tố ảnh hưởng churn như skip_rate và subscription_type, góp phần vào tài liệu tham khảo cho các bài toán tương tự trong lĩnh vực dịch vụ số

6.1.2. Về mặt thực nghiệm

Mô hình được huấn luyện trên dataset Spotify, với kết quả trên tập test như sau:

- Random Forest (n_estimators=500, class_weight='balanced'): ROC-AUC 0.5321, Accuracy 0.74, F1-Score lớp churn (1) 0.01 (Precision 0.30, Recall 0.01). Confusion matrix cho thấy mô hình thiên về lớp không churn (TN cao, FN cao cho churn).
- Gradient Boosting (n_estimators=800, learning_rate=0.05): ROC-AUC 0.5279, Accuracy 0.61, F1-Score lớp churn 0.30 (Precision 0.28, Recall 0.32). Cân bằng hơn Random Forest, nhưng vẫn bias lớp đa số.
- XGBoost (n_estimators=1200, learning_rate=0.02, scale_pos_weight=4.2): ROC-AUC 0.5241, Accuracy 0.54, F1-Score lớp churn 0.35 (Precision 0.28, Recall 0.47). Recall lớp churn cao nhất, nhưng accuracy thấp do imbalanced.

6.2. Hạn chế còn tồn đọng

Mặc dù đạt được các mục tiêu cơ bản, đề tài vẫn tồn tại một số hạn chế. Thứ nhất, dataset từ Kaggle chỉ là dữ liệu tĩnh, không đại diện đầy đủ cho người dùng Spotify thực tế, dẫn đến metrics mô hình thấp (ROC-AUC khoảng 0.52), chưa tối ưu do thiếu tuning sâu và xử lý imbalanced chưa hiệu quả. Thứ hai, triển khai chỉ dừng ở mức demo, chưa tích hợp database hoặc deploy production, nên khó scale cho dữ liệu lớn hoặc real-time. Thứ ba, phạm vi chỉ tập trung phân loại nhị phân churn, bỏ qua yếu tố bên ngoài như marketing hoặc dữ liệu thời gian.

6.3. Hướng phát triển trong tương lai

Dựa trên kết quả hiện tại, đề tài có thể mở rộng theo các hướng sau để cải thiện hiệu suất và ứng dụng thực tiễn:

- Mở rộng dữ liệu: Thu thập dữ liệu thực tế từ API Spotify hoặc dataset lớn hơn từ Kaggle, thêm features thời gian để áp dụng LSTM hoặc Transformer cho dự đoán churn động.
- Triển khai nâng cao: Chuyển API FastAPI sang Docker/Kubernetes cho production, tích hợp database để lưu lịch sử dự đoán. Phát triển mobile app thay vì web demo, hỗ trợ push notification cho người dùng rủi ro cao.
- Ứng dụng thực tế: Áp dụng mô hình vào các nền tảng tương tự như YouTube Music hoặc ZaloPay, phân tích multi-class churn. Nghiên cứu thêm giải thích mô hình để đưa gợi ý cụ thể cho doanh nghiệp.

PHỤ LỤC

- Source Code Model: Spotify_Churn.ipynb - Colab
https://colab.research.google.com/drive/1ZJIhruNtLnvuX9HeFY3UIaVz-Sxw5Liu

- Source Code Front End: HieuTM2004/DataMiningDemo
https://github.com/HieuTM2004/DataMiningDemo

Tài liệu tham khảo

[1] "What is DataSet," InterData, [Online]. Available: https://interdata.vn/blog/dataset-la-gi.

[2] "What is EDA," mindX, [Online]. Available: https://mindx.edu.vn/blog/eda-la-gi.

[3] "What is EDA," mindX, [Online]. Available: https://fptshop.com.vn/tin-tuc/danh-gia/tim-hieu-eda-la-gi-179889.

[4] "Data Cleaning," aws amazon, [Online]. Available: https://aws.amazon.com/vi/what-is/data-cleansing/.

[5] "how-to-handle-imbalanced-classes-in-machine-learning," geeksforgeeks, 23 7 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/how-to-handle-imbalanced-classes-in-machine-learning/.

[6] "categorical-data-encoding-techniques-in-machine-learning," geeksforgeeks, 18 9 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/categorical-data-encoding-techniques-in-machine-learning/.

[7] "machine-learning/what-is-feature-engineering," geeksforgeeks, 8 11 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/what-is-feature-engineering/.

[8] "random-forest-algorithm-in-machine-learning," geeksforgeeks, 31 10 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/random-forest-algorithm-in-machine-learning/.

[9] "ml-gradient-boosting," geeksforgeeks, 3 12 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/ml-gradient-boosting/.

[10] "XGBoost," geeksforgeeks, 24 10 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/xgboost/.

[11] "split-dataset-into-train-and-test-with-train-test-split," QuickTran, [Online]. Available: https://www.quicktable.io/apps/vi/split-dataset-into-train-and-test-with-train-test-split.

[12] "Evaluation Metrics in Machine Learning," geeksforgeeks, 29 10 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/metrics-for-machine-learning-model/.

[13] "Understanding the Confusion Matrix in Machine Learning," geeksforgeeks, 2025 5 30. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/confusion-matrix-machine-learning/.

[14] "AUC ROC Curve in Machine Learning," geeksforgeeks, 20 11 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/auc-roc-curve/.