Phân Tích Và Dự Đoán Khách Hàng Rời Bỏ Dịch Vụ Ngân Hàng Theo Thời Gian Thực

Giảng viên hướng dẫn: TS. Đỗ Trọng Hợp

Thực hiện: Nhóm

Huỳnh Trung Hiếu1, Nguyễn Tấn Đạt1, Phan Tấn Cảnh1

Đại Học Công Nghệ Thông Tin

Đại Học Quốc Gia Thành Phố Hồ Chí Minh Thành Phố Hồ Chí Minh, Việt Nam

22540006, 22540003, 22540002

**Tóm tắt.** Trong thời đại hiện nay, sự phát triển mạnh mẽ của các hệ thống xử lý dữ liệu lớn, nền tảng tính toán phân tán và học máy đã thúc đẩy việc xây dựng các ứng dụng thông minh, hỗ trợ doanh nghiệp trong việc dự đoán và phân tích hành vi khách hàng. Một trong những bài toán quan trọng là dự đoán khách hàng có nguy cơ rời bỏ dịch vụ, giúp các tổ chức tài chính và ngân hàng có chiến lược giữ chân khách hàng hiệu quả hơn. Trong phạm vi đồ án này, nhóm đã nghiên cứu và xây dựng hệ thống phân tích và dự đoán khách hàng rời bỏ ngân hàng theo thời gian thực, áp dụng Apache Spark và Kafka. Hệ thống được thiết kế nhằm phục vụ mục đích học tập, nghiên cứu, đồng thời có khả năng mở rộng để ứng dụng thực tế trong ngành tài chính – ngân hàng. Kiến trúc hệ thống gồm các thành phần chính: Kafka Producer mô phỏng dữ liệu khách hàng theo thời gian thực từ tập dữ liệu lịch sử; Kafka Broker tiếp nhận và lưu trữ dữ liệu stream; Spark Streaming đọc dữ liệu từ Kafka, xử lý tiền xử lý và trích xuất đặc trưng; Mô hình học máy được huấn luyện trên tập dữ liệu lịch sử để dự đoán khả năng rời bỏ của khách hàng; Streamlit và Plotly được sử dụng để hiển thị kết quả phân tích dưới dạng bảng và biểu đồ trực quan. Bên cạnh đó, hệ thống được thiết kế theo kiến trúc module hóa, giúp dễ dàng tích hợp thêm các tính năng mới, mở rộng quy mô hoặc áp dụng cho các lĩnh vực khác. Nhà quản trị có thể thực hiện các thao tác thu phát luồng dữ liệu, theo dõi dự đoán, hiển thị kết quả trực tiếp trên giao diện web hoặc xuất dữ liệu báo cáo phục vụ phân tích.

**Từ khóa:** Phân tích churn, Kafka, Apache Spark, Học máy, Streamlit, Plotly, Dự đoán khách hàng rời bỏ

# GIỚI THIỆU

Hiện nay, trong lĩnh vực tài chính – ngân hàng, việc dự đoán khách hàng có nguy cơ rời bỏ dịch vụ là một bài toán quan trọng, giúp doanh nghiệp hiểu rõ hành vi khách hàng và đưa ra các chiến lược giữ chân phù hợp. Nhiều nghiên cứu đã chỉ ra rằng việc dự đoán và ngăn chặn khách hàng rời bỏ có thể giúp ngân hàng giảm thiểu tổn thất doanh thu và tối ưu hóa chiến lược kinh doanh [1]. Sự phát triển mạnh mẽ của dữ liệu lớn (Big Data) và các nền tảng xử lý phân tán đã mở ra nhiều cơ hội để triển khai các hệ thống phân tích rời bỏ khách hàng theo thời gian thực, giúp các tổ chức tài chính phản ứng kịp thời trước xu hướng rời bỏ của khách hàng.

Trong bài viết này, nhóm đề xuất ứng dụng phân tích và dự đoán khách hàng rời bỏ ngân hàng theo thời gian thực, sử dụng Apache Spark và Kafka để xử lý dữ liệu streaming. Hệ thống đề xuất bao gồm hai giai đoạn chính: huấn luyện mô hình học máy và triển khai hệ thống phân tích thời gian thực. Trong đó, mô hình học máy được huấn luyện trên tập dữ liệu lịch sử có nguồn từ Kaggle [2], bao gồm thông tin giao dịch và đặc điểm khách hàng, nhằm tìm ra các yếu tố ảnh hưởng đến quyết định rời bỏ. Sau khi lựa chọn mô hình có độ chính xác cao nhất, hệ thống sẽ sử dụng mô hình này để dự đoán trên luồng dữ liệu giả lập, được mô phỏng nhằm phản ánh hành vi khách hàng trong thời gian thực. Kết quả dự đoán sẽ được tổng hợp và hiển thị trên giao diện web thông qua Streamlit và Plotly, giúp người dùng dễ dàng theo dõi và phân tích dữ liệu. Việc triển khai hệ thống này không chỉ giúp kiểm chứng tính khả thi của việc ứng dụng công nghệ xử lý dữ liệu lớn trong dự đoán rời bỏ khách hàng, mà còn tạo tiền đề để phát triển các hệ thống thực tế trong tương lai. Bên cạnh đó, hệ thống có thể mở rộng để tích hợp thêm các nguồn dữ liệu khác, cải thiện độ chính xác của mô hình và nâng cao khả năng dự đoán.

Bài viết này được cấu trúc như sau: Phần 1 giới thiệu tổng quan về bài toán rời bỏ khách hàng và hệ thống đề xuất. Phần 2 mô tả các nghiên cứu liên quan đến hệ thống. Phần 3 trình bày thiết kế và triển khai hệ thống, bao gồm kiến trúc, công nghệ sử dụng và mô hình dự đoán. Phần 4 cung cấp kết quả thực nghiệm, so sánh độ chính xác giữa các mô hình học máy áp dụng vào hệ thống. Cuối cùng, phần kết luận tổng kết những kết quả đạt được và hướng phát triển trong tương lai.

# CÁC NGHIÊN CỨU LIÊN QUAN

Các nghiên cứu trước đây đã áp dụng Machine Learning để phân tích hành vi khách hàng và dự đoán tỷ lệ rời bỏ trong ngành ngân hàng. Các mô hình như Decision Trees, Random Forest và Support Vector Machines đã được sử dụng để phân loại khách hàng có nguy cơ rời bỏ [3]. Ngoài ra, các yếu tố như tần suất giao dịch, loại hình dịch vụ và phản hồi khách hàng cũng đóng vai trò quan trọng trong việc dự đoán churn [4]. Tuy nhiên, hầu hết các nghiên cứu này chỉ dựa trên dữ liệu tĩnh mà chưa tận dụng phân tích theo thời gian thực. Một số công trình tập trung vào phân tích lịch sử giao dịch bằng cách áp dụng các phương pháp khai thác dữ liệu để xây dựng mô hình dự đoán churn [5], [6]. Các kỹ thuật như hồi quy logistic, k-nearest neighbor và các mô hình thống kê khác đã được sử dụng, nhưng chưa có sự đánh giá toàn diện trên nhiều tập dữ liệu khác nhau [7], [8]. Trong khi đó, các nghiên cứu về hệ thống dự đoán theo thời gian thực đã cho thấy hiệu quả của việc tích hợp Machine Learning và Big Data trong việc giám sát và phát hiện sớm khách hàng có nguy cơ rời bỏ [9], [10]. Việc triển khai các mô hình này vào hệ thống CRM giúp ngân hàng phản ứng kịp thời và cải thiện chiến lược giữ chân khách hàng [10].

# HỆ THỐNG ĐỀ XUẤT

Phần này trình bày thiết kế và triển khai hệ thống được đề xuất, gồm có hai phần chính: Phần 3.1 mô tả thiết kế và kiến trúc hệ thống, và Phần 3.2 trình bày việc hiện thực và triển khai hệ thống được đề xuất.

## Thiết kế và kiến trúc hệ thống

Hệ thống phân tích và dự đoán khách hàng rời bỏ ngân hàng theo thời gian thực được xây dựng qua hai giai đoạn chính. Giai đoạn đầu huấn luyện các mô hình Machine Learning (ML) và Deep Learning (DL) trên dữ liệu lịch sử từ Kaggle, bao gồm các đặc trưng như điểm tín dụng, tuổi, số dư tài khoản, cùng các đặc trưng suy luận, được tiền xử lý và huấn luyện với các thuật toán như Logistic Regression, Random Forest, SVM, KNN, GCN trên scikit-learn và Random Forest trên Spark ML; mô hình tối ưu nhất được lưu để dự đoán thời gian thực. Giai đoạn hai sử dụng dữ liệu giả lập từ tập test, gửi qua Kafka Topic 'producer', xử lý bằng Spark Streaming, dự đoán churn với mô hình đã huấn luyện, rồi lưu kết quả vào Kafka Topic 'prediction' và PostgreSQL, đồng thời trực quan hóa qua ứng dụng web Streamlit-Plotly với khả năng điều khiển luồng dữ liệu linh hoạt.

A diagram of a company

AI-generated content may be incorrect.

**Hình 1.** Kiến trúc hệ thống tổng quan.

Hệ thống các thành phần xử lý gồm có ba phần: Hệ thống xử lý offline, online, và ứng dụng web được mô tả như sau:

* + - ***Hệ thống xử lý offline:*** Hệ thống xử lý offline đảm nhận nhiệm vụ tiền xử lý dữ liệu và huấn luyện mô hình học máy, dựa trên tập dữ liệu lịch sử khách hàng của ngân hàng. Dữ liệu bao gồm các thông tin cá nhân như CustomerId, Geography, Gender, Age, Tenure, cùng các thông tin tài chính như CreditScore, Balance, NumOfProducts, HasCrCard, IsActiveMember và EstimatedSalary. Ngoài ra, tập dữ liệu cũng được mở rộng với các đặc trưng suy luận như BalanceSalaryRatio, CreditScoreAgeRatio, nhằm tăng cường khả năng phân biệt khách hàng rời bỏ và không rời bỏ. Trong phạm vi nghiên cứu này, nhóm triển khai hai hướng tiếp cận huấn luyện mô hình: (i) sử dụng scikit-learn để huấn luyện các mô hình truyền thống như SVM, Random Forest, GCN và KNN, và (ii) sử dụng Spark ML để huấn luyện mô hình Random Forest trong môi trường xử lý phân tán. Sau khi tối ưu hóa, các mô hình sẽ được lưu trữ để sử dụng trong hệ thống xử lý online. Các bước thực hiện như sau:
      * **Bước 1:** Chuẩn bị môi trường và nạp dữ liệu. Trước tiên, cần cấu hình môi trường làm việc với PyTorch, PySpark và Scikit-learn để đảm bảo quá trình tiền xử lý và huấn luyện mô hình diễn ra thuận lợi. Dữ liệu lịch sử giao dịch của khách hàng được nạp từ tệp CSV hoặc cơ sở dữ liệu vào hệ thống bằng Pandas hoặc Spark DataFrame (đối với Spark ML). Sau khi dữ liệu được tải lên, tiến hành kiểm tra lược đồ để xác định các kiểu dữ liệu, đảm bảo tính nhất quán và xử lý các giá trị thiếu.
      * **Bước 2:** Nhóm áp dụng các phương pháp tiền xử lý khác nhau tùy theo framework huấn luyện mô hình. Đối với scikit-learn, dữ liệu được chuẩn hóa bằng StandardScaler() để đưa tất cả các đặc trưng về cùng một khoảng giá trị. Đối với Spark ML, nhóm sử dụng VectorAssembler() để kết hợp các đặc trưng thành một vector duy nhất, đảm bảo tính tương thích với các thuật toán học máy trên Spark. Ngoài ra, do tập dữ liệu mất cân bằng, nhóm áp dụng phương pháp SMOTEN (Synthetic Minority Over-sampling Technique for Nominal Features) trong scikit-learn để cân bằng tỷ lệ giữa hai nhóm khách hàng rời bỏ và không rời bỏ**.**
      * **Bước 3:** Dữ liệu được chia thành 80% để huấn luyện và 20% để kiểm tra. Đối với scikit-learn, nhóm sử dụng train\_test\_split() để tạo tập huấn luyện và kiểm tra. Trong Spark ML, nhóm sử dụng phương thức randomSplit() của Spark DataFrame để đảm bảo phân chia ngẫu nhiên với tỷ lệ tương tự.
      * **Bước 4:** Huấn luyện mô hình bằng cách áp dụng các thuật toán học máy và học sâu. Với scikit-learn, các mô hình phổ biến được sử dụng trong bài toán dự đoán khách hàng rời bỏ bao gồm logistic regression (LR), random forest (RF), support vector machine (SVM), k-nearest neighbors (KNN) và graph convolution network (GCN). Mô hình GCN được thiết kế để xử lý dữ liệu có cấu trúc đồ thị bằng cách tận dụng thông tin từ các nút lân cận. Trong nghiên cứu này, dữ liệu được biểu diễn dưới dạng đồ thị thông qua k-nearest neighbors (KNN) graph, trong đó mỗi điểm dữ liệu (nút) được kết nối với k nút gần nhất. Với GCN, mô hình bao gồm hai lớp GCNConv, với lớp đầu tiên có nhiệm vụ trích xuất đặc trưng từ dữ liệu đầu vào và lớp thứ hai thực hiện phân loại. Hàm kích hoạt ReLU được sử dụng để tăng khả năng biểu diễn phi tuyến của mô hình, trong khi đầu ra được chuẩn hóa bằng log-softmax để tối ưu hóa bài toán phân loại. Trong khi đó, thuật toán KNN là một phương pháp học máy không tham số, hoạt động dựa trên nguyên lý láng giềng gần nhất. Trong nghiên cứu này, KNN được áp dụng với k = 9, sử dụng khoảng cách Euclidean để xác định các điểm gần nhất và trọng số khoảng cách để tăng độ chính xác của mô hình. Mô hình được huấn luyện trên tập dữ liệu mở rộng bằng SMOTE, sau đó được đánh giá bằng các chỉ số độ chính xác (accuracy), độ chính xác (precision), độ nhạy (recall) và F1-score. Mô hình logistic regression (LR) là một phương pháp phân loại tuyến tính dựa trên hàm sigmoid, thường được sử dụng cho các bài toán phân loại nhị phân. Trong nghiên cứu này, mô hình được huấn luyện trên tập dữ liệu đã cân bằng và kiểm định trên tập thử nghiệm. Các chỉ số đánh giá bao gồm độ chính xác, precision, recall và F1-score, đồng thời biểu diễn kết quả qua ma trận nhầm lẫn (Confusion Matrix). Tiếp theo, mô hình random forest (RF) là một phương pháp học tập tổ hợp dựa trên tập hợp nhiều cây quyết định (decision trees), giúp giảm thiểu hiện tượng quá khớp (overfitting) và cải thiện độ chính xác. Trong nghiên cứu này, RF được huấn luyện trên tập dữ liệu đã cân bằng và đánh giá hiệu suất thông qua các chỉ số accuracy, precision, recall, F1-score. Kết quả phân loại được trực quan hóa bằng ma trận nhầm lẫn để đánh giá độ chính xác của mô hình trên từng lớp. Cuối cùng, mô hình support vector machine (SVM) sử dụng hàm kernel đa thức (Polynomial Kernel) để ánh xạ dữ liệu lên không gian đặc trưng có chiều cao hơn, giúp cải thiện khả năng phân tách phi tuyến. Mô hình được huấn luyện và đánh giá trên tập dữ liệu đã cân bằng, với đầu ra bao gồm các chỉ số accuracy, precision, recall, F1-score và biểu diễn trực quan bằng ma trận nhầm lẫn. Mô hình SVM cho thấy hiệu suất khá tốt trong bài toán phân loại khách hàng có nguy cơ rời bỏ. Với Spark ML, nhóm triển khai Random Forest Classifier và sử dụng Cross Validation để tối ưu hóa số lượng cây quyết định (numTrees = 50, 100) và độ sâu (maxDepth = 5, 10). Tiêu chí phân tách được mặc định là Gini impurity, giúp đánh giá mức độ thuần khiết của các node trong cây quyết định.
      * **Bước 5:** Đánh giá hiệu suất để chọn ra mô hình tối ưu nhất. Các thước đo quan trọng được sử dụng bao gồm Accuracy (độ chính xác), Precision (độ chính xác của dự đoán dương), Recall (khả năng phát hiện khách hàng rời bỏ), và F1 Score (điểm trung bình điều hòa giữa Precision và Recall). Những chỉ số này giúp so sánh hiệu quả của các mô hình và xác định mô hình nào hoạt động tốt nhất trên tập kiểm tra.Trong bài toán dự đoán khách hàng rời bỏ, Recall thường là một yếu tố quan trọng vì việc phát hiện đúng các khách hàng có khả năng rời bỏ giúp ngân hàng có thể đưa ra các chiến lược giữ chân kịp thời. Do đó, cần xem xét kỹ sự đánh đổi giữa Precision và Recall để chọn mô hình phù hợp. Sau khi phân tích kết quả, mô hình có hiệu suất tốt nhất sẽ được lựa chọn để triển khai trong hệ thống online, sẵn sàng dự đoán khách hàng rời bỏ trong thời gian thực**.**
      * **Bước 6:** Sau khi đánh giá hiệu suất, mô hình có độ chính xác cao nhất được lưu để sử dụng trong hệ thống xử lý online. Mô hình scikit-learn được lưu dưới dạng .pkl và .pth thông qua joblib.dump(), trong khi mô hình Spark ML được lưu bằng phương thức write().overwrite().save("path/to/model"). Việc lưu trữ mô hình theo định dạng này đảm bảo khả năng tái sử dụng mà không cần huấn luyện lại từ đầu khi tích hợp vào hệ thống Spark Streaming.

A diagram of a computer system

AI-generated content may be incorrect.

**Hình 2.** Hệ thống xử lý offline.

* + - ***Hệ thống xử lý online*:** Thành phần xử lý online, được thực hiện gồm các bước như sau:
      * **Bước 1:** Luồng dữ liệu khách hàng mới được thu thập theo thời gian thực từ nguồn giả lập bằng dữ liệu được lấy từ tập test và gửi vào Kafka. Các producer sẽ liên tục ghi nhận thông tin giao dịch, hành vi khách hàng và các chỉ số quan trọng khác, sau đó đẩy dữ liệu vào Kafka topic. Điều này đảm bảo dòng dữ liệu được truyền tải một cách liên tục và có thể mở rộng dễ dàng.
      * **Bước 2:** Apache Spark Streaming sẽ tiêu thụ dữ liệu từ Kafka để phân tích và xử lý. Trước khi sử dụng cho dự đoán, dữ liệu sẽ trải qua các bước tiền xử lý như chuyển đổi định dạng, làm sạch dữ liệu, cập nhật schema, trích xuất các thuộc tính quan trọng và chuẩn hóa để phù hợp với mô hình đã huấn luyện. Những bước này giúp đảm bảo rằng dữ liệu đầu vào nhất quán và có chất lượng cao, tương tự như quy trình xử lý dữ liệu trong giai đoạn huấn luyện offline.
      * **Bước 3:** Sau khi dữ liệu đã được tiền xử lý, mô hình Machine Learning đã huấn luyện có độ chính xác cao nhất sẽ được áp dụng để phân tích và dự đoán khả năng khách hàng rời bỏ ngân hàng theo thời gian thực. Kết quả dự đoán sẽ được gửi trở lại Kafka topic 'prediction' để tiếp tục xử lý đồng thời dữ liệu này cũng được ghi vào cơ sở dữ liệu PostgreSQL nhằm hỗ trợ lưu trữ lâu dài và phục vụ các truy vấn sau này.

A diagram of a software development process

AI-generated content may be incorrect.

**Hình 3.** Hệ thống xử lý online.

* + - ***Ứng dụng web:*** Ứng dụng Streamlit đọc dữ liệu trực tiếp từ Kafka topic 'prediction' để hiện thị lên giao diện web, và dùng Plotly để tạo các biểu đồ tròn, cột và bảng, trực quan hóa các dữ liệu báo cáo và thống kê. Ngoải ra, từ giao diện web, có thể gọi dịch vụ xử lý để bắt đầu hoặc dừng thu phát phân tích luồng dữ liệu.

## Triển Khai Hệ thống

Trong phần hiện thực và triển khai, hệ thống xử lý offline được thực hiện trên môi trường ảo tạo trên Window, sử dụng Apache Spark 3.5.2, Java 8(1.8.0\_341), Python 3.8 và một số thư viện khác(chi tiết thư viện trong file requirements.txt). Các bước thực hiện đã được đề cập chi tiết trong phần 3.1. Mặt khác, hệ thống xử lý online và ứng dụng web, và các công cụ quản trị khác, được hiện thực trên máy chủ Window, Java 8(1.8.0\_341), Python 3.8, Spark 3.5.2, Kafka 2.12-3.9.0, Streamlit 1.40.1, Plotly 6.0.0, v.v. và phần cứng CPU core I5-12400f , bộ nhớ 16 GB, ổ cứng SSD 256 GB. Hệ thống chương trình có cấu trúc như hình sau:

A screenshot of a computer screen

AI-generated content may be incorrect.

**Hình 4.** Cấu trúc chương trình.

* **Công cụ quản trị hệ thống:** Giám sát trạng thái, khởi động và dừng các dịch vụ của Kafka Server và Zookeeper. Hỗ trợ tạo, xóa, liệt kê và kiểm tra nội dung các topic trong Kafka. Thiết lập biến môi trường, cài đặt dịch vụ và các thư viện cần thiết cho hệ thống.
* **Dịch vụ xử lý:** Thiết lập kết nối với Apache Spark và Kafka để thu thập và xử lý dữ liệu khách hàng theo thời gian thực. Thực hiện chuyển đổi, trích xuất các thuộc tính quan trọng, cập nhật lược đồ dữ liệu, làm sạch và tiền xử lý dữ liệu. Áp dụng mô hình học máy đã được huấn luyện từ hệ thống xử lý offline, lưu trữ trong thư mục “models”, để phân tích và dự đoán khả năng khách hàng rời bỏ dịch vụ. Kết quả dự đoán được xuất ra console, bộ nhớ, tệp hoặc Kafka. Hệ thống streaming liên tục nhận dữ liệu mô phỏng theo thời gian thực, đẩy vào Kafka để xử lý.
* **Dữ liệu:** Các báo cáo và thống kê kết quả phân tích được lưu trữ vào database “churn\_prediction” trên Postgres để phục vụ theo dõi và đánh giá sau này.
* **Dịch vụ web:** Hệ thống tải dữ liệu từ Kafka topic để hiển thị báo cáo và thống kê trên giao diện web bằng Streamlit và Plotly. Bên cạnh việc trực quan hóa dữ liệu, giao diện web còn cung cấp các nút điều khiển “Start Streaming” và “Stop Streaming”, cho phép người dùng khởi động hoặc dừng luồng xử lý dữ liệu theo thời gian thực. Khi kích hoạt, hệ thống sẽ kết nối đến các dịch vụ xử lý, nhận dữ liệu từ Kafka và cập nhật báo cáo ngay trên giao diện một cách liên tục.

# THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ

## Chuẩn bị dữ liệu

**Bảng 1.** Dữ liệu huấn luyện và kiểm tra.

|  |  |  |
| --- | --- | --- |
| **Phân loại** | **Huấn luyện** | **Kiểm tra** |
| 0 (không rời bỏ) | 6370 | 1593 |
| 1 (rời bỏ) | 1630 | 407 |

A graph with numbers and a bar

AI-generated content may be incorrect.

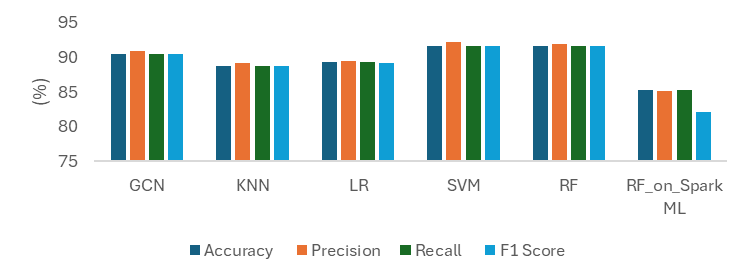
**Hình 5.** Dữ liệu huấn luyện và kiểm tra.

## Đánh giá kết quả

Phần này trình bày việc đánh giá kết quả và so sánh độ chính xác của các mô hình dùng các thuật toán logistic regression (LR), random forest (RF), random forest on SparkML, support vector machine (SVM), k-nearest neighbors (KNN) và graph convolution network (GCN). Kết quả thực nghiệm thu được ở bảng sau:

**Bảng 2.** Độ chính xác của các mô hình học máy.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Accuracy (%)** | **Precision (%)** | **Recall (%)** | **F1 Score (%)** |
| GCN | 90.52 | 90.87 | 90.52 | 90.50 |
| KNN | 88.80 | 89.13 | 88.80 | 88.77 |
| LR | 89.27 | 89.43 | 89.27 | 89.26 |
| SVM | 91.69 | 92.19 | 91.69 | 91.67 |
| RF | 91.66 | 91.99 | 91.66 | 91.64 |
| RF\_on\_Spark ML | 85.56 | 85.56 | 85.56 | 85.56 |



**Hình 6.** Độ chính xác của các mô hình học máy.

Kết quả ở Bảng 2 cho thấy, mô hình sử dụng thuật toán Support Vector Machine (SVM) trên scikit-learn đạt độ chính xác cao nhất, với accuracy = 91.69%. Điều này cho thấy SVM có khả năng phân tách dữ liệu rất tốt, đặc biệt khi sử dụng kernel đa thức (Polynomial Kernel) giúp cải thiện khả năng phân loại khách hàng rời bỏ. Tuy nhiên, việc triển khai SVM trên scikit-learn vẫn tồn tại một số hạn chế: (1) Thời gian huấn luyện dài hơn trên tập dữ liệu lớn, do SVM có độ phức tạp tính toán cao khi số lượng mẫu tăng; (2) Khả năng mở rộng kém hơn so với Spark ML, vì scikit-learn không hỗ trợ xử lý dữ liệu phân tán mà chỉ chạy trên một máy đơn lẻ; (3) Việc triển khai trong hệ thống Spark Streaming cần thêm bước chuyển đổi dữ liệu, vì scikit-learn không tích hợp trực tiếp với Spark ML, đòi hỏi phải load mô hình .pkl vào Spark Streaming để thực hiện dự đoán theo thời gian thực. Do đó, mặc dù đạt độ chính xác cao, SVM trên scikit-learn có thể không phải là lựa chọn tối ưu nếu hệ thống mở rộng với dữ liệu rất lớn.

Để khắc phục những hạn chế này, nhóm đã thử nghiệm mô hình Random Forest trên Spark ML (RF\_SparkML) nhằm tận dụng khả năng xử lý dữ liệu lớn trong môi trường phân tán. Kết quả thực nghiệm cho thấy RF\_SparkML đạt accuracy = 85.56%, thấp hơn 6.13% so với SVM trên scikit-learn (91.69%). Mặc dù độ chính xác không cao bằng, nhưng RF\_SparkML có lợi thế về tốc độ huấn luyện nhanh hơn trên tập dữ liệu lớn, nhờ khả năng tính toán song song của Spark ML.

Điều này cho thấy sự đánh đổi giữa độ chính xác và khả năng mở rộng: mô hình SVM trên scikit-learn cho kết quả tốt hơn nhưng bị giới hạn về hiệu suất khi dữ liệu tăng, trong khi RF\_SparkML có thể xử lý dữ liệu lớn hơn nhưng độ chính xác thấp hơn do các hạn chế trong quá trình tiền xử lý và tối ưu tham số.

Để có cái nhìn toàn diện hơn, nhóm đã triển khai đồng thời cả hai mô hình trong hệ thống dự đoán thời gian thực, nhằm đánh giá không chỉ hiệu suất trên tập dữ liệu tĩnh mà còn kiểm tra khả năng hoạt động trong môi trường xử lý streaming. Cả SVM trên scikit-learn và RF trên SparkML đều được tích hợp vào Spark Streaming, nơi dữ liệu liên tục được gửi từ Kafka đến mô hình để đưa ra dự đoán ngay lập tức. Kết quả thử nghiệm trong môi trường streaming tiếp tục khẳng định sự khác biệt giữa hai mô hình. SVM trên scikit-learn duy trì độ chính xác cao, nhưng lại gặp hạn chế về tốc độ xử lý, do dữ liệu phải được chuyển đổi định dạng và xử lý trên một máy đơn lẻ. Trong khi đó, RF trên SparkML có độ trễ thấp hơn khi xử lý dữ liệu lớn, nhờ khả năng chạy phân tán trên cụm Spark. Tuy nhiên, mô hình này lại có tỷ lệ dự đoán sai cao hơn, đặc biệt trong các trường hợp dữ liệu mất cân bằng.

Điều này cho thấy không có mô hình nào hoàn toàn vượt trội trong mọi khía cạnh. Nếu yêu cầu chính của hệ thống là độ chính xác tối đa, SVM trên scikit-learn vẫn là lựa chọn ưu tiên. Ngược lại, nếu hệ thống cần xử lý dữ liệu lớn theo thời gian thực với tốc độ cao, RF trên SparkML có thể là một lựa chọn tốt hơn, dù độ chính xác thấp hơn. Do đó, nhóm quyết định duy trì cả hai mô hình song song, để tối ưu hóa hiệu suất trong các tình huống sử dụng khác nhau.

Dưới đây là một số kết quả về trực quan dữ liệu sau khi xử lý và lưu vào database. Xem chi tiết demo tại đường dẫn:  [IE212.P11.VB2 - Google Drive.](https://drive.google.com/file/d/1Wj_pjzS-6KpFmZ377S38AJD2Ct5i4hHU/view?usp=drive_link)

A screenshot of a computer

AI-generated content may be incorrect.

**Hình 7.** Giao diện ứng dụng web với mô hình SVM (trên Sklearn).

A screenshot of a computer

AI-generated content may be incorrect.

**Hình 8.** Giao diện ứng dụng web với mô hình Random Forest (trên Spark ML).

A screenshot of a computer

AI-generated content may be incorrect.

**Hình 9.** Thông tin thành viên.

A screenshot of a computer

AI-generated content may be incorrect.

**Hình 10.** Dữ liệu được lưu vào Database Postgres.

# KẾT LUẬN

Trong bài báo cáo này, nhóm đã nghiên cứu và phát triển một ứng dụng phân tích, dự đoán khách hàng rời bỏ dịch vụ ngân hàng theo thời gian thực bằng cách tích hợp Apache Spark và Kafka. Nội dung được trình bày bao gồm: tổng quan các nghiên cứu liên quan về phân tích churn; thiết kế và kiến trúc hệ thống với Kafka đảm nhiệm streaming dữ liệu, Spark xử lý dữ liệu hiệu quả, cùng Streamlit và Plotly để trực quan hóa kết quả một cách sinh động. Trong quá trình triển khai, nhóm đã thử nghiệm nhiều mô hình học máy như Logistic Regression, KNN, SVM, Random Forest, và GCN, đồng thời kiểm tra hiệu quả của Random Forest trên Spark ML (RF\_SparkML) trong môi trường xử lý dữ liệu lớn phân tán. Kết quả thực nghiệm cho thấy SVM trên scikit-learn đạt hiệu suất cao nhất (accuracy 91.69%), được tích hợp để dự đoán churn theo thời gian thực, trong khi RF\_SparkML (accuracy 85.56%) phù hợp hơn cho dữ liệu streaming quy mô lớn nhờ tốc độ xử lý nhanh.

Tuy nhiên, nghiên cứu vẫn tồn tại một số hạn chế cần lưu ý. Thứ nhất, hệ thống sử dụng dữ liệu giả lập từ tập test thay vì dữ liệu thực từ ngân hàng, điều này có thể ảnh hưởng đến tính đại diện và khả năng áp dụng thực tế. Thứ hai, hiệu suất mô hình trong điều kiện streaming thực tế chưa được đánh giá đầy đủ, đặc biệt về độ trễ xử lý và khả năng mở rộng khi lượng dữ liệu tăng đột biến. Những hạn chế này phần nào làm giảm tính thuyết phục của hệ thống khi triển khai ngoài môi trường học thuật.

Về hướng phát triển tương lai, hệ thống có thể được cải tiến bằng cách tích hợp dữ liệu giao dịch ngân hàng thực tế để tăng độ chính xác và tính thực tiễn. Ngoài ra, việc tối ưu hóa mô hình dự đoán, bổ sung các tính năng phân tích nâng cao (như phát hiện xu hướng churn theo nhóm khách hàng hoặc đề xuất chính sách giữ chân tự động) sẽ nâng cao giá trị ứng dụng. Với những điều chỉnh này, hệ thống hứa hẹn trở thành một công cụ hỗ trợ hiệu quả cho ngành tài chính – ngân hàng trong việc giảm tỷ lệ rời bỏ và tối ưu hóa chiến lược kinh doanh.

# TÀI LIỆU THAM KHẢO

[1] W. Verbeke, K. Dejaeger, D. Martens, J. Hur, and B. Baesens, “New insights into churn prediction in the telecommunication sector: A profit driven data mining approach,” *Eur. J. Oper. Res.*, vol. 218, no. 1, pp. 211–229, Apr. 2012, doi: 10.1016/j.ejor.2011.09.031.

[2] “Bank Customer Churn Prediction.” Accessed: Feb. 27, 2025. [Online]. Available: https://www.kaggle.com/datasets/shubhammeshram579/bank-customer-churn-prediction

[3] W. Verbeke, D. Martens, C. Mues, and B. Baesens, “Building comprehensible customer churn prediction models with advanced rule induction techniques,” *Expert Syst. Appl.*, vol. 38, no. 3, pp. 2354–2364, Mar. 2011, doi: 10.1016/j.eswa.2010.08.023.

[4] S. A. Neslin, S. Gupta, W. Kamakura, J. Lu, and C. H. Mason, “Defection Detection: Measuring and Understanding the Predictive Accuracy of Customer Churn Models,” *J. Mark. Res.*, vol. 43, no. 2, pp. 204–211, May 2006, doi: 10.1509/jmkr.43.2.204.

[5] Tan, Pang-Ning, M. Steinbach, M. Adeyeye Oshin, V. Kumar, and Vipin, *Introduction to Data Mining*. 2005.

[6] U. Fayyad, “From Data Mining to Knowledge Discovery in Databases”.

[7] A. Lemmens and C. Croux, “Bagging and Boosting Classification Trees to Predict Churn,” *J. Mark. Res.*, vol. 43, no. 2, pp. 276–286, May 2006, doi: 10.1509/jmkr.43.2.276.

[8] J. Burez and D. Van Den Poel, “Handling class imbalance in customer churn prediction,” *Expert Syst. Appl.*, vol. 36, no. 3, pp. 4626–4636, Apr. 2009, doi: 10.1016/j.eswa.2008.05.027.

[9] W. Verbeke, K. Dejaeger, D. Martens, J. Hur, and B. Baesens, “New insights into churn prediction in the telecommunication sector: A profit driven data mining approach,” *Eur. J. Oper. Res.*, vol. 218, no. 1, pp. 211–229, Apr. 2012, doi: 10.1016/j.ejor.2011.09.031.

[10] Y. Huang and T. Kechadi, “An effective hybrid learning system for telecommunication churn prediction,” *Expert Syst. Appl.*, vol. 40, no. 14, pp. 5635–5647, Oct. 2013, doi: 10.1016/j.eswa.2013.04.020.