MỞ ĐẦU

1. Lý do chọn đề tài:

Trong thời đại công nghệ 4.0 ngày nay, việc nắm bắt thông tin được coi là cơ sở của mọi hoạt động sản xuất, kinh doanh. Các cá nhân hoặc tổ chức nào thu thập, hiểu được công nghệ và hoạt động dựa trên các công nghệ 4.0 sẽ đạt được những thành công trong mọi hoạt động sản xuất kinh doanh. Công nghệ thông tin (CNTT) hiện nay cho phép ta khai thác được tri thức hữu dụng từ Cơ sở dữ liệu (CSDL) gọi là kỹ thuật Khai phá dữ liệu (DM).

Khai phá dữ liệu là quá trình tính toán để tìm ra các mẫu trong các bộ dữ liệu lớn liên quan đến các phương pháp tại giao điểm của máy học, thống kê và các hệ thống cơ sở dữ liệu. Đây là một lĩnh vực liên ngành của khoa học máy tính. Mục tiêu tổng thể của quá trình khai phá dữ liệu là trích xuất thông tin từ một bộ dữ liệu và chuyển nó thành một cấu trúc dễ hiểu để sử dụng tiếp. Ngoài bước phân tích thô, nó còn liên quan tới cơ sở dữ liệu và các khía cạnh quản lý dữ liệu, xử lý dữ liệu trước, suy xét mô hình và suy luận thống kê, các thước đo thú vị, các cân nhắc phức tạp, xuất kết quả về các cấu trúc được phát hiện, trực quan hóa và cập nhật trực tuyến. Khai phá dữ liệu là bước phân tích của quá trình "khám phá kiến thức trong cơ sở dữ liệu" hoặc KDD (Knowledge Discovery in Databases) [1]. Khai phá dữ liệu (KPDL) trong cơ sở dữ liệu (CSDL) đang là một xu hướng quan trọng của nền công nghệ thông tin (CNTT) hiện nay.

KPDL có khả năng ứng dụng vào rất nhiều lớp bài toán thực tế khác nhau, là qui trình mà các ngân hàng sử dụng để biến những dữ liệu thô thành thông tin hữu ích. Bằng cách dùng phần mềm để tìm mẫu hình trong các tập dữ liệu, các ngân hàng có thể hiểu hơn về khách hàng của họ và phát triển được những chiến lược marketing hiệu quả, giúp tăng doanh thu và giảm chi phí. Việc khai phá dữ liệu phụ thuộc vào việc thu thập dữ liệu một cách hiệu quả, lưu trữ kho dữ liệu và xử lí máy tính.

Các đợt tiếp thị chào hàng tạo thành một chiến lược điển hình để nâng cao hoạt động kinh doanh. Các ngân hàng sử dụng tiếp thị trực tiếp khi nhắm đến các mục tiêu phân khúc khách hàng bằng cách liên hệ với họ để đáp ứng một mục tiêu cụ thể. Tập trung hóa các tương tác khách hàng từ xa giúp giảm bớt việc quản lý hoạt động của các đợt. Việc liên lạc như vậy cho phép giao tiếp với khách hàng qua nhiều kênh khác nhau: điện thoại cố định, điện thoại di động đang được sử dụng rộng rãi nhất. Tiếp thị được thực hiện thông qua một trung tâm liên lạc được gọi là tiếp thị qua điện thoại. Địa chỉ liên hệ có thể thực hiện trong và ngoài nước, tùy thuộc vào việc bên nào đã thực hiện liên hệ (khách hàng hoặc trung tâm liên hệ), với mỗi trường hợp đặt ra những thách thức khác nhau. Công nghệ cho phép thực hiện thương mại bằng cách tập trung vào việc tối đa hóa giá trị lâu dài của khách hàng thông qua việc đánh giá thông tin sẵn có và các chỉ số khách hàng, do đó cho phép các ngân hàng xây dựng các mối quan hệ lâu dài và chặt chẽ hơn phù hợp với yêu cầu kinh doanh [1]. Ngoài ra, cần nhấn mạnh rằng nhiệm vụ lựa chọn nhóm khách hàng tốt nhất, tức là có nhiều khả năng đăng ký một sản phẩm hơn.

Trong luận văn này, em mạnh dạn đề xuất phương pháp khai phá dữ liệu (DM) để dự đoán sự thành công của các cuộc gọi qua điện thoại trong hoạt động tiếp thị các sản phẩm của ngân hàng; Để góp phần nâng cao hiệu quả của việc ứng dụng marketing (tiếp thị) trong hoạt động kinh doanh ở các ngân hàng, em đã chọn đề tài "**Ứng dụng khai phá dữ liệu xây dựng hệ thống phân tích hoạt động tiếp thị ngân hàng**" cho đề tài tốt nghiệp của mình. Mục tiêu của đề tài là xuất phát từ những đặc điểm chung về hoạt động marketing trong ngành ngân hàng và thực trạng ứng dụng của nó ở các ngân hàng để tìm ra những giải pháp giúp cho các nhà quản trị ngân hàng nâng cao hiệu quả việc ứng dụng marketing trong lĩnh vực kinh doanh của mình.

Kỹ thuật khai phá dữ liệu mà em áp dụng là mô hình hồi quy logistic (LR), cây quyết định (DT). Việc khai phá dữ liệu như vậy đã tạo nên mô hình thu được là đáng tin cậy và có giá trị đối với các nhà quản lý đợt tiếp thị qua điện thoại của ngân hàng [1].

Em xin chân thành cảm ơn Phó giáo sư – Tiến sĩ Võ Thị Lưu Phương đã tận tình giúp đỡ, hướng dẫn để em hoàn thành được đề tài này.

2. Tổng quan về vấn đề nghiên cứu:

Trong đề tài hướng đến đề xuất một Hệ thống hỗ trợ quyết định (DSS) sử dụng công nghệ thông tin để hỗ trợ việc ra quyết định của nhà quản lý. DSS cá nhân và thông minh có thể tự động dự đoán kết quả của một cuộc gọi điện thoại để tiếp thị các sản phẩm của ngân hàng bằng cách sử dụng cách tiếp cận tới khai phá dữ liệu (DM). DSS như vậy có giá trị để hỗ trợ các nhà quản lý trong việc ưu tiên và lựa chọn những khách hàng tiếp theo sẽ được liên hệ trong đợt tiếp thị của ngân hàng [1]. Ví dụ: bằng cách sử dụng mô hình hồi quy logistic để phân tích xác suất thành công của việc tiếp thị qua điện thoại để người quản lý quyết định là cần liên hệ với bao nhiêu khách hàng và những khách hàng nào. Do đó, thời gian và chi phí trong số các đợt như vậy sẽ bị giảm. Ngoài ra, bằng cách thực hiện ít hơn và các cuộc gọi điện thoại hiệu quả hơn .

Để thực hiện được mục đích ý tưởng đề ra cho việc đóng góp chính của công việc này là: Tập trung vào tính năng kỹ thuật, là một khía cạnh quan trọng trong DM và đề xuất các chỉ số kinh tế và xã hội chung ngoài các thuộc tính sản phẩm và khách hàng được sử dụng phổ biến hơn của ngân hàng. Mô hình DM bằng cách sử dụng đánh giá và phân loại số liệu. Đề tài cũng chỉ ra các mô hình tốt nhất có thể mang lại lợi ích cho tiếp thị qua điện thoại của ngân hàng trong việc kinh doanh.

3. Mục đích nghiên cứu:

Mục đích chính: Nâng cao hiệu quả tiếp thị qua điện thoại của ngân hàng trong việc kinh doanh bằng việc sử dụng mô hình hồi quy logistic (LR) và cây quyết định (DT) trong khai phá dữ liệu.

4. Đối tượng và phạm vi nghiên cứu:

Đối tượng nghiên cứu: Các mô hình khai phá dữ liệu (DM); Hệ thống hỗ trợ quyết định (DSS) sử dụng công nghệ thông tin; Tập dữ liệu tiếp thị ngân hàng (khách hàng và ngân hàng); Công cụ hỗ trợ lập trình Python và Anacoda3 và một số công cụ hỗ trợ khai phá dữ liệu.

Phạm vi nghiên cứu: Nghiên cứu về khai phá dữ liệu dựa trên mô hình hồi quy logistic (LR) và cây quyết định (DT) trong khai phá dữ liệu; Bài toán tiếp thị ngân hàng để dự đoán dữ liệu khách hàng có đăng ký một sản phẩm hơn hay không?

5. Giả thuyết nghiên cứu:

Xây dựng chương trình dự báo kết quả thông qua các cuộc gọi điện thoại tiếp thị qua điện thoại để tiếp thị các khoản tiền gửi dài hạn hiệu quả nhất.

Khi các nhân viên thực hiện các cuộc gọi điện thoại đến danh sách khách hàng để tiếp thị sản phẩm hoặc nếu trong khi khách hàng gọi đến trung tâm liên lạc của ngân hàng vì bất kỳ lý do nào khác, khách hàng được yêu cầu đăng ký sản phẩm. Do đó, kết quả là một nhị phân liên hệ không thành công hoặc thành công.

6. Câu hỏi nghiên cứu:

Trong ngành ngân hàng, tối ưu hóa nhằm mục tiêu cho tiếp thị qua điện thoại là một vấn đề then chốt, dưới áp lực ngày càng tăng nhằm tăng lợi nhuận và giảm chi phí thì việc lựa chọn trong 2 mô hình thì mô hình nào cho ra kết quả tối ưu nhất?

Đặt ra bao nhiêu phần trăm tiếp thị ngân hàng thành công, bao nhiêu phần trăm không thành công?

7. Phương pháp nghiên cứu: Để hoàn thành hệ thống phân tích hoạt động trong Tiếp thị ngân hàng, em sử dụng ngôn ngữ lập trình Python *và* Anacoda3 để thực hiện được mục tiêu này cho đề tài của mình.

Chương 1: CƠ SỞ LÝ LUẬN

1.1. Tổng quan về phát hiện tri thức và khai phá dữ liệu

Khai phá dữ liệu là một quá trình trích xuất và khám phá các mẫu trong tập dữ liệu lớn liên quan đến các phương pháp tại điểm giao nhau của máy học, thống kê và hệ thống cơ sở dữ liệu.

Khai phá dữ liệu là một lĩnh vực con liên ngành của khoa học máy tính và thống kê với mục tiêu tổng thể là trích xuất thông tin (bằng các phương pháp thông minh) từ một tập dữ liệu và chuyển đổi thông tin thành một cấu trúc dễ hiểu để sử dụng thêm [1].

Khai phá dữ liệu là bước phân tích của quá trình "Khám phá kiến thức trong cơ sở dữ liệu", hay còn gọi là KDD. Bên cạnh bước phân tích thô, nó cũng bao gồm các khía cạnh quản lý cơ sở dữ liệu và dữ liệu, xử lý trước dữ liệu, xem xét mô hình và suy luận, chỉ số đo mức độ thú vị, cân nhắc độ phức tạp, xử lý sau các cấu trúc đã phát hiện, trực quan hóa và cập nhật trực tuyến [1].

Khai phá dữ liệu từ cơ sở dữ liệu bao gồm nhiều công đoạn như: xác định vấn đề, tập hợp và chọn lọc dữ liệu, khai thác dữ liệu, đánh giá kết quả, giải thích dữ liệu, áp dụng tri thức vào thực tế.

1.2. Quá trình phát hiện tri thức và khai phá dữ liệu

Gom dữ liệu (Gathering): Tập hợp dữ liệu là bước đầu tiên trong quá trình KPDL. Đây là bước được khai thác trong một CSDL, một kho dữ liệu và thậm chí các dữ liệu từ các nguồn ứng dụng Web.

Trích lọc dữ liệu (Selection): Ở giai đoạn này dữ liệu được lựa chọn hoặc phân chia theo một số tiêu chuẩn nào đó, ví dụ chọn tất cả những người có tuổi đời từ hai mươi lăm đến ba mươi lăm và có trình độ đại học.

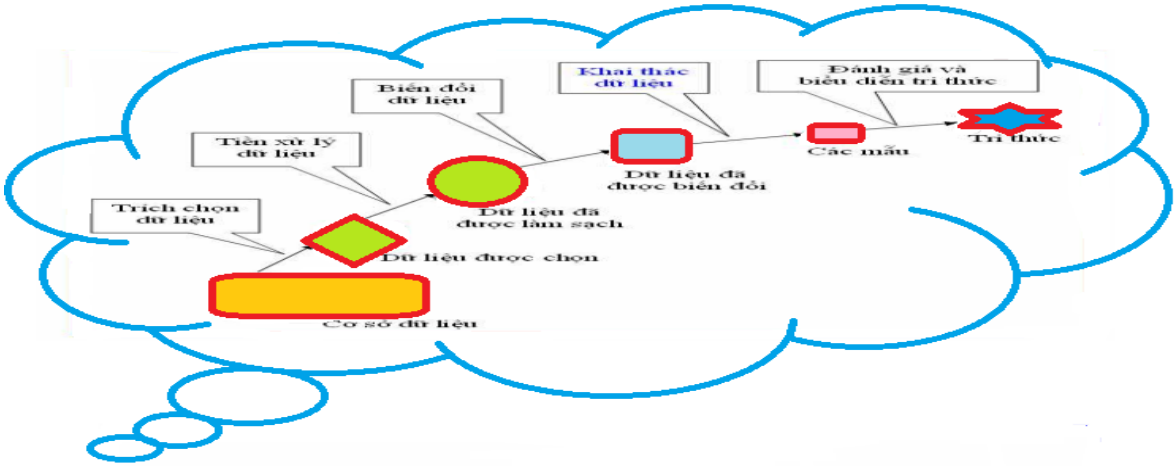
Làm sạch, tiền xử lý và chuẩn bị trước dữ liệu (Cleansing, Pre-processing and Preparation): Giai đoạn thứ ba này là giai đoạn hay bị sao lãng, nhưng thực tế nó là một bước rất quan trọng trong quá trình KPDL. Một số lỗi thường mắc phải trong khi gom dữ liệu là tính không đủ chặt chẽ, logic. Vì vậy, dữ liệu thường chứa

các giá trị vô nghĩa và không có khả năng kết nối dữ liệu. Ví dụ: tuổi = sáu trăm bảy mươi ba. Giai đoạn này sẽ tiến hành xử lý những dạng dữ liệu không chặt chẽ nói trên. Những dữ liệu dạng này được xem như thông tin dư thừa, không có giá trị. Bởi vậy, đây là một quá trình rất quan trọng vì dữ liệu này nếu không được "làm sạch - tiền xử lý - chuẩn bị trước" thì sẽ gây nên những kết quả sai lệch nghiêm trọng.

Chuyển đổi dữ liệu (Transformation): Tiếp theo là giai đoạn chuyển đổi dữ liệu, dữ liệu đưa ra có thể sử dụng và điều khiển được bởi việc tổ chức lại nó. Dữ liệu đã được chuyển đổi phù hợp với mục đích khai thác.

Phát hiện và trích mẫu dữ liệu (Pattern Extraction and Discovery): Đây là bước mang tính tư duy trong KPDL. Ở giai đoạn này nhiều thuật toán khác nhau đã được sử dụng để trích ra các mẫu từ dữ liệu. Thuật toán thường dùng là nguyên tắc phân loại, nguyên tắc kết hợp hoặc các mô hình dữ liệu tuần tự, v.v.

Đánh giá kết quả mẫu (Evaluation of Result): Đây là giai đoạn cuối trong quá trình KPDL. Ở giai đoạn này, các mẫu dữ liệu được chiết xuất ra bởi phần mềm KPDL. Không phải bất cứ mẫu dữ liệu nào cũng đều hữu ích, đôi khi nó còn bị sai lệch. Vì vậy, cần phải ưu tiên những tiêu chuẩn đánh giá để chiết xuất ra các tri thức cần chiết xuất ra.



Hình 1.1: Các giai đoạn trong quá trình khai phá dữ liệu

Trên đây là sáu giai đoạn trong quá trình KPDL, trong đó giai đoạn 5 là giai đoạn được quan tâm nhiều nhất hay còn gọi đó là KPDL.

1.3. Các phương pháp khai phá dữ liệu

- ***Phương pháp nghiên cứu lý luận***: Thu thập, đọc hiểu, phân tích thông tin, dữ liệu từ các tài liệu, giáo trình, sách liên quan đến khai phá dữ liệu.

- ***Phương pháp nghiên cứu thực tiễn***: Tiến hành nghiên cứu các kỹ thuật cho phép phân lớp trong khai phá dữ liệu, ứng dụng các kỹ thuật đó để xây dựng mô hình dự đoán kết quả tiếp thị ngân hàng dựa vào các thông tin đầu vào. Đề tài tiến hành so sánh kết quả của các mô hình để lựa chọn mô hình cho kết quả vượt trội nhất. Từ đó, xây dựng chương trình dự báo kết quả thông qua các cuộc gọi điện thoại tiếp thị qua điện thoại để tiếp thị các khoản tiền gửi dài hạn hiệu quả nhất. Việc xây dựng mô hình được tiến hành theo các bước:

- Làm sạch và tích hợp dữ liệu.
- Lựa chọn dữ liệu và chuyển đổi dữ liệu
- Khai thác dữ liệu
- Sự trực quan hóa
- Biểu diễn và đánh giá mô hình

Dữ liệu là mỗi bản ghi bao gồm mục tiêu đầu ra, kết quả liên hệ ({“thất bại”, “thành công”}) và các tính năng đầu vào ứng viên. Chúng bao gồm các thuộc tính tiếp thị qua điện thoại (ví dụ: chỉ đường cuộc gọi), chi tiết sản phẩm (ví dụ: lãi suất được cung cấp) và thông tin khách hàng (ví dụ: tuổi). Các bản ghi này đã được làm giàu với các tính năng ảnh hưởng xã hội và kinh tế (ví dụ: tỷ lệ thay đổi thất nghiệp).

- ***Phương pháp nghiên cứu tài liệu:*** Tìm hiểu ngôn ngữ lập trình, hệ quản trị Cơ sở dữ liệu (CSDL), Xây dựng ứng dụng.

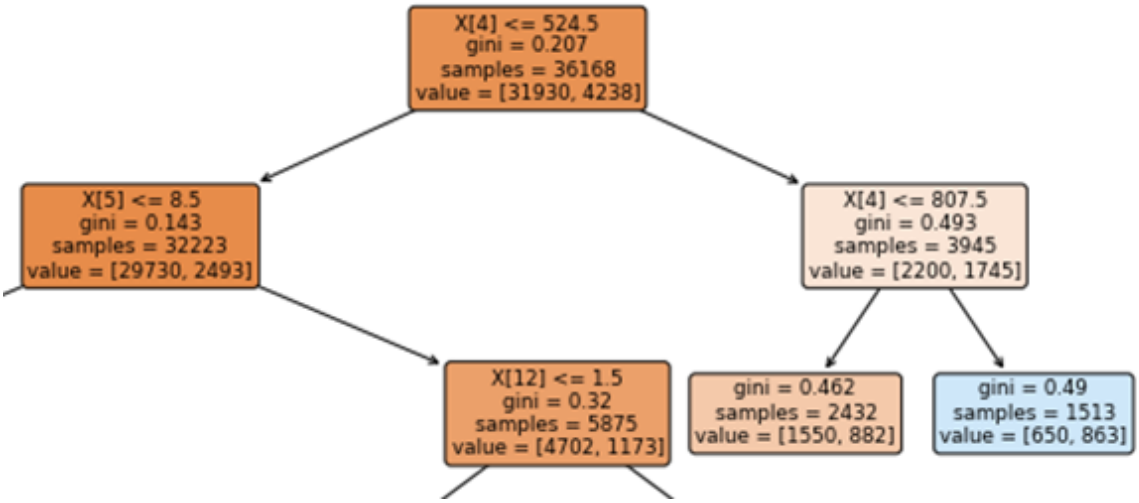
1.4. Mô hình khai phá dữ liệu

- **Mô hình Hồi quy Logistic (LR)**: Mục tiêu của hồi qui Logistic là nghiên cứu mối tương quan giữa một (hay nhiều) yếu tố nguy cơ (*risk factor*) và đối tượng phân tích (*outcome*). Chẳng hạn như đối với nghiên cứu mối tương quan giữa thói quen hút thuốc lá và nguy cơ mắc ung thư phổi thì yếu tố nguy cơ ở đây là thói quen hút thuốc lá và đối tượng phân tích ở đây là nguy cơ mắc ung thư phổi. Trong hồi

qui logistic thì các đối tượng nghiên cứu thường được thể hiện qua các biến số nhị phân (binary) như *xảy ra/ không xảy ra; chết/sống; có/không,…* còn các yếu tố nguy cơ có thể được thể hiện qua các biến số liên tục (tuổi, huyết áp,…) hoặc các biến nhị phân (giới tính) hay các biến thứ bậc (thu nhập: Cao, trung bình, thấp). Vấn đề đặt ra cho nghiên cứu dạng này là làm sao để ước tính độ tương quan của các yếu tố nguy cơ và đối tượng phân tích. Các phương pháp phân tích như hồi qui tuyến tích không áp dụng được vì biến phụ thuộc không phải là biến liên tục mà là biến nhị phân [3].

- Mô hình Cây quyết định (DT): Cây quyết định là một trong những mô hình có khả năng diễn giải cao và có thể thực hiện cả nhiệm vụ phân loại và hồi quy. Như vậy cho thấy Cây Quyết định là mô hình cấu trúc giống như cây lộn ngược. Tại thời điểm này, bạn có thể có một câu hỏi như chúng ta đã có các mô hình họ máy học cổ điển như hồi quy tuyến tính và hồi quy logistic để thực hiện các nhiệm vụ hồi quy và phân loại trong trường hợp như vậy thì sự cần thiết của một mô hình khác như Cây quyết định là gì. Câu trả lời cho câu hỏi này là để thực hiện các mô hình tuyến tính cổ điển, chúng ta cần đảm bảo rằng dữ liệu được sử dụng để đào tạo mô hình không có tất cả các bất thường như giá trị bị thiếu, các giá trị ngoại lệ cần được xử lý, đa cộng tuyến cần được giải quyết. Toàn bộ quá trình tiền xử lý dữ liệu cần được thực hiện trước đó. Trong khi trong Cây quyết định, chúng ta không cần phải thực hiện bất kỳ loại xử lý trước dữ liệu nào trước đó. Cây Quyết định đủ mạnh để xử lý tất cả các loại vấn đề như vậy để đi đến quyết định. Ngoài ra, Cây quyết định có khả năng xử lý dữ liệu phi tuyến mà các mô hình tuyến tính cổ điển không xử lý được. Do đó Cây quyết định đủ đa dạng để thực hiện cả nhiệm vụ hồi quy và phân loại. Toàn bộ những ưu và nhược điểm liên quan đến Cây Quyết định có thể được thảo luận chi tiết trong phần sau của bài viết này. Trước đó, hãy bắt đầu tìm hiểu Cây quyết định. Cây quyết định xây dựng cây bằng cách đặt một loạt câu hỏi vào dữ liệu để đi đến quyết định. Do đó người ta nói rằng Cây Quyết định bắt chước quá trình quyết định của con người. Trong quá trình xây dựng cây, nó chia toàn bộ

dữ liệu thành các tập dữ liệu con cho đến khi đưa ra quyết định. Hãy cùng tìm hiểu một vài thuật ngữ liên quan đến cây Quyết định để hiểu rõ hơn về Cây quyết định.



Hình 1.2: Minh họa cây quyết định

1.5. Kết luận

KPDL là một lĩnh vực được quan tâm và ứng dụng rộng rãi. Một số ứng dụng điển hình trong KPDL có thể liệt kê: phân tích dữ liệu và hỗ trợ ra quyết định; điều trị y học; phát hiện văn bản; tin sinh học; tài chính và TTCK; bảo hiểm...

Quá trình nghiên cứu tổng quan về khai phá dữ liệu giúp chúng ta hiểu được các bước trong qui trình khai phá dữ liệu, phương pháp, dạng dữ liệu có thể khai phá và những vấn đề cần giải quyết trong khai phá dữ liệu.

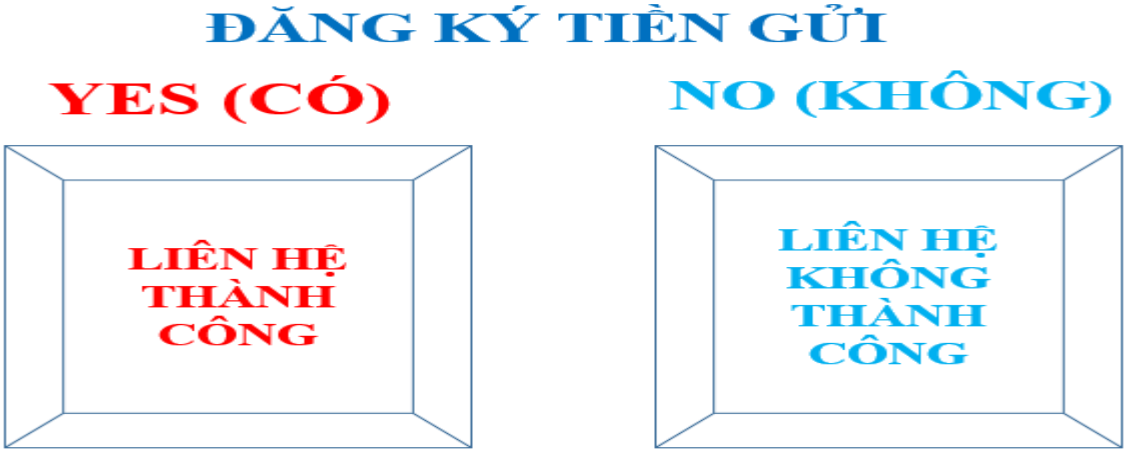
Chương 2:
PHƯƠNG PHÁP TIẾP CẬN THEO HƯỚNG DỮ LIỆU ĐỂ DỰ ĐOÁN SỰ THÀNH CÔNG CỦA TIẾP THỊ QUA ĐIỆN THOẠI NGÂN HÀNG

2.1. Tổng quan cơ sở dữ liệu Tiếp thị ngân hàng.

Nghiên cứu này tập trung vào mục tiêu thông qua các cuộc gọi điện thoại tiếp thị qua điện thoại để tiếp thị các sản phẩm. Trong một đợt, các nhân viên thực hiện các cuộc gọi điện thoại đến danh sách khách hàng để tiếp thị sản phẩm hoặc nếu trong khi khách hàng gọi đến trung tâm liên lạc của ngân hàng vì bất kỳ lý do nào khác, khách hàng được yêu cầu đăng ký sản phẩm. Do đó, kết quả là một nhị phân liên hệ không thành công hoặc thành công [1].

2.2. Phân tích yêu cầu chức năng tập dữ liệu.

- Dữ liệu tiếp thị qua điện thoại của ngân hàng (Bank telemarketing data): Nghiên cứu này tập trung vào việc thông qua các cuộc gọi điện thoại tiếp thị qua điện thoại để tiếp thị các khoản tiền gửi dài hạn. Trong các cuộc gọi điện thoại đến danh sách khách hàng để tiếp thị tiền ký gửi, nếu trong khi khách hàng gọi đến số đường dây nóng của ngân hàng đều được miễn phí và vì bất kỳ lý do nào khách được yêu cầu đăng ký tiền gửi (inbound) đến ngân hàng, kết quả là một nhị phân liên hệ không thành công hoặc thành công.

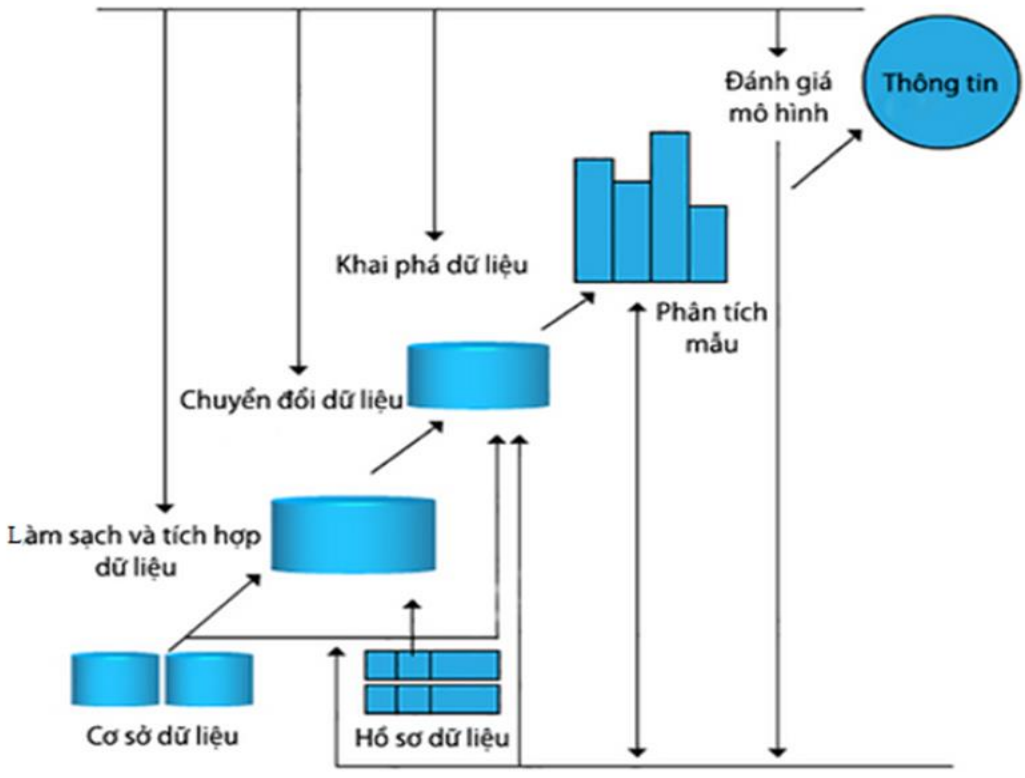


Hình 2.1: Khách hàng có đăng ký tiền gửi hay không đăng ký tiền gửi

- Nghiên cứu này xem xét dữ liệu thực được thu thập từ một cửa hàng bán lẻ ở ngân hàng Bồ Đào Nha từ tháng 5 năm 2008 đến tháng 6 năm 2013, trong tổng số 45.211 liên hệ điện thoại. Tập dữ liệu không cân bằng, vì chỉ có 3.560 (12,70%) bản ghi có liên quan với việc tiếp thị thành công [2].

- Mỗi bản ghi bao gồm mục tiêu đầu ra, kết quả liên hệ ({“thất bại ”,“ thành công ”}) và các tính năng đầu vào ứng viên. Chúng bao gồm các thuộc tính tiếp thị qua điện thoại (ví dụ: hướng cuộc gọi), chi tiết sản phẩm (ví dụ: lãi suất được cung cấp) và thông tin khách hàng (ví dụ: tuổi). Các bản ghi này đã được làm giàu với các đặc điểm ảnh hưởng xã hội và kinh tế (ví dụ: tỷ lệ thất nghiệp tỷ giá hối đoái), bằng cách thu thập dữ liệu bên ngoài từ ngân hàng trung ương của Bồ Đào Nha.

2.3. Thiết kế hệ thống



Hình 2.2 – Sơ đồ hệ thống cơ sở dữ liệu

Cơ sở dữ liệu

Dữ liệu có liên quan đến các đợt tiếp thị trực tiếp của một tổ chức ngân hàng Bồ Đào Nha. Các đợt tiếp thị dựa trên các cuộc gọi điện thoại. Thông thường, cần

có nhiều liên hệ với cùng một khách hàng, để truy cập xem sản phẩm (tiền gửi có kỳ hạn ngân hàng) sẽ được (có) hay không (không) được đăng ký.

- Trong trường hợp này, em xin sẽ sử dụng máy học để hiểu mẫu và dự đoán phân loại hoặc nhãn, em sử dụng một số mô hình dự đoán để dự đoán bằng cách sử dụng dữ liệu đào tạo và thử nghiệm. Mô hình dự đoán mà em sử dụng là mô hình Logistic và Cây Quyết định.

- Bộ dữ liệu đến từ kho lưu trữ máy học UCI và nó có liên quan đến các đợt tiếp thị trực tiếp (gọi điện thoại) của một tổ chức ngân hàng Bồ Đào Nha. Các đợt tiếp thị dựa trên các cuộc gọi điện thoại. Bộ dữ liệu này chứa các trường được phân tách bằng dấu phẩy.

- Các đợt tiếp thị dựa trên các cuộc gọi điện thoại. Thông thường, yêu cầu nhiều hơn một địa chỉ liên hệ với cùng một khách hàng, để truy cập xem sản phẩm (tiền gửi có kỳ hạn ngân hàng) sẽ được (hoặc không) đăng ký [2].

- Trong dự án này, em cần xây dựng một mô hình để quyết định xem liệu một đợt có thành công trong việc thu hút khách hàng đăng ký tiền gửi có kỳ hạn hay không.

2.4. Xây dựng cơ sở dữ liệu tiếp thị ngân hàng.

- Tiếp thị qua điện thoại qua ngân hàng là một phương pháp tiếp thị trực tiếp mà một người (có thể là bán hàng) khách hàng tiềm năng mua sản phẩm hoặc dịch vụ, qua điện thoại hoặc qua cuộc hẹn gặp mặt trực tiếp hoặc hội nghị qua web. Tiếp thị qua điện thoại cũng có thể bao gồm các cuộc bán hàng đã tính lại được lập trình để phát qua điện thoại bằng cách quay số tự động.

- Ngân hàng là một trong những tổ chức sử dụng phương thức tiếp thị qua điện thoại để bán các sản phẩm, dịch vụ ngân hàng. Tiếp thị qua điện thoại là một phương pháp phổ biến được ngân hàng sử dụng để bán hàng, vì các sản phẩm và dịch vụ của ngân hàng đôi khi quá phức tạp đối với một số người dùng không thể hiểu được. Người dùng hoặc người dùng mục tiêu sẽ dễ dàng hiểu sản phẩm hoặc dịch vụ hơn nếu nó giải thích trực tiếp. Một lợi thế của tiếp thị qua điện thoại theo

từng người, người dùng mục tiêu có thể trực tiếp đặt câu hỏi, nếu họ không hiểu điều gì đó.

- Ngày nay, Tiếp thị qua điện thoại có liên quan tiêu cực đến các trò gian lận và lừa đảo khác nhau, chẳng hạn như các kế hoạch kim tự tháp và với các sản phẩm và dịch vụ được định giá quá cao. Các công ty tiếp thị qua điện thoại gian lận thường được gọi là "thổi phồng tiếp thị qua điện thoại" hoặc đơn giản là "thổi phồng". Tiếp thị qua điện thoại thường bị chỉ trích là một hoạt động kinh doanh phi đạo đức do nhận thức về các kỹ thuật bán hàng áp lực cao trong các cuộc gọi không được yêu cầu. Tiếp thị qua điện thoại các công ty điện thoại có thể tham gia vào việc đánh sập điện thoại, hành vi chuyển đổi dịch vụ điện thoại của khách hàng mà họ không biết hoặc không được họ cho phép.

- Ngân hàng với tư cách là tổ chức tài chính thực sự quan tâm đến danh tiếng tốt và thương hiệu tốt, và một trong những điều tồi tệ là tiếp thị qua điện thoại có thể tự làm ảnh hưởng đến danh tiếng của nó. Vì vậy, chúng tôi cần tìm hiểu mục tiêu của chúng tôi sẽ không mua sản phẩm hoặc dịch vụ nào nếu ngân hàng cung cấp sản phẩm hoặc dịch vụ bằng cách sử dụng tiếp thị qua điện thoại. Nó có thể giúp bảo vệ danh tiếng ngân hàng bằng cách không làm phiền mục tiêu mà chúng ta đã biết sẽ không mua sản phẩm.

2.4.1. Cơ sở bộ dữ liệu

bộ dữ liệu: bank-full.csv [6]. Số phiên bản: 45211.

Dữ liệu khách hàng ngân hàng

- Age - tuổi (số)
- Job - công việc: loại công việc (phân loại: 'quản trị viên.', 'cổ áo xanh', 'doanh nhân', 'người giúp việc', 'quản lý', 'nghỉ hưu', 'tự kinh doanh', 'dịch vụ', 'sinh viên', ' kỹ thuật viên ',' thất nghiệp ',' không xác định ')
- Marital - tình trạng hôn nhân (phân loại: 'đã ly hôn', 'đã kết hôn', 'độc thân', 'không xác định'; lưu ý: 'đã ly hôn' có nghĩa là đã ly hôn hoặc góa bụa)
- Education - giáo dục: (phân loại "không xác định", "trung học", "tiểu học", "đại học")

- Default - có tín dụng trong tình trạng vỡ nợ? (nhị phân: "yes", "no")
- Balance - số dư: số dư trung bình hàng năm, tính bằng **euro** (số)
- Housing - nhà ở: có cho vay mua nhà không? (nhị phân: "yes", "no")
- Loan - vay: có vay cá nhân không? (nhị phân: "yes", "no")

Dữ liệu liên quan đến người liên hệ cuối cùng của đợt hiện tại

- Contact - liên hệ: loại liên lạc liên hệ (phân loại: "không xác định", "điện thoại", "di động")
- Day - ngày: ngày liên hệ cuối cùng của tháng (số)
- Month - tháng: tháng liên hệ cuối cùng trong năm (phân loại: "jan", "feb", "mar", ..., "nov", "dec")
- Duration - thời lượng: thời lượng liên hệ cuối cùng, tính bằng giây (số)

Các thuộc tính khác:

- Campaign - đợt: số lượng liên hệ được thực hiện trong đợt này và cho khách hàng này (số, bao gồm liên hệ cuối cùng)
- Pdays - số ngày trôi qua sau khi khách hàng được liên hệ lần cuối từ một đợt trước đó (số, -1 có nghĩa là khách hàng chưa được liên hệ trước đó)
- Previous - trước: số lượng địa chỉ liên hệ được thực hiện trước đợt này và cho khách hàng này (số)
- Poutcome - kết quả của đợt tiếp thị trước đó (phân loại: 'thất bại', 'không tồn tại', 'thành công')

Biến đầu ra (mục tiêu mong muốn): y: khách hàng đã đăng ký tiền gửi có kỳ hạn chưa? (nhị phân: "yes", "no") [4]

Bảng 2.1: Dữ liệu tiếp thị ngân hàng (Tập dữ liệu bank-full.csv)

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	no
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
45206	51	technician	married	tertiary	no	825	no	no	cellular	17	nov	977	3	-1	0	unknown	yes
45207	71	retired	divorced	primary	no	1729	no	no	cellular	17	nov	456	2	-1	0	unknown	yes
45208	72	retired	married	secondary	no	5715	no	no	cellular	17	nov	1127	5	184	3	success	yes
45209	57	blue-collar	married	secondary	no	668	no	no	telephone	17	nov	508	4	-1	0	unknown	no
45210	37	entrepreneur	married	secondary	no	2971	no	no	cellular	17	nov	361	2	188	11	other	no

45211 rows × 17 columns

2.4.2. Giá trị dữ liệu xây dựng trong các cột của tập dữ liệu:

Xem giá trị dữ liệu xây dựng trong các cột của tập dữ liệu:

Số lượng của cột công việc

banktelemarket['job'].astype("category").value_counts()

Bảng 2.2: Số lượng cột công việc

```
blue-collar      9732
management       9458
technician       7597
admin.           5171
services         4154
retired          2264
self-employed    1579
entrepreneur     1487
unemployed       1303
housemaid        1240
student           938
unknown           288
Name: job, dtype: int64
```

Công việc blue-collar: 9.732 người

Công việc management :9.458 người

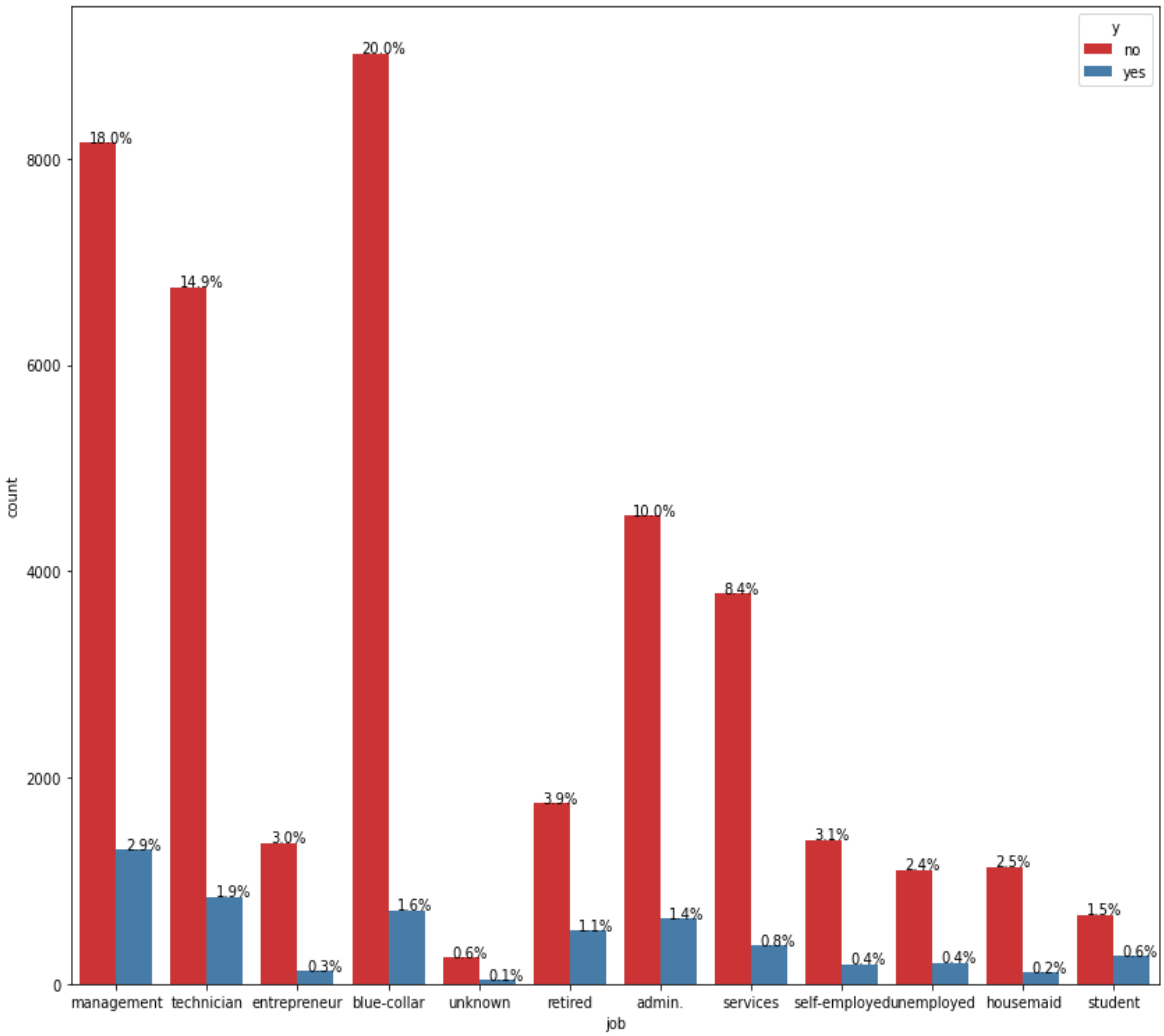
Công việc technician: 7.597 người

Công việc admin.: 5.171 người

Công việc services: 4.154 người

Công việc student: 938 người

Công việc unknown: 288 người



Hình 2.3 – Tỷ lệ khách hàng đăng ký tiền gửi hay không đăng ký theo nghề nghiệp

- Khách hàng làm công việc nhân viên quản lý có tỷ lệ đăng ký tiền gửi có kỳ hạn cao hơn, nhưng cũng cao thứ 2 khi không đăng ký. Điều này đơn giản là vì chúng ta có nhiều khách hàng làm nhân viên quản lý hơn bất kỳ nghề nào khác.
- Khách hàng làm công việc blue-collar có tỷ lệ đăng ký tiền gửi có kỳ hạn cao đứng thứ 3, nhưng cũng cao nhất khi không đăng ký.

  # Số lượng của cột hôn nhân
  banktelemarket['marital'].astype("category").value_counts()

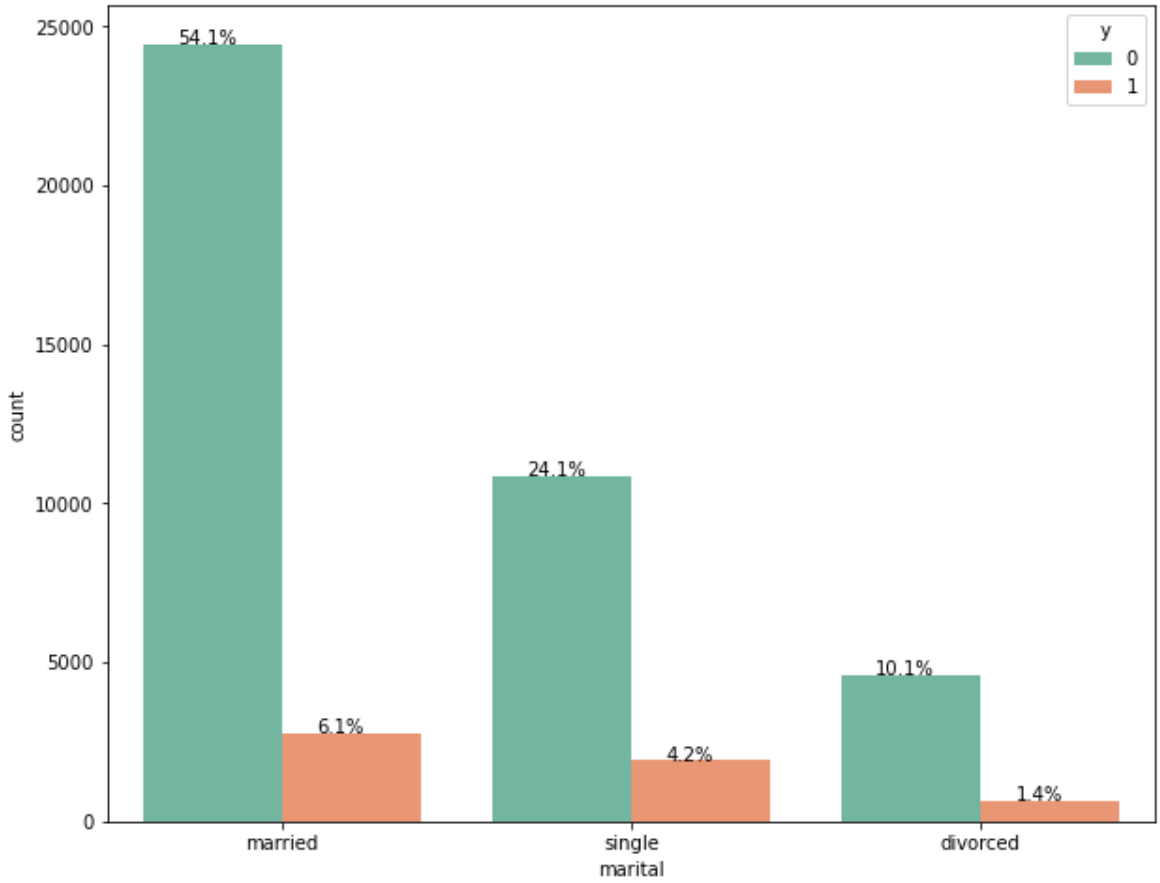
Bảng 2.3: Số lượng cột hôn nhân

```
married     27214
single      12790
divorced     5207
Name: marital, dtype: int64
```

Số người đã kết hôn (married): 27.214 người

Số người còn độc thân (single): 12.790 người

Số người đã đã ly dị (divorced): 5.207 người

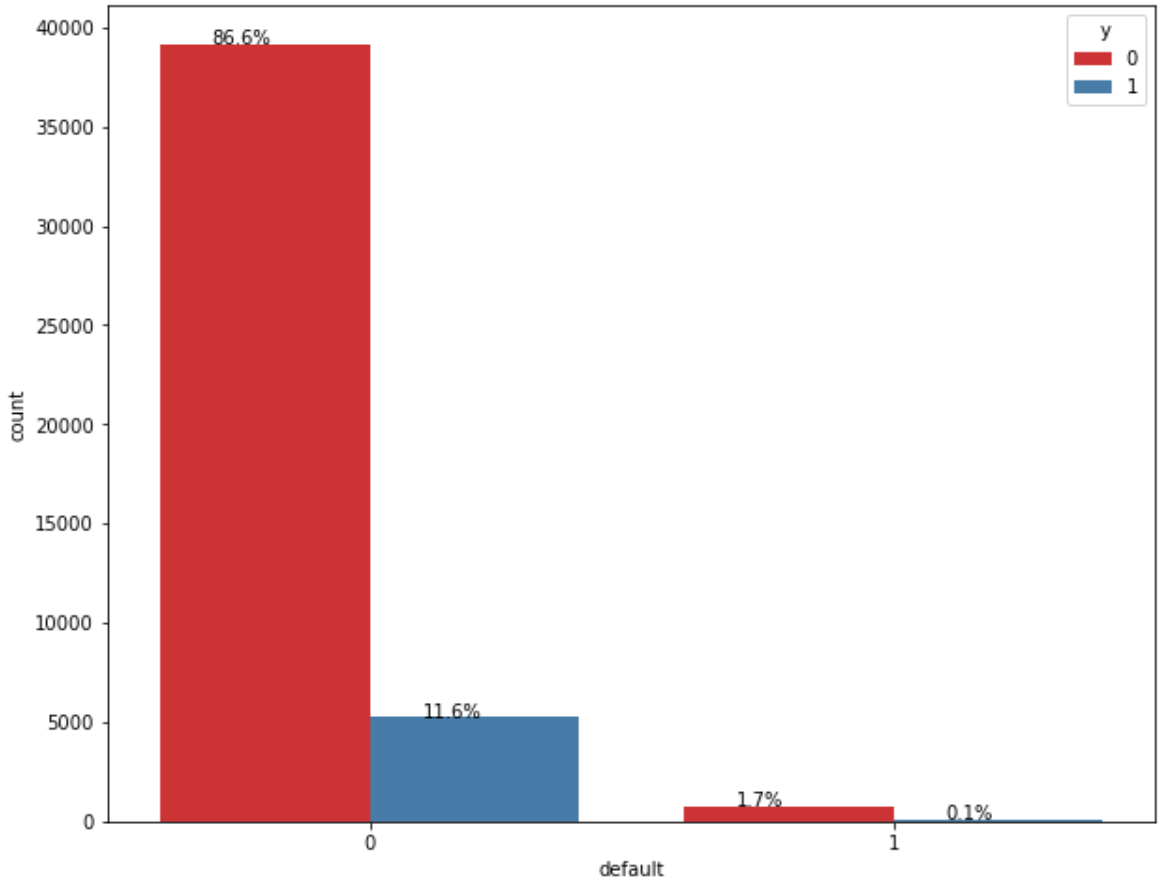


Hình 2.4: Tỷ lệ người trong hôn nhân có đăng ký tiền gửi hay không đăng ký

```
# Số lượng của cột có tín dụng trong tình trạng vỡ nợ
banktelemarket['default'].astype("category").value_counts()
0    44396
1      815
Name: default, dtype: int64
```

Số lượng người trong tình trạng không vỡ nợ: 44.396 người

Số lượng người trong tình trạng có vỡ nợ: 815 người



Hình 2.5: Tỷ lệ người trong vỡ nợ có đăng ký tiền gửi hay không đăng ký

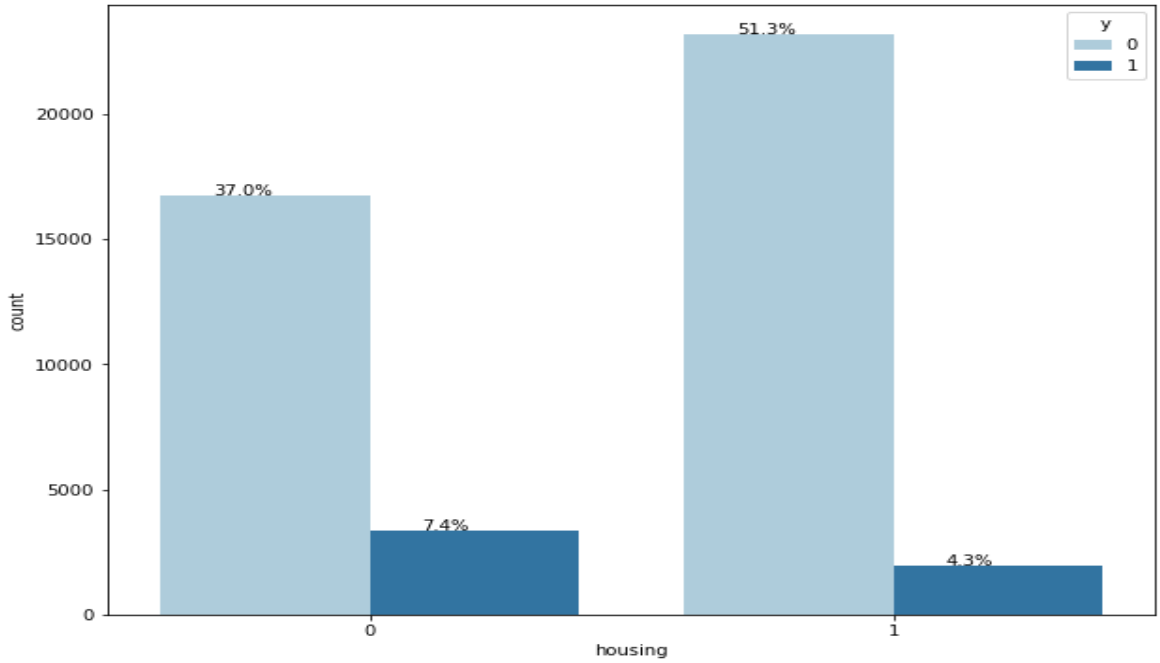
Số lượng của cột nhà ở: có cho vay mua nhà không?

banktelemarket['housing'].astype("category").value_counts()

1 25130

0 20081

Name: housing, dtype: int64



Hình 2.6: Tỷ lệ người có nhà có đăng ký tiền gửi hay không đăng ký

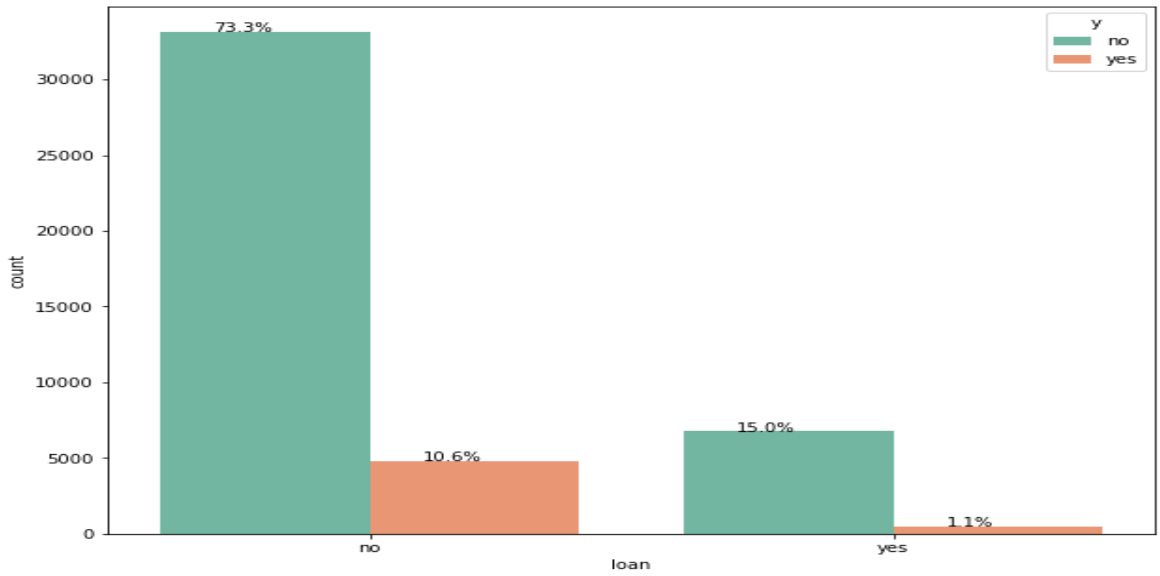
Số lượng của cột vay: có vay cá nhân không?

banktelemarket['loan'].astype("category").value_counts()

0 37967

1 7244

Name: loan, dtype: int64



Hình 2.7: Tỷ lệ người có khoảng vay có đăng ký tiền gửi hay không đăng ký

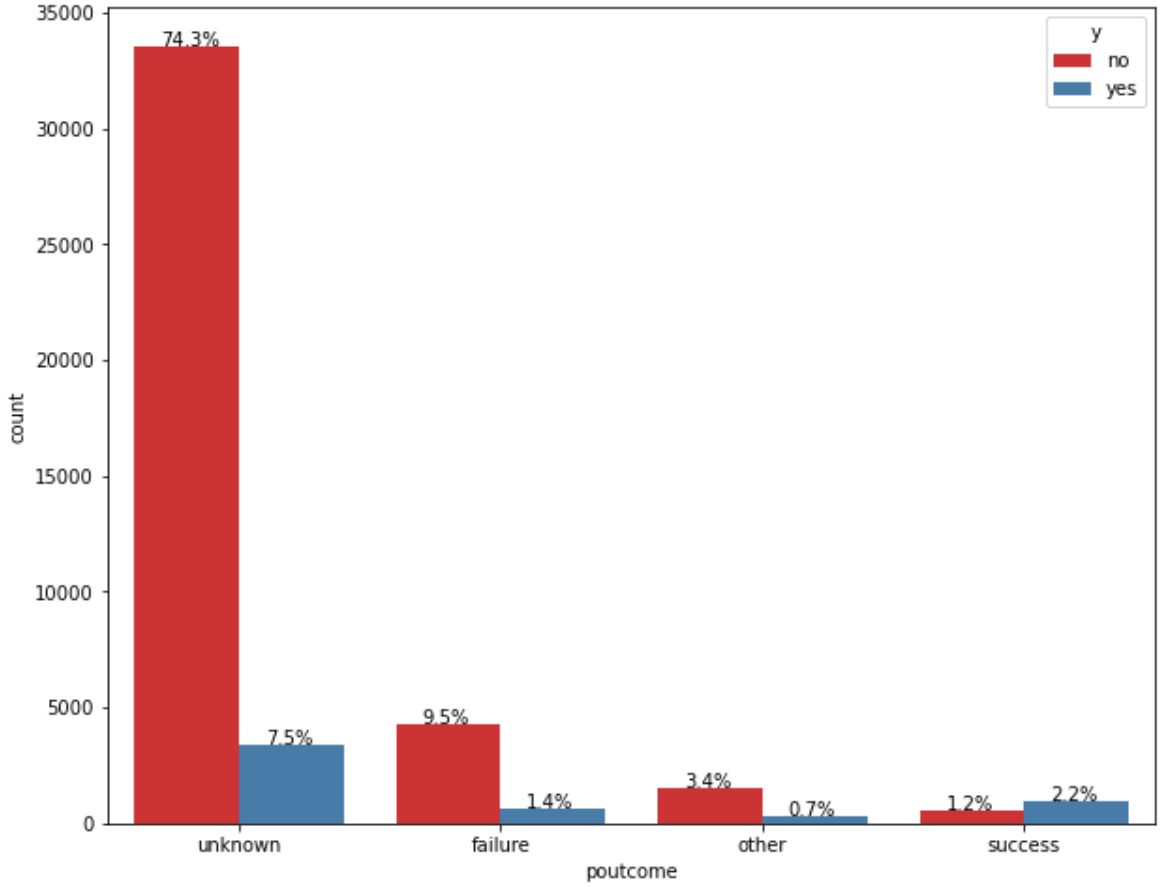
```
# Số lượng của cột của đợt tiếp thị trước đó (phân loại:
# 'không tồn tại', 'thất bại', 'khác', 'thành công')
banktelemarket['poutcome'].astype("category").value_counts()
unknown    36959
failure     4901
other       1840
success     1511
Name: poutcome, dtype: int64
```

Unknown (không xác định): 36.959 người

Failure (thất bại): 4.901 người

Other (khác): 1.840 người

Success (thành công): 1.511 người



Hình 2.8: Tỷ lệ người đợt trước đó có đăng ký tiền gửi hay không đăng ký

Trong chương này tìm hiểu tập dữ liệu ngân hàng bank-full.csv. Các thuộc tính đầu vào và đầu ra của dữ liệu dự đoán tiếp thị ngân hàng. Nói chung, bộ dữ liệu chứa dữ liệu tiếp thị có thể được sử dụng cho 2 mục tiêu kinh doanh khác nhau: Dự đoán kết quả của chiến dịch tiếp thị cho từng khách hàng và làm rõ các yếu tố ảnh hưởng đến kết quả chiến dịch. Điều này giúp tìm ra cách thực hiện các chiến dịch tiếp thị hiệu quả hơn. Tìm hiểu phân khúc khách hàng, sử dụng dữ liệu khách hàng đăng ký tiền gửi có kỳ hạn. Điều này giúp xác định hồ sơ của khách hàng, những người có nhiều khả năng mua sản phẩm hơn và phát triển các chiến dịch tiếp thị được nhắm mục tiêu hơn.

2.5. Mục tiêu của mô hình

- Làm sạch dữ liệu và loại bỏ các cột không mong muốn
- Mô tả sự phân bố của các tính năng khác nhau và mối tương quan giữa chúng.
- Thực hiện kỹ thuật tính năng để trích xuất các tính năng chính xác cho mô hình.
- Xây dựng mô hình

2.6. Đánh giá mô hình

Khi đã hoàn thành việc xây dựng mô hình và phân tích phần dư và đã đưa ra dự đoán trên bộ thử nghiệm, chỉ cần đảm bảo bạn sử dụng y_test và y_pred. Trong đó y_test là tập dữ liệu kiểm tra cho biến mục tiêu và y_pred là biến chứa các giá trị dự đoán của biến mục tiêu trên tập kiểm tra.

Chương 3: KHAI PHÁ DỮ LIỆU BẰNG MÔ HÌNH HỒI QUY LOGISTIC, CÂY QUYẾT ĐỊNH

3.1. Mô hình Logistic

Dữ liệu liên quan đến các đợt tiếp thị trực tiếp của một Tổ chức Ngân hàng Bồ Đào Nha. Các đợt tiếp thị dựa trên các cuộc gọi điện thoại. Thông thường, cần có nhiều hơn một liên hệ với cùng một khách hàng, để truy cập xem sản phẩm (tiền gửi có kỳ hạn ngân hàng) sẽ được đăng ký (có) hay không được chọn (không) (biến y). Chúng tôi muốn biết liệu chiến lược tiếp thị ngân hàng có thành công hay không, vì vậy chúng tôi cần chuyển biến kết quả thành 0 và 1 để thực hiện hồi quy logistic. Lưu ý rằng cột đầu tiên của tập dữ liệu là chỉ mục.

3.1.1. Các bước thực hiện mô hình

Bước 1: Nhập dữ liệu [7]

```
# nhập tất cả các thư viện được sử dụng trong nghiên cứu điển hình
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sys
import statsmodels.api as sm
import plotly.express as px
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.feature_selection import RFE
from sklearn.preprocessing import OrdinalEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import LabelEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.metrics import accuracy_score
from sklearn import tree
from sklearn.metrics import precision_score, recall_score, precision_recall_curve
from collections import Counter
from matplotlib import colors
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

- Bảy câu lệnh đầu tiên nhập các gói numpy, pandas, matplotlib.pyplot, seaborn, sys, statsmodels.api và plotly.express.

- Mười lăm câu lệnh tiếp theo nhập các mô-đun được chỉ định từ sklearn.
- Câu lệnh collections nhập module sử dụng để lưu trữ các bộ sưu tập dữ liệu.
- Matplotlib là một thư viện sử dụng để vẽ các đồ thị trong Python.
- Statsmodels tính năng thống kê

Bảng 3.1: Mô tả dữ liệu ngân hàng

```
# nhập dữ liệu và đọc nó
pd.options.display.max_columns=None
banktelemarket = pd.read_csv('C:\\input\\bank-full.csv', header=0, sep=';')
banktelemarket.head()
banktelemarket
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	no
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
45206	51	technician	married	tertiary	no	825	no	no	cellular	17	nov	977	3	-1	0	unknown	yes
45207	71	retired	divorced	primary	no	1729	no	no	cellular	17	nov	456	2	-1	0	unknown	yes
45208	72	retired	married	secondary	no	5715	no	no	cellular	17	nov	1127	5	184	3	success	yes
45209	57	blue-collar	married	secondary	no	668	no	no	telephone	17	nov	508	4	-1	0	unknown	no
45210	37	entrepreneur	married	secondary	no	2971	no	no	cellular	17	nov	361	2	188	11	other	no

45211 rows × 17 columns

Ta thấy có 45.211 dòng và 17 cột hiện có. Ta sẽ chỉ sử dụng một số cột từ những cột này để phát triển mô hình.

Bước 2: Kiểm tra dữ liệu

```
# đếm giá trị null của các cột để xem có cột đó có dòng dữ liệu nào trống không
banktelemarket.isnull().sum()
```

```
age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
y            0
dtype: int64
```

Để xem trong tập bank-full.csv cột nào có dòng dữ liệu trống không có số liệu?.

```
# hình dạng của dữ liệu
banktelemarket.shape
```

```
(45211, 17)
```

```
banktelemarket.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        45211 non-null  int64 
 1   job        45211 non-null  object
 2   marital    45211 non-null  object
 3   education  45211 non-null  object
 4   default    45211 non-null  object
 5   balance    45211 non-null  int64 
 6   housing    45211 non-null  object
 7   loan       45211 non-null  object
 8   contact    45211 non-null  object
 9   day        45211 non-null  int64 
 10  month      45211 non-null  object
 11  duration   45211 non-null  int64 
 12  campaign   45211 non-null  int64 
 13  pdays      45211 non-null  int64 
 14  previous   45211 non-null  int64 
 15  poutcome   45211 non-null  object
 16  y          45211 non-null  object
dtypes: int64(7), object(10)
memory usage: 5.9+ MB
```

Thông tin trên cho thấy tất cả 17 cột đều có số liệu ở mỗi dòng. Không có dòng dữ liệu trống ở mỗi cột (non-null).

Bước 3: Thao tác xử lý dữ liệu

```
# khía cạnh thống kê của khung dữ liệu (số liệu trong bộ dữ liệu bank-full.csv)
banktelemarket.head(10)
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	no
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no
5	35	management	married	tertiary	no	231	yes	no	unknown	5	may	139	1	-1	0	unknown	no
6	28	management	single	tertiary	no	447	yes	yes	unknown	5	may	217	1	-1	0	unknown	no
7	42	entrepreneur	divorced	tertiary	yes	2	yes	no	unknown	5	may	380	1	-1	0	unknown	no
8	58	retired	married	primary	no	121	yes	no	unknown	5	may	50	1	-1	0	unknown	no
9	43	technician	single	secondary	no	593	yes	no	unknown	5	may	55	1	-1	0	unknown	no

Tiếp theo, chúng ta cần làm sạch dữ liệu. Dữ liệu có thể chứa một số hàng có NaN. Để loại bỏ các hàng có giá trị NaN, chúng ta sử dụng lệnh sau:

banktelemarket = banktelemarket.dropna()

Tập dữ liệu **bank-full.csv** không chứa bất kỳ hàng nào có giá trị NaN, vì vậy bước này không thực sự bắt buộc. Tuy nhiên, nói chung rất khó để phát hiện ra các hàng có giá trị NaN như vậy trong một cơ sở dữ liệu khổng lồ. Vì vậy, luôn an toàn hơn khi chạy câu lệnh trên để làm sạch dữ liệu.

Điều tiếp theo cần làm là kiểm tra sự phù hợp của từng cột đối với mô hình mà ta đang xây dựng.

Bất cứ khi thực hiện một cuộc khảo sát, chúng ta thu thập càng nhiều thông tin càng tốt từ khách hàng, với ý tưởng rằng thông tin này sẽ hữu ích theo cách này hay cách khác vào thời điểm sau này. Để giải quyết vấn đề hiện tại, chúng ta phải thu thập thông tin có liên quan trực tiếp đến vấn đề của mô hình.

Bây giờ, chúng ta hãy xem cách chọn các trường dữ liệu hữu ích.

print(list(banktelemarket.columns))

['age', 'job', 'marital', 'education', 'default', 'balance', 'housing', 'loan', 'contact', 'day', 'month', 'duration', 'campaign', 'pdays', 'previous', 'poutcome', 'y']

Kết quả hiển thị tên của tất cả các cột trong cơ sở dữ liệu. Cột cuối cùng “y” là một giá trị Boolean cho biết liệu khách hàng này có gửi tiền có kỳ hạn với ngân hàng hay không? Các giá trị của trường này là “yes” hoặc “no”.

Bước 4: Loại bỏ cột không mong muốn:

Kiểm tra tên cột, chúng sẽ biết rằng một số cột không có ý nghĩa gì đối với vấn đề hiện tại. Ví dụ, các trường như tháng, ngày_thứ_lần, đợt, v.v. không có ích gì đối với chúng ta. Chúng ta sẽ loại bỏ các cột này khỏi cơ sở dữ liệu của chúng ta. Để thả một cột, chúng ta sử dụng câu lệnh drop như hình dưới đây:

banktelemarket.drop(banktelemarket.columns[[0, 3, 5, 8, 9, 10, 11, 12, 13, 14]],
  axis = 1, inplace = True)

Câu lệnh có nghĩa là drop (xóa) các cột 0, 3, 5, 8, v.v. Để đảm bảo rằng index (chỉ mục) được chọn đúng, ta làm như sau:

banktelemarket.columns[4]

'loan'

Điều này sẽ in tên cột cho index đã cho. Sau khi loại bỏ các cột không bắt buộc, kiểm tra dữ liệu bằng câu lệnh head. Kết quả màn hình được hiển thị như sau:

	job	marital	default	housing	loan	poutcome	y
0	management	married	no	yes	no	unknown	no
1	technician	single	no	yes	no	unknown	no
2	entrepreneur	married	no	yes	yes	unknown	no
3	blue-collar	married	no	yes	no	unknown	no
4	unknown	single	no	no	no	unknown	no

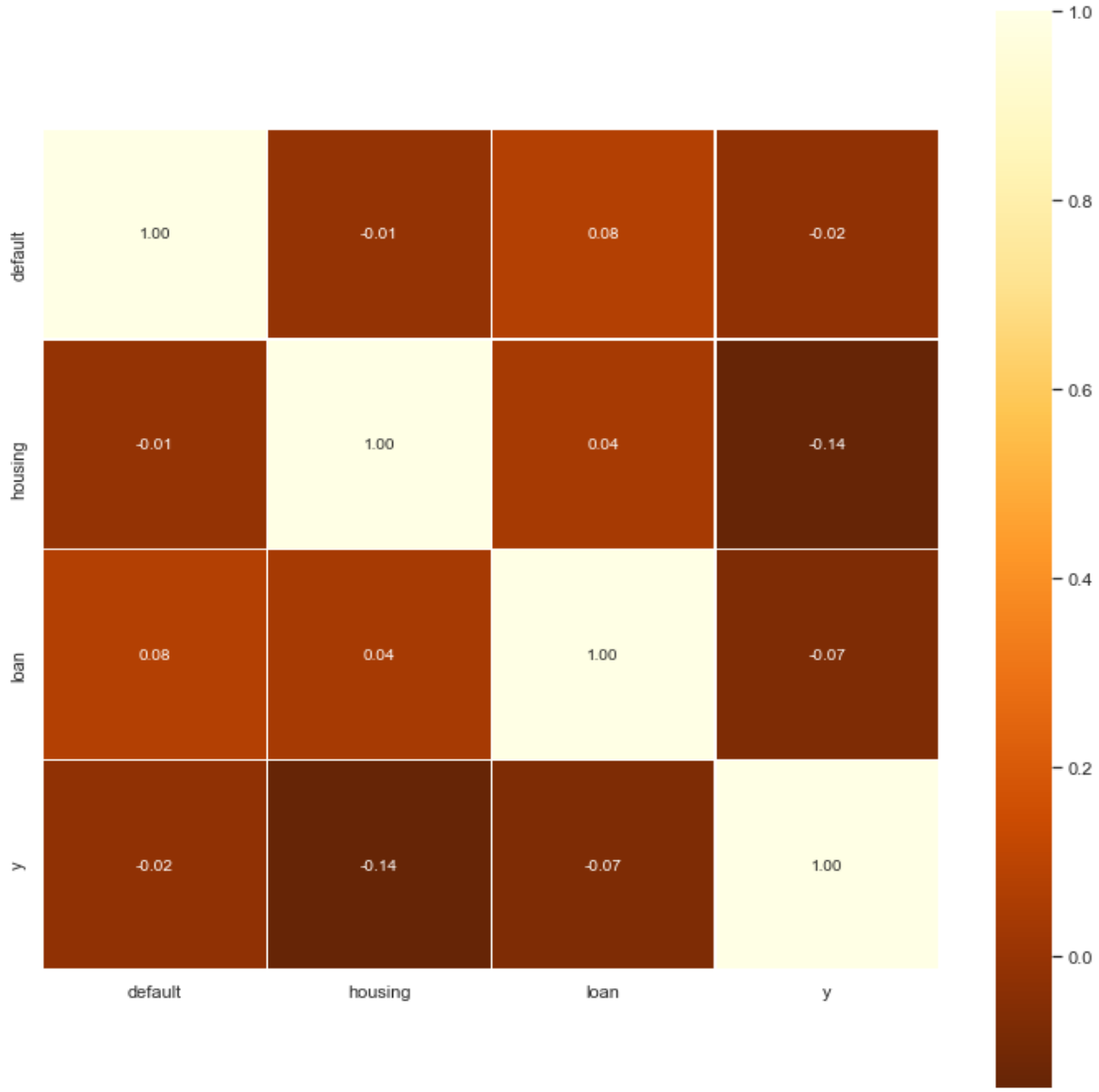
Bây giờ, ta chỉ có những cột quan trọng đối với việc phân tích và dự đoán dữ liệu. Tầm quan trọng của Data Scientist (nhà khoa học) xuất hiện ở bước này. Data Scientist phải chọn các cột thích hợp để xây dựng mô hình.

Ví dụ: loại công việc thoạt nhìn có thể không thuyết phục được mọi người đưa vào cơ sở dữ liệu nhưng nó sẽ là một giá trị rất hữu ích. Không phải tất cả các khách hàng sẽ mở tín dụng. Những người có thu nhập thấp hơn có thể không mở tín dụng trong khi những người có thu nhập cao hơn thường sẽ gửi tiền của họ vào tín dụng. Vì vậy, loại công việc trở nên phù hợp đáng kể trong trường hợp này. Như vậy, hãy chọn cẩn thận các cột mà chúng ta cảm thấy có liên quan cho phân tích của mình.

Trong phần tiếp theo chúng ta sẽ chuẩn bị dữ liệu để xây dựng mô hình. Thay đổi các cột defaul, housing, loan, y có giá trị Yes và No thành giá trị 1 và 0.

```
# danh sách các biến cần được thay đổi
col = ['default','housing','loan','y']
# định nghĩa hàm
def convert(x):
return x.map({'yes':1,'no':0})
# gọi hàm
banktelemarket[col] = banktelemarket[col].apply(convert)
```

```
# Hình dung các mối tương quan của tính năng
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(banktelemarket._get_numeric_data().astype(float).corr(),
        square=True, cmap='YlOrBr_r', linewidths=.5,
        annot=True, fmt='.2f').figure.tight_layout()
plt.show()
```



Hình 3.1: Biểu đồ tương quan với cột mục tiêu

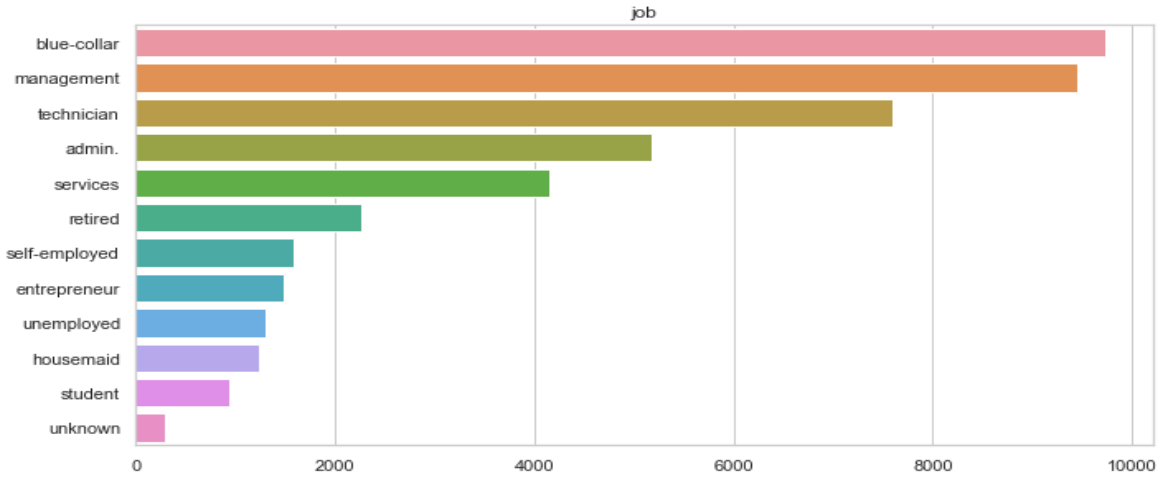
Chúng ta sẽ không bỏ bất kỳ cột nào vì không cột nào có tương quan cao với cột mục tiêu.

Bước 5: Trực quan hóa và Phân tích Dữ liệu

Phân phối dữ liệu danh mục

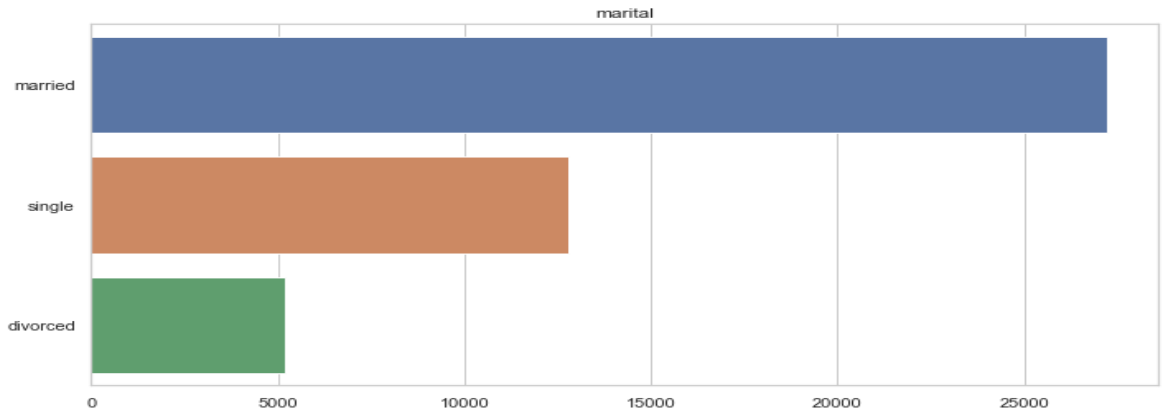
Chúng ta sẽ bắt đầu với phân tích khám phá các cột dữ liệu bằng cách sử dụng gói seaborn để vẽ biểu đồ.

```
category_features = banktelemarket.select_dtypes(include=['object', 'bool']).columns.values
for col in category_features:
    plt.figure(figsize=(10,5))
    sns.barplot(banktelemarket[col].value_counts().values,
                banktelemarket[col].value_counts().index, data=banktelemarket)
    plt.title(col)
    plt.tight_layout()
```



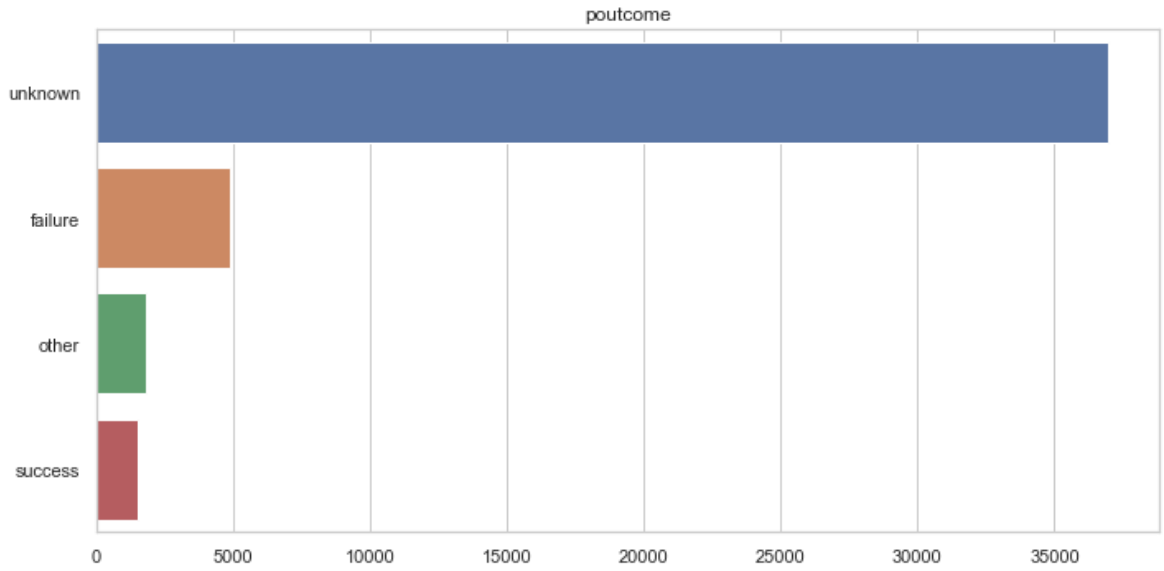
Hình 3.2: Biểu đồ công việc

Job - Công việc: Đối tượng của các chiến dịch này chủ yếu nhắm mục tiêu là blue-collar, nhân viên quản lý và kỹ thuật viên.



Hình 3.3: Biểu đồ tình trạng hôn nhân

Marital - Tình trạng hôn nhân: Đa số đã có gia đình; khách hàng đã kết hôn gấp đôi những người độc thân.

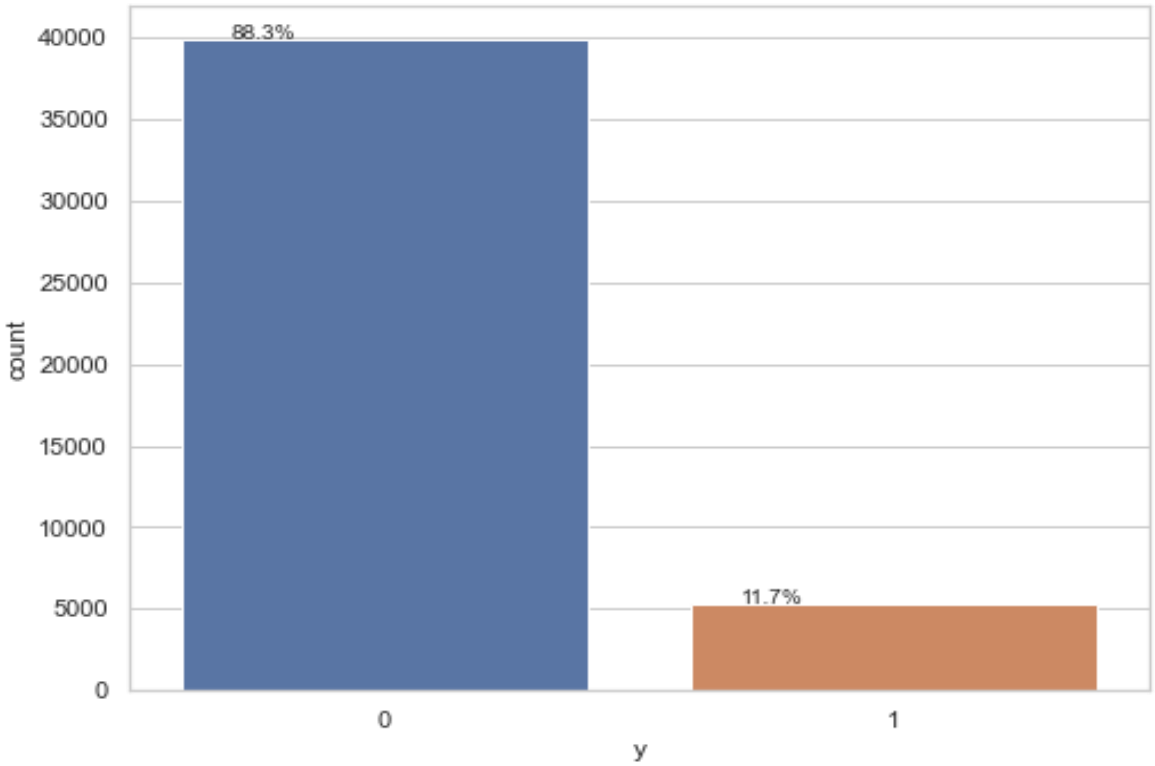


Hình 3.4: Biểu đồ đợt tiếp thị trước đó

Đợt tiếp thị trước đó – poutcome: Khách hàng unknown (*không xác định*) chiếm đa số, kế đến là failure (*thất bại*).

Bước 6: Đăng ký tiền gửi có kỳ hạn

```
# Kiểm tra xem tập dữ liệu có cân bằng hay không
plt.figure(figsize = (8,6))
total = len(banktelemarket["y"])
ax = sns.countplot(x = 'y', data = banktelemarket)
for p in ax.patches:
   ax.annotate('{:.1f}%'.format(100*p.get_height()/total),
               (p.get_x()+0.1, p.get_height()+5))
plt.show()
```

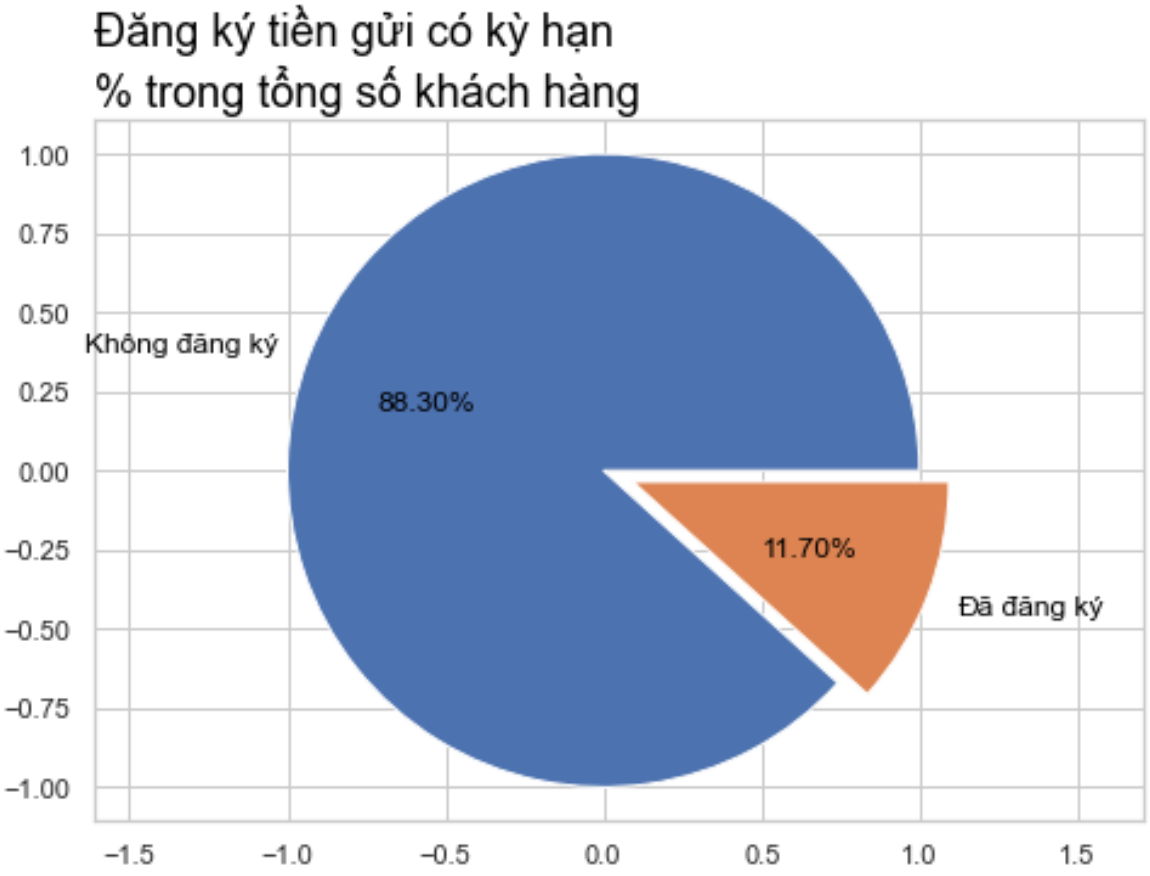


Hình 3.5: Tỷ lệ khách hàng đăng ký tiền gửi và không đăng ký

Từ sơ đồ trên, chúng ta có thể nói rằng tập dữ liệu gần như là không cân bằng.

```
# Biểu đồ tròn
labels = ["Không đăng ký", "Đã đăng ký"]
explode = (0, 0.1)  # chỉ "loại ra" phần thứ 2 (Ví dụ. 'Đã đăng ký')
# mô tả hình ảnh
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.pie(banktelemarket['y'].value_counts(),
       labels = labels,
       explode = explode,
       autopct ='%1.2f%%',
       frame = True,
       textprops = dict(color ="black", size=12))

ax.axis('equal')
plt.title('Đăng ký tiền gửi có kỳ hạn\n% trong tổng số khách hàng',
     loc='left',
     color = 'black',
     fontsize = '18')
plt.show()
```

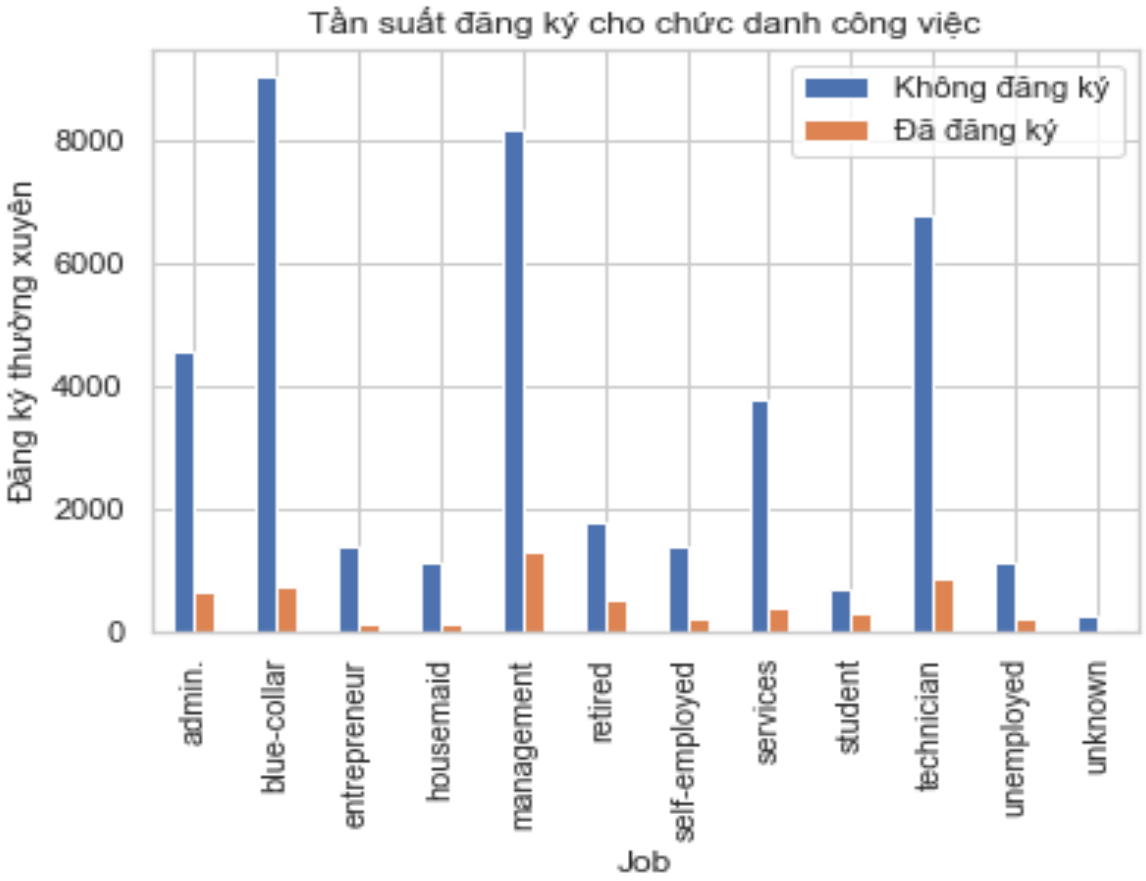


Hình 3.6: Biểu đồ tròn tỷ lệ đăng ký tiền gửi

Qua các biểu đồ, ta thấy chỉ 11,70 % khách hàng đăng ký tiền gửi có kỳ hạn. Các lớp của chúng ta không cân bằng khi giá trị khả quan (đã đăng ký) chỉ là 11,70%. Khách hàng không đăng ký tiền gửi có kỳ hạn chiếm tới 88.30%.

Bước 7: Những khách hàng nào có nhiều khả năng đăng ký tiền gửi có kỳ hạn hơn?

```
table = pd.crosstab(banktelemarket.job, banktelemarket.y)
table.columns = ['Không đăng ký', 'Đã đăng ký']
table.plot(kind='bar')
plt.grid(True)
plt.title('Tần suất đăng ký cho chức danh công việc')
plt.xlabel('Job')
plt.ylabel('Đăng ký thường xuyên')
```



Hình 3.7: Tỷ lệ đăng ký thường xuyên

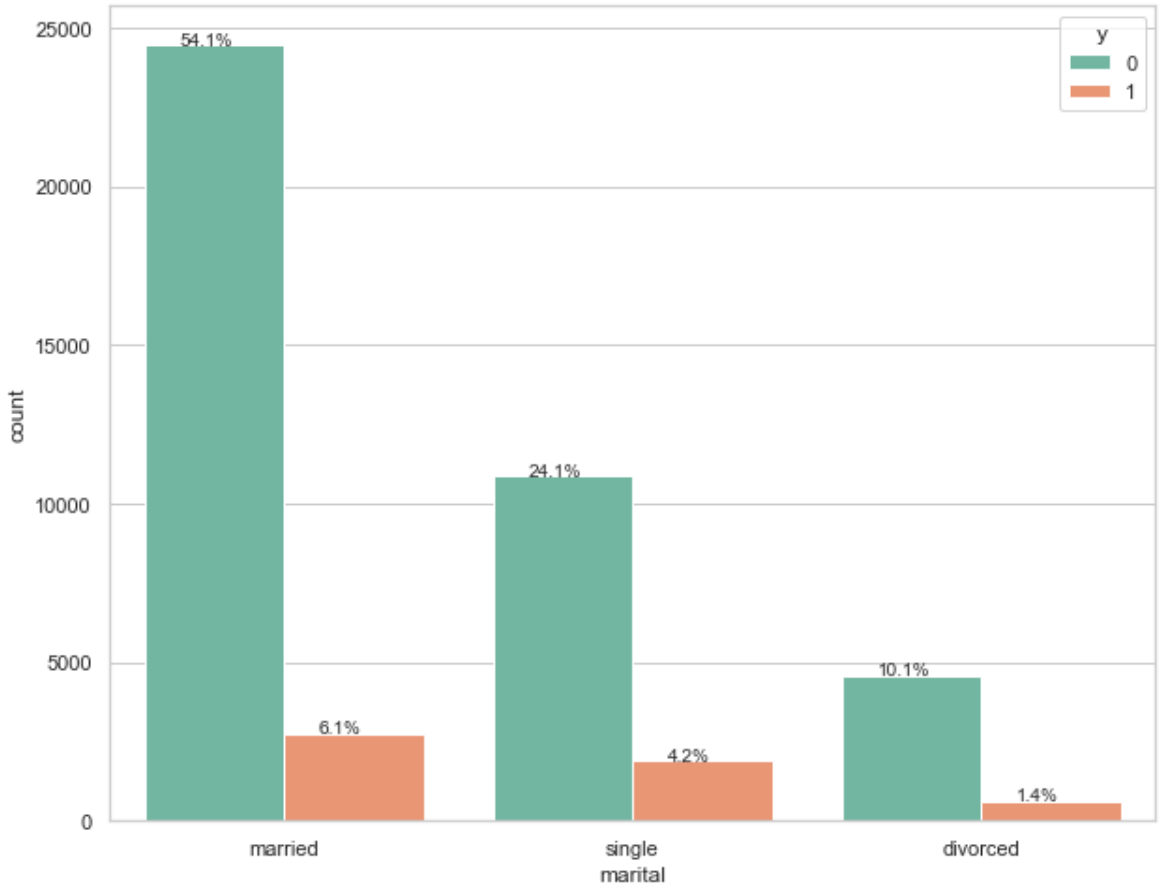
```
table = pd.crosstab(banktelemarket.job, banktelemarket.y)
table = round(table.div(table.sum(axis=1), axis=0).mul(100), 2)
table.columns=['notsubcribed', 'subcribed']
table.sort_values(by=['subcribed'], ascending=False).loc[:, 'subcribed']
```

```
job
student          28.68
retired          22.79
unemployed       15.50
management       13.76
admin.           12.20
self-employed    11.84
unknown          11.81
technician       11.06
services          8.88
housemaid         8.79
entrepreneur      8.27
blue-collar       7.27
Name: subcribed, dtype: float64
```

Khách hàng mục tiêu là quản trị viên, người yêu thích công việc kinh doanh, nhưng tần suất sinh viên và người đã nghỉ hưu đăng ký tiền gửi có kỳ hạn khá cao (28,68% đối với sinh viên và 22,79% đối với người đã nghỉ hưu).

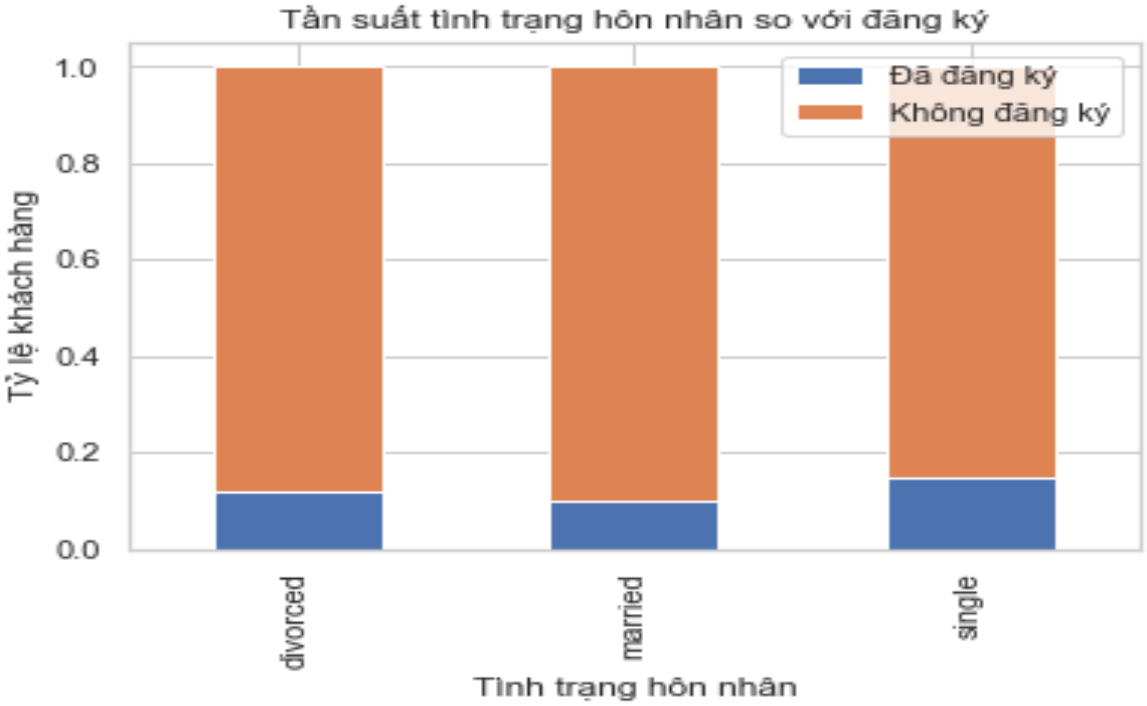
Vai trò của tình trạng hôn nhân trong đăng ký

```
plt.figure(figsize = (10,8))
total = len(banktelemarket["marital"])
ax = sns.countplot(x = 'marital', data = banktelemarket, hue = 'y', palette = 'Set2')
for p in ax.patches:
   ax.annotate('{:.1f}%'.format(100*p.get_height()/total), (p.get_x()+0.1, p.get_height()+5))
plt.show()
```



Hình 3.8: Tỷ lệ tình trạng hôn nhân

```
table = pd.crosstab(banktelemarket.marital,banktelemarket.y)
table = table.div(table.sum(1).astype(float), axis=0)
table.columns = ['Không đăng ký', 'Đã đăng ký']
# Ordering stacked bars and plot the chart
table[['Đã đăng ký', 'Không đăng ký']].plot(kind='bar', stacked=True)
plt.title('Tần suất tình trạng hôn nhân so với đăng ký')
plt.xlabel('Tình trạng hôn nhân')
plt.ylabel('Tỷ lệ khách hàng')
```

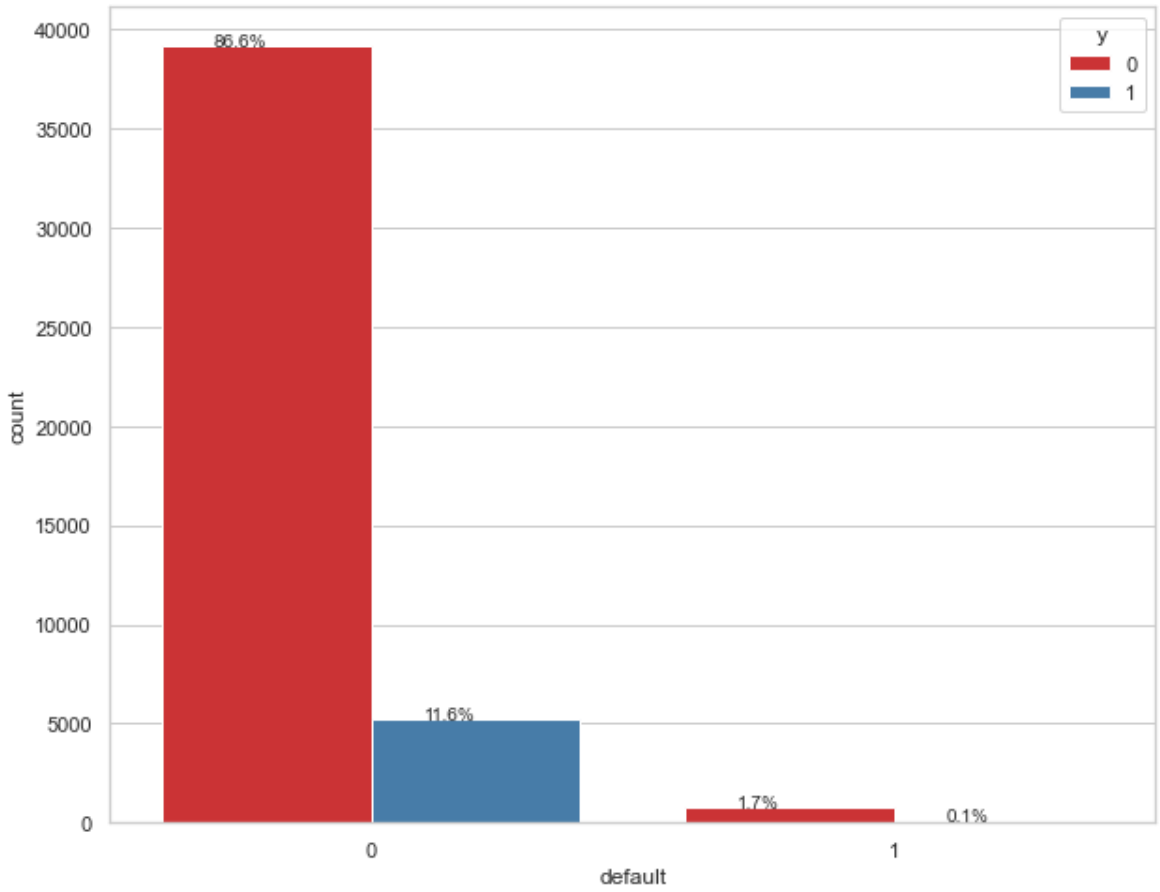


Hình 3.9: Tầng suất tình trạng hôn nhân so với đăng ký

Không có tác động đáng kể của tình trạng hôn nhân đến đăng ký của khách hàng.

Vai trò của tín dụng trong tình trạng vỡ nợ? (nhị phân: "yes", "no") trong đăng ký?

```
plt.figure(figsize = (10,8))
total = len(banktelemarket["default"])
sns.set_palette("Paired")
ax = sns.countplot(x = 'default', data = banktelemarket, hue = 'y', palette = 'Set1')
for p in ax.patches:
    ax.annotate('{:.1f}%'.format(100*p.get_height()/total), (p.get_x()+0.1, p.get_height()+5))
plt.show()
```

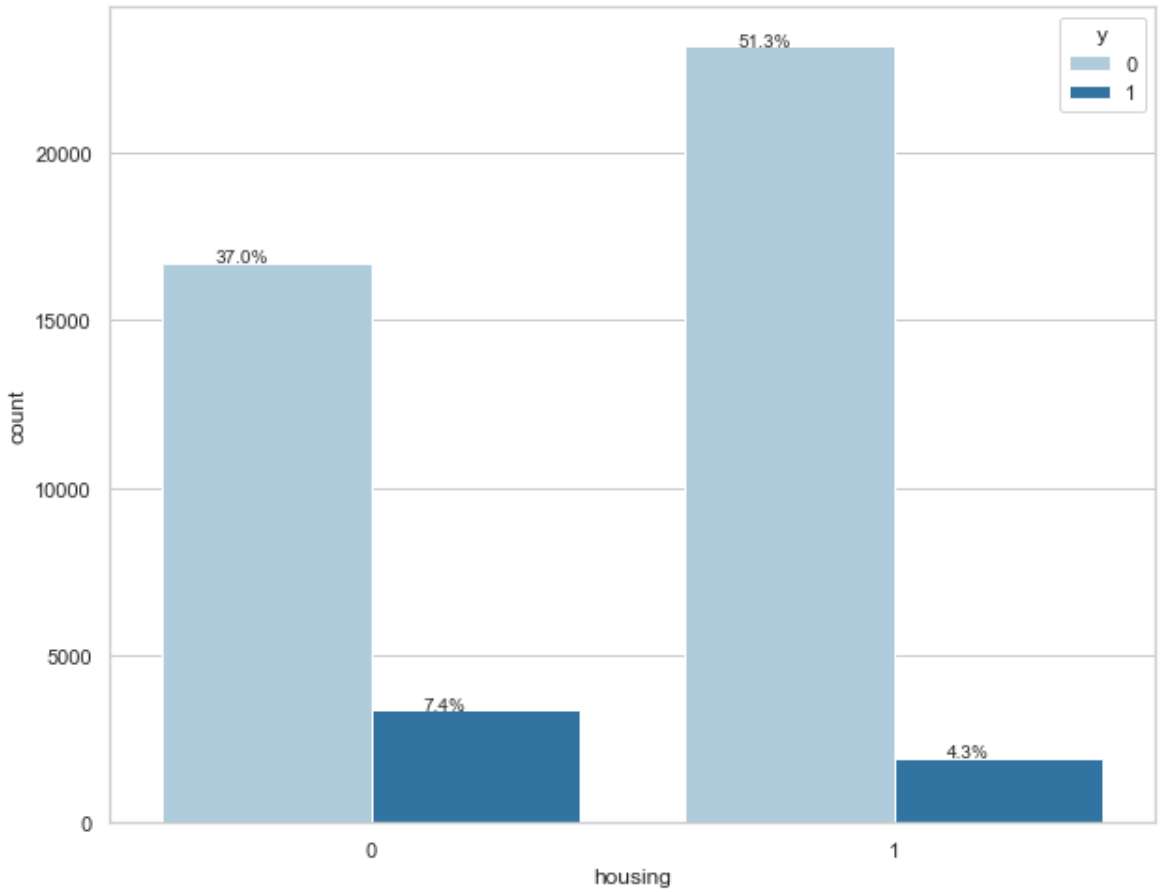


Hình 3.10: Tỷ lệ khách hàng có tín dụng vỡ nợ

Có sự phân bổ đồng đều giữa các khách hàng không có tín dụng trong tình trạng vỡ nợ, do đó tính năng này sẽ ít đóng góp hơn vào việc dự đoán mà khách hàng có đăng ký tiền gửi có kỳ hạn hay không?

Vai trò của nhà ở trong đăng ký?

```
plt.figure(figsize = (10,8))
total = len(banktelemarket["housing"])
ax = sns.countplot(x = 'housing', data = banktelemarket, hue = 'y')
for p in ax.patches:
    ax.annotate('{:.1f}%'.format(100*p.get_height()/total), (p.get_x()+0.1, p.get_height()+5))
plt.show()
```



Hình 3.11: Tỷ lệ khách hàng có nhà ở

Khách hàng có nhà ở có tỷ lệ đăng ký tiền gửi có kỳ hạn thấp 4.3%

Để tạo bộ phân loại, chúng ta phải chuẩn bị dữ liệu ở định dạng được yêu cầu bởi mô-đun xây dựng bộ phân loại. Chúng ta chuẩn bị dữ liệu bằng cách thực hiện **One Hot Encoding** mã hóa) để chuyển đổi các biến thứ tự và phân loại thành các giá trị số.

Ta sẽ thảo luận về ý nghĩa của việc mã hóa dữ liệu. Đầu tiên, chạy đoạn code sau:

data = pd.get_dummies(banktelemarket, columns =['job', 'marital', 'default', 'housing', 'loan', 'poutcome'])

Như trong command câu lệnh trên sẽ tạo ra one hot encoding. Nó đã tạo ra những gì? Kiểm tra dữ liệu đã tạo được gọi là **“data”** như sau :

data.head()

	y	job_admin.	job_blue-collar	job_entrepreneur	job_housemaid	job_management	job_retired	job_self-employed	job_services	job_student	job_technician	job_unemploy
0	0	0	0	0	0	1	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	1	
2	0	0	0	1	0	0	0	0	0	0	0	
3	0	0	1	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	0	

5 dòng x 25 cột

Để hiểu dữ liệu trên, ta sẽ liệt kê tên cột bằng cách chạy lệnh data.columns

```
data.columns

Index(['y', 'job_admin.', 'job_blue-collar', 'job_entrepreneur',
       'job_housemaid', 'job_management', 'job_retired', 'job_self-employed',
       'job_services', 'job_student', 'job_technician', 'job_unemployed',
       'job_unknown', 'marital_divorced', 'marital_married', 'marital_single',
       'default_0', 'default_1', 'housing_0', 'housing_1', 'loan_0', 'loan_1',
       'poutcome_failure', 'poutcome_other', 'poutcome_success',
       'poutcome_unknown'],
      dtype='object')
```

Bây giờ, mình sẽ giải thích cách one hot encoding thực hiện bằng lệnh **get_dummies**. Cột đầu tiên trong cơ sở dữ liệu mới được tạo là trường "y" cho biết liệu khách hàng này đã đăng ký tín dụng hay chưa? Bây giờ, hãy xem xét các cột được mã hóa. Cột được mã hóa đầu tiên là **"job"**. Trong cơ sở dữ liệu, sẽ thấy rằng cột **"job"** có nhiều giá trị có thể có như "admin", "blue-collar", "entrepreneur", v.v. Đối với mỗi giá trị, một cột mới được tạo trong cơ sở dữ liệu, với tên cột được thêm vào dưới dạng tiền tố.

Do đó, ta có các cột được gọi là **"job_admin", "job_blue-collar"**, v.v. Đối với mỗi cột được mã hóa trong cơ sở dữ liệu gốc, sẽ tìm thấy danh sách các cột được thêm vào cơ sở dữ liệu đã tạo với tất cả các giá trị có thể mà cột đó nhận trong cơ sở dữ liệu gốc. Kiểm tra cẩn thận danh sách các cột để hiểu cách dữ liệu được ánh xạ tới cơ sở dữ liệu mới.

Để hiểu dữ liệu được tạo ta sẽ in ra toàn bộ dữ liệu bằng lệnh data. Kết quả như sau:

```
data.head(9)
```

	y	job_admin.	job_blue-collar	job_entrepreneur	job_housemaid	job_management	job_retired	job_self-employed	job_services	job_student	job_technician	job_unemplo
0	0	0	0	0	0	1	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	1	
2	0	0	0	1	0	0	0	0	0	0	0	
3	0	0	1	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	0	
5	0	0	0	0	0	1	0	0	0	0	0	
6	0	0	0	0	0	1	0	0	0	0	0	
7	0	0	0	1	0	0	0	0	0	0	0	
8	0	0	0	0	0	0	1	0	0	0	0	
9	0	0	0	0	0	0	0	0	0	0	1	
10	0	1	0	0	0	0	0	0	0	0	0	
11	0	1	0	0	0	0	0	0	0	0	0	
12	0	0	0	0	0	0	0	0	0	0	1	
13	0	0	0	0	0	0	0	0	0	0	1	
14	0	0	0	0	0	0	0	0	1	0	0	
15	0	0	0	0	0	0	1	0	0	0	0	
16	0	1	0	0	0	0	0	0	0	0	0	
17	0	0	1	0	0	0	0	0	0	0	0	
18	0	0	0	0	0	0	1	0	0	0	0	
19	0	0	0	0	0	0	0	0	1	0	0	

Một phần màn hình xuất ra bên dưới cơ sở dữ liệu được hiển thị ở đây để tham khảo. Để hiểu dữ liệu được ánh xạ (mapped), chúng ta hãy kiểm tra hàng thứ 4.

	y	job_admin.	job_blue-collar	job_entrepreneur	job_housemaid	job_management	job_retired	job_self-employed	job_services	job_student	job_technician	job_unemploye
0	0	0	0	0	0	1	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	1	
2	0	0	0	1	0	0	0	0	0	0	0	
3	0	0	1	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	0	

job_unemployed	marital_divorced	marital_married	marital_single	default_0	default_1	housing_0	housing_1	loan_0	loan_1
0	0	1	0	1	0	0	1	1	0
0	0	0	1	1	0	0	1	1	0
0	0	1	0	1	0	0	1	0	1
0	0	1	0	1	0	0	1	1	0
0	0	0	1	1	0	1	0	1	0

Cho ta thấy khách hàng này chưa đăng ký tín dụng như được chỉ ra bởi giá trị trong cột "y". Khách hàng này là khách hàng “blue-collar”. Cuộn qua theo chiều ngang sẽ cho bạn biết rằng khách hàng này có “housing (*nhà ở*) ” và không có “loan (*khoản vay*)” nào.

Ta cần xử lý thêm một số dữ liệu trước khi có thể bắt đầu xây dựng mô hình của mình.

Nếu ta kiểm tra các cột trong cơ sở dữ liệu được ánh xạ, sẽ thấy sự hiện diện của một số cột kết thúc bằng "không xác định" (unknown). Ví dụ: kiểm tra cột ở chỉ mục 12 bằng lệnh sau:

```
data.columns[12]
'job_unknown'
```

Điều này cho thấy công việc khách hàng được chỉ định là không xác định. Rõ ràng, không có ích gì khi đưa các cột như vậy vào phân tích và xây dựng mô hình cả. Do đó, tất cả các cột có giá trị "không xác định" sẽ bị loại bỏ. Điều này được thực hiện như sau:

```
data.drop(data.columns[[12]], axis=1, inplace=True)
```

Đảm bảo rằng chỉ định số cột chính xác. Trong trường hợp không chắc chắn, có thể kiểm tra tên cột bất cứ lúc nào bằng cách chỉ định chỉ mục của nó trong lệnh cột như được mô tả trước đó.

Sau khi loại bỏ các cột không mong muốn, chúng ta có thể kiểm tra danh sách cuối cùng của các cột như sau:

```
data.columns
Index(['y', 'job_admin.', 'job_blue-collar', 'job_entrepreneur',
       'job_housemaid', 'job_management', 'job_retired', 'job_self-employed',
       'job_services', 'job_student', 'job_technician', 'job_unemployed',
       'marital_divorced', 'marital_married', 'marital_single', 'default_0',
       'default_1', 'housing_0', 'housing_1', 'loan_0', 'loan_1',
       'poutcome_failure', 'poutcome_other', 'poutcome_success',
       'poutcome_unknown'],
      dtype='object')
```

Bây giờ dữ liệu đã sẵn sàng xây dựng mô hình. Ta chia toàn bộ tập dữ liệu thành hai phần, giả sử 80/20 phần trăm. Sử dụng 80% dữ liệu để xây dựng mô hình và 20% phần còn lại để kiểm tra độ chính xác trong dự đoán của mô hình đã tạo. Ở đây, chúng ta chia dữ liệu thành tập huấn luyện và thử nghiệm để chúng ta có thể điều chỉnh và đánh giá mô hình. Chúng ta sẽ sử dụng hàm train_test_split () từ scikit-learning và sử dụng 80% dữ liệu đào tạo (train) và 20% để kiểm tra (test).

Trước khi chia dữ liệu, ta tách dữ liệu thành hai mảng X và Y. Mảng X chứa tất cả các tính năng (cột dữ liệu) mà ta muốn phân tích và mảng Y là mảng một chiều gồm các giá trị boolean là đầu ra của dự đoán.

Đầu tiên, thực thi câu lệnh Python sau để tạo mảng X:

X = data.iloc[:,1:]

Để kiểm tra nội dung của X, sử dụng **head** để in một vài bản ghi ban đầu. Mảng X như sau :

```
X.head ()
```

	job_admin.	job_blue-collar	job_entrepreneur	job_housemaid	job_management	job_retired	job_self-employed	job_services	job_student	job_technician	job_unemployed
0	0	0	0	0	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0
2	0	0	1	0	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0

Mảng có một số hàng và 24 cột. Tiếp theo, ta sẽ tạo mảng đầu ra chứa các giá trị “y”. Để tạo một mảng cho cột giá trị dự đoán, hãy sử dụng câu lệnh Python sau:

Y = data.iloc[:,0]

Kiểm tra nội dung bằng head(). Kết quả như sau:

Y.head()

0 0

1 0

2 0

3 0

4 0

Name: y, dtype: int64

Bây giờ, chia nhỏ dữ liệu bằng lệnh sau:

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state=0)

print("Số giao dịch tập dữ liệu X_train: ", X_train.shape)

print("Số giao dịch tập dữ liệu Y_train: ", Y_train.shape)

print("Số giao dịch tập dữ liệu X_test: ", X_test.shape)

print("Số giao dịch tập dữ liệu Y_test: ", Y_test.shape)

Số giao dịch tập dữ liệu X_train: (36168, 24)

Số giao dịch tập dữ liệu Y_train: (36168,)

Số giao dịch tập dữ liệu X_test: (9043, 24)

Số giao dịch tập dữ liệu Y_test: (9043,)

80% dữ liệu đào tạo (train) trong tổng số 45.211 dòng của tập dữ liệu là 36.168 trong 24 cột tập dữ liệu.

20% để kiểm tra (test) trong tổng số 45.211 dòng của tập dữ liệu là 9.043 trong 24 cột tập dữ liệu.

job_admin.	job_blue-collar	Job_entrepreneur	job_housemaid	Job_management	job_retired	job_self-employed	job_services	job_student	job_technician	Job_unemployed	Marital_divorced	marital_married	marital_single	default_0	default_1	housing_0	housing_1	loan_0	loan_1	poutcome_failure	poutcome_other	poutcome_success	poutcome_unknown

Bốn mảng được tạo ra gọi là **X_train, Y_train, X_test** và **Y_test**. Chúng ta có thể kiểm tra nội dung của các mảng này bằng cách sử dụng lệnh head. Ta sử dụng các mảng X_train và Y_train để đào tạo mô hình và các mảng X_test , Y_test để kiểm tra và xác thực.

Bây giờ, ta tiến hành xây dựng bộ phân loại. Ta sẽ sử dụng một mô hình dựng sẵn từ sklearn.

Việc tạo bộ phân loại hồi quy Logistic từ bộ công cụ sklearn là không cần thiết và được thực hiện trong một câu lệnh sau:

classifier = LogisticRegression(solver='lbfgs',random_state=0)

Sau khi bộ phân loại được tạo, sẽ cung cấp dữ liệu huấn luyện (Training) của mình vào bộ phân loại để nó có thể điều chỉnh các thông số bên trong và sẵn sàng dự đoán dữ liệu trong tương lai. Để điều chỉnh trình phân loại câu lệnh sử dụng như sau:

classifier.fit(X_train, Y_train)

Để kiểm tra trình phân loại, ta sử dụng dữ liệu kiểm tra được tạo ở giai đoạn trước. Gọi phương thức dự đoán trên đối tượng đã tạo và truyền vào mảng X của dữ liệu thử nghiệm như sau:

predicted_y = classifier.predict(X_test)

Điều này tạo ra một mảng một chiều cho toàn bộ tập dữ liệu huấn luyện đưa ra dự đoán cho mỗi hàng trong mảng X. Bạn có thể kiểm tra mảng này bằng cách sử dụng lệnh sau:

predicted_y

array([0, 0, 0, ..., 0, 0, 0], dtype=int64)

Kết quả đầu ra chỉ ra rằng ba khách hàng đầu tiên và cuối cùng không phải là ứng viên tiềm năng cho Tiền gửi có kỳ hạn. Bạn có thể kiểm tra toàn bộ mảng để phân loại khách hàng tiềm năng

3.1.2. Kết quả mô hình dự đoán

```
for x in range(len(predicted_y)):
  if (predicted_y[x] == 1):
    print(x, end="\t")
```

Kết quả sau khi chạy đoạn code trên:

Bảng 3.2: Danh sách khách hàng tiềm năng

26	84	115	160	210	259	302	304	318	339	364	371	381	393	447	544
594	631	673	709	825	837	862	868	888	941	988	1060	1074	1179	1223	1278
1311	1377	1379	1405	1414	1441	1494	1540	1567	1578	1592	1599	1614	1671	1678	1689
1770	1772	1783	1784	1863	1872	1889	1908	1928	1935	1939	1956	1957	1970	1990	1994
2017	2030	2109	2115	2122	2123	2148	2245	2280	2337	2428	2431	2433	2492	2493	2513
2520	2531	2582	2620	2692	2720	2742	2781	2784	2796	2851	2895	2897	2964	2994	3000
3065	3076	3104	3116	3123	3144	3159	3169	3214	3228	3270	3281	3354	3369	3392	3451
3488	3537	3539	3614	3681	3690	3711	3752	3761	3863	3917	3930	3934	3941	3945	3958
3974	3995	4057	4092	4111	4178	4208	4219	4231	4232	4270	4285	4290	4352	4355	4369
4380	4430	4459	4478	4491	4516	4538	4552	4566	4567	4607	4610	4628	4646	4732	4748
4760	4892	4946	5010	5013	5029	5037	5108	5129	5159	5169	5250	5266	5287	5324	5380
5382	5403	5416	5495	5519	5549	5573	5604	5686	5713	5733	5776	5791	5800	5808	5811
5843	5844	6049	6099	6100	6101	6128	6137	6145	6212	6241	6295	6380	6410	6412	6458
6489	6512	6574	6695	6714	6718	6778	6804	6814	6817	6912	6974	7029	7033	7070	7079
7091	7133	7150	7178	7227	7244	7253	7299	7373	7385	7413	7424	7436	7453	7455	7558
7704	7748	7825	7840	7868	7951	7967	8092	8114	8133	8144	8151	8195	8208	8210	8269
8340	8356	8378	8396	8465	8476	8533	8542	8543	8547	8551	8623	8660	8715	8730	8733
8742	8795	8832	8835	8843	8849	8858	8870	8876	8881	8895	8909	9033			

Kết quả đầu ra hiển thị chỉ mục (index) của tất cả các hàng là ứng cử viên có thể đăng ký tín dụng.

Để kiểm tra độ chính xác của mô hình, hãy sử dụng phương pháp cho điểm trên bộ phân loại như sau:

```
print('Độ chính xác của bộ phân loại hồi quy logistic trên bộ thử nghiệm: {:.2f} %'.format(classifier.score(X_test, Y_test)))
```

Độ chính xác của bộ phân loại hồi quy logistic trên bộ thử nghiệm: 0.89 %

Kết quả: Độ chính xác của bộ phân loại hồi quy logistic trên bộ thử nghiệm 0.89

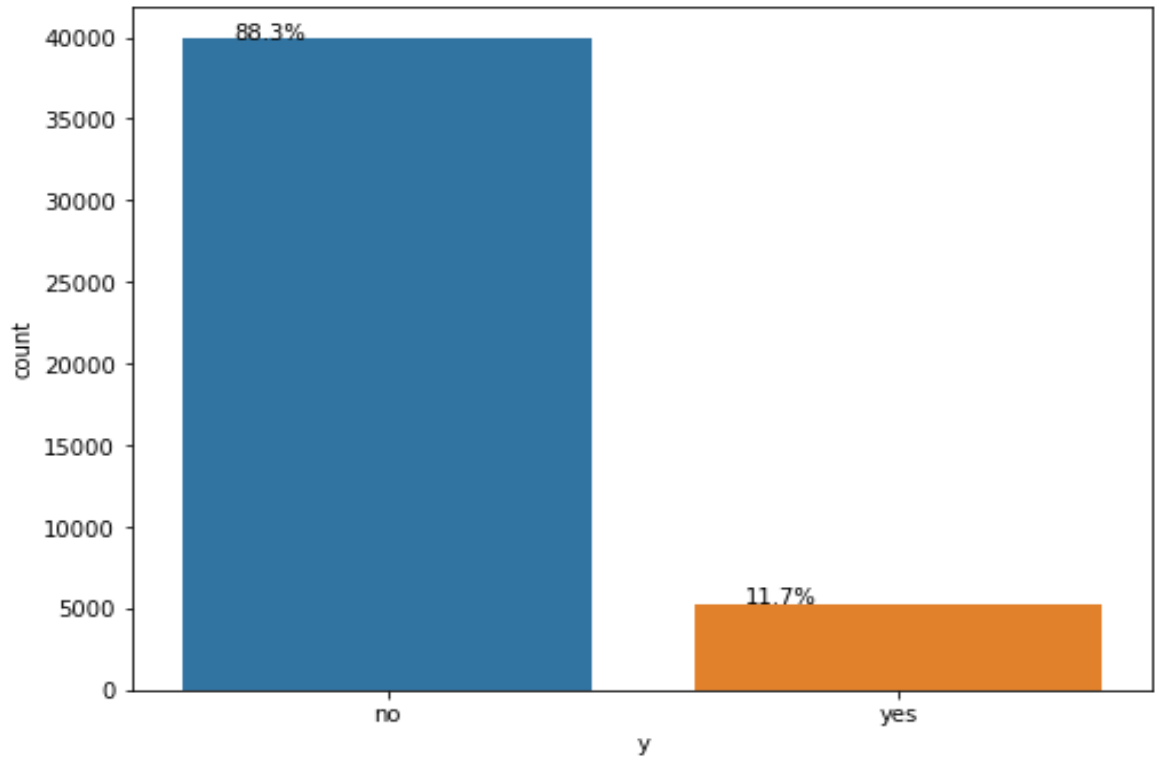
3.2 Mô hình Cây Quyết định (DT)

3.2.1. Các bước thực hiện mô hình

Sử dụng tập dữ liệu bank-full ở mô hình logistic ở trên

Bước 1: Chúng ta kiểm tra lại cho chắc chắn ở phần mô hình Cây quyết định này. Kiểm tra xem tập dữ liệu có cân bằng hay không? [8]

```
# Kiểm tra xem tập dữ liệu có cân bằng hay không
plt.figure(figsize = (8,6))
total = len(banktelemarket["y"])
ax = sns.countplot(x = 'y', data = banktelemarket)
for p in ax.patches:
    ax.annotate('{:.1f}%'.format(100*p.get_height()/total), (p.get_x()+0.1, p.get_height()+5))
plt.show()
```



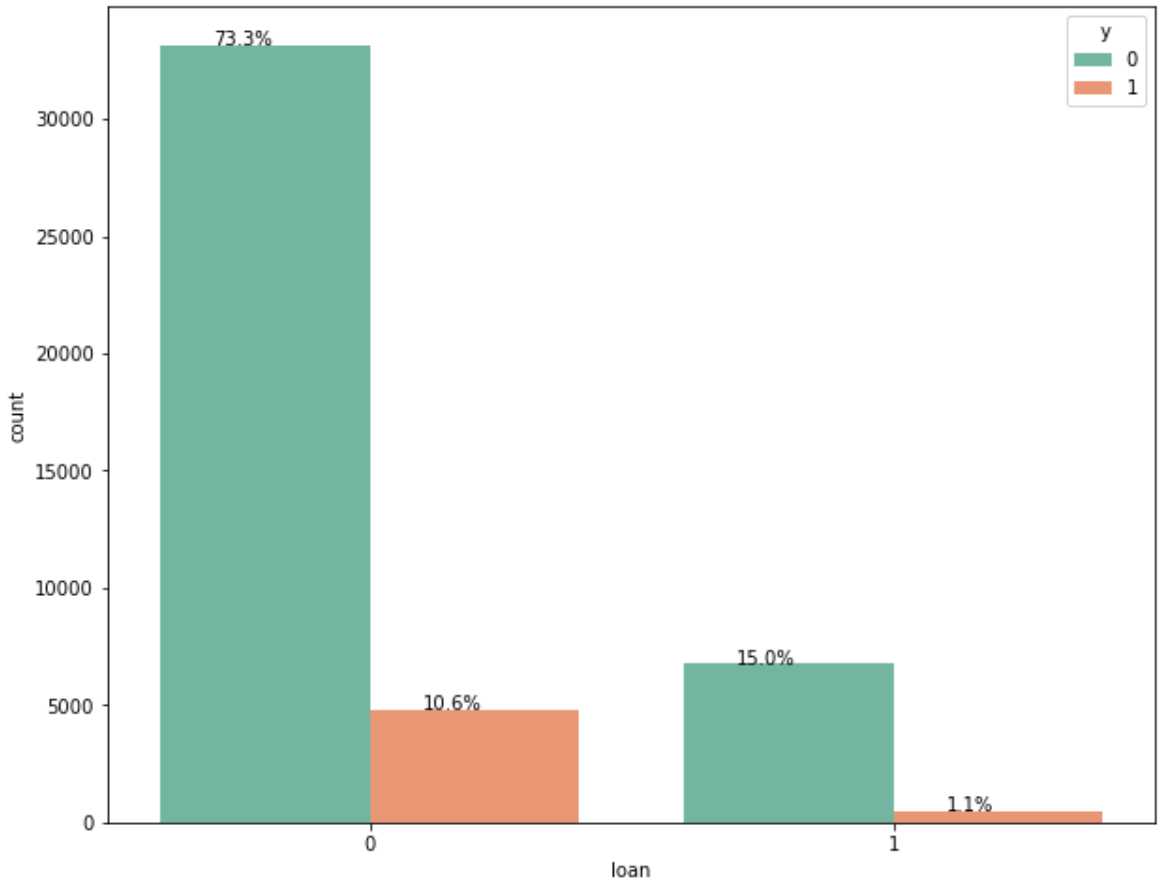
Hình 3.12 – Mô tả tập dữ liệu không cân bằng

Từ sơ đồ trên, chúng ta có thể nói rằng tập dữ liệu không cân bằng.

Bước 2: Phân tích đơn biến của các biến phân loại.

```
plt.figure(figsize = (10,8))
total = len(banktelemarket["loan"])
ax = sns.countplot(x = 'loan', data = banktelemarket, hue = 'y', palette = 'Set2')
```

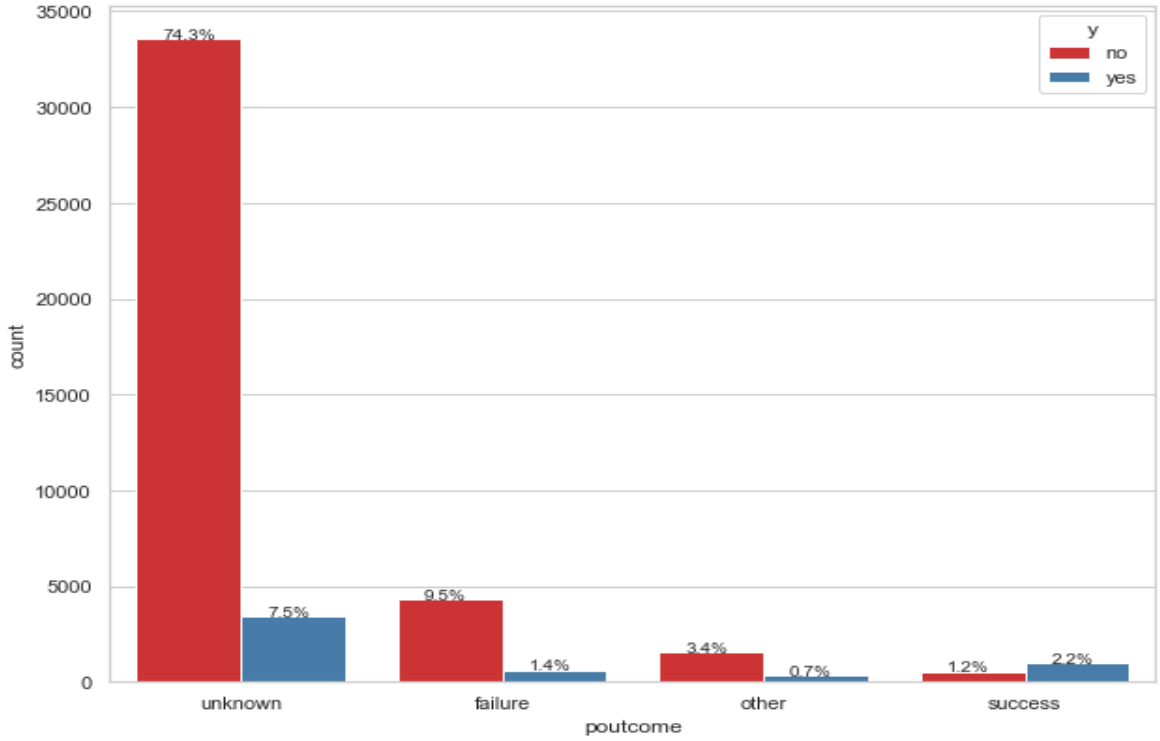
```
for p in ax.patches:
    ax.annotate('{:.1f}%'.format(100*p.get_height()/total),(p.get_x()+0.1,
p.get_height()+5))
plt.show()
```



Hình 3.13: Tỷ lệ khách hàng có vay

Số lượng người chưa vay nhiều hơn do đó người chưa vay có nhiều khả năng đăng ký tiền gửi có kỳ hạn hơn.

```
plt.figure(figsize = (10,8))
total = len(banktelemarket["poutcome"])
ax = sns.countplot(x = 'poutcome', data = banktelemarket, hue = 'y', palette =
'Set1')
for p in ax.patches:
    ax.annotate('{:.1f}%'.format(100*p.get_height()/total), (p.get_x()+0.1,
p.get_height()+5))
plt.show()
```

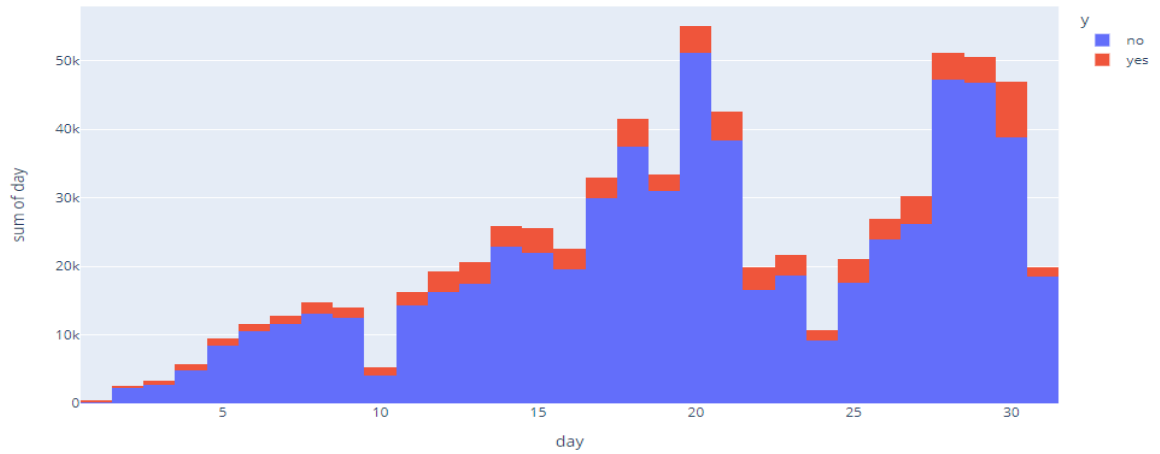


Hình 3.14: Tỷ lệ khách hàng của tiếp thị trước đó

Đa số những người đã đăng ký tiền gửi có kỳ hạn không có kết quả tiếp thị trước đó (*unknown*) có nghĩa là họ là khách hàng mới, chúng ta có thể đưa ra giả định rằng tính năng này có thể giữ một số giá trị trong việc dự đoán biến mục tiêu, đặc biệt là mục poutcome_success.

fig = px.histogram(banktelemarket, x = "day", y = "day", color = "y")

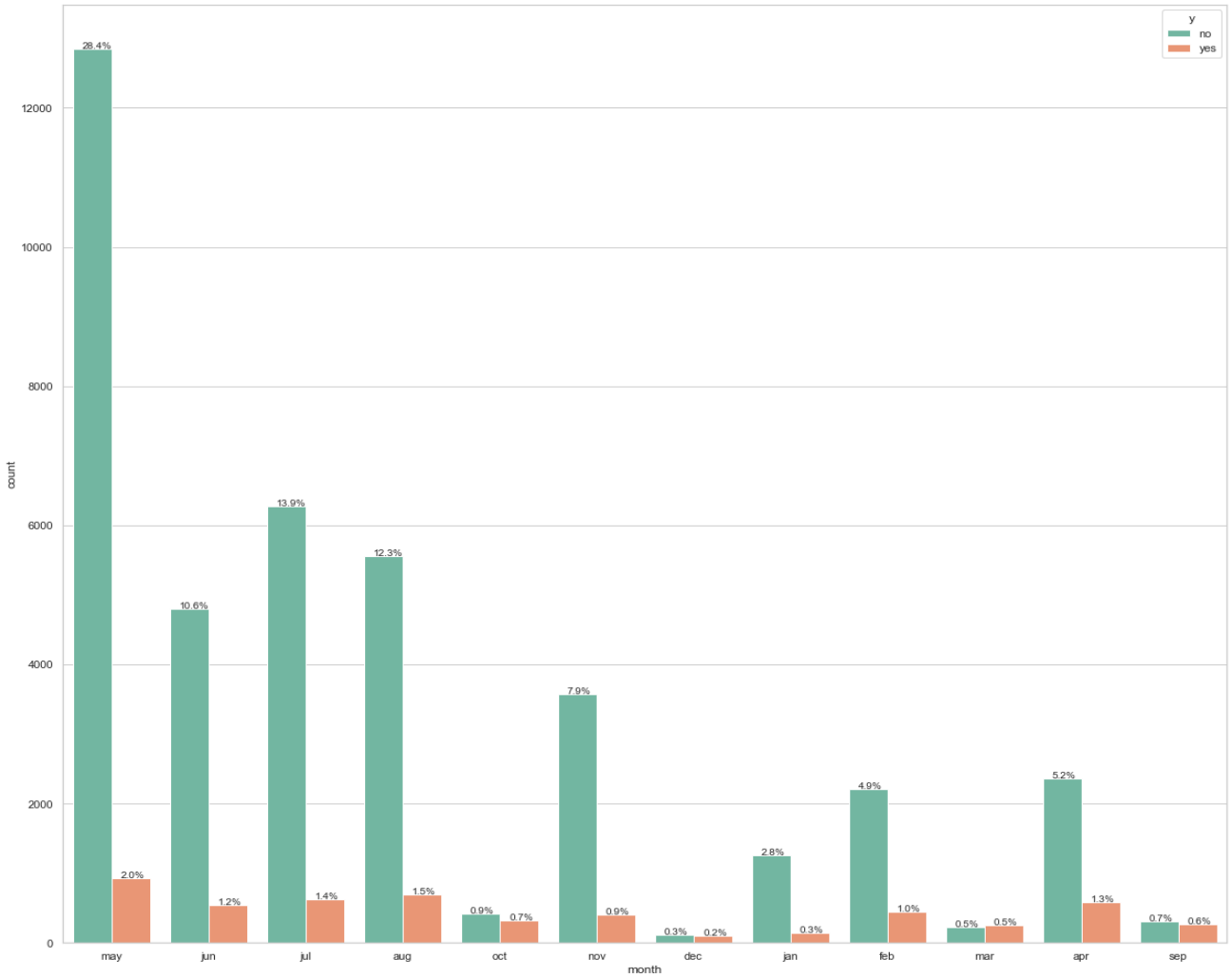
fig.show()



Hình 3.15: Tỷ lệ ngày

Từ biểu đồ trên, chúng ta có thể thấy rằng đó là một biểu đồ phân bố đồng đều do đó chúng ta có thể kết luận rằng đặc điểm này sẽ không hữu ích lắm trong việc dự đoán biến mục tiêu.

```
plt.figure(figsize = (20,18))
total = len(banktelemarket["month"])
ax = sns.countplot(x = 'month', data = banktelemarket, hue = 'y', palette = 'Set2')
for p in ax.patches:
   ax.annotate('{:.1f}%'.format(100*p.get_height()/total), (p.get_x()+0.1, p.get_height()+5))
plt.show()
```

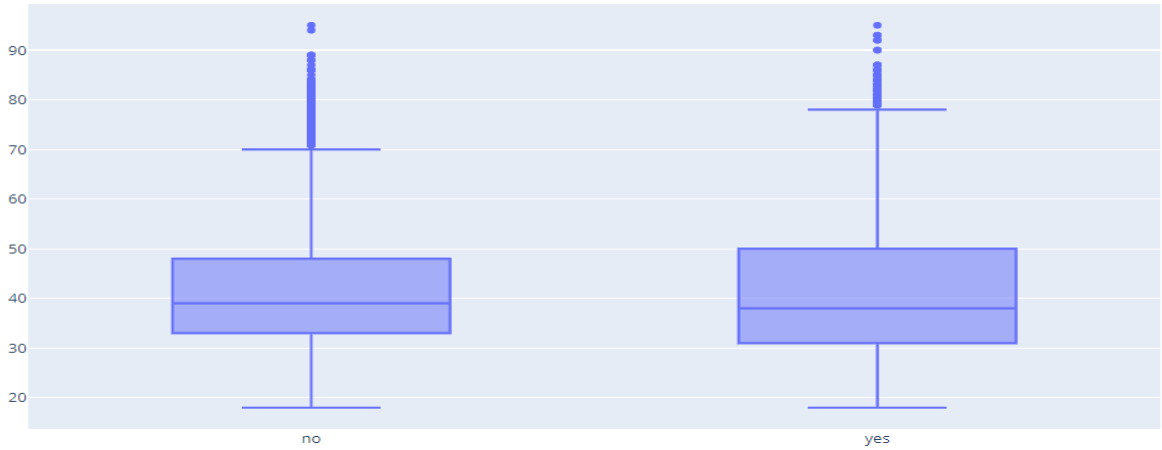


Hình 3.16: Tỷ lệ khách hàng có hoặc không đăng ký theo tháng

Trong tháng 5, tiếp theo là tháng 8, tháng 7 và tháng 4, số lượng người đăng ký gửi tiền có kỳ hạn nhiều hơn.

Phân tích giải pháp đơn biến trên các tính năng số

```
fig = px.box(x = banktelemarket["y"], y = banktelemarket["age"])
fig.show()
```

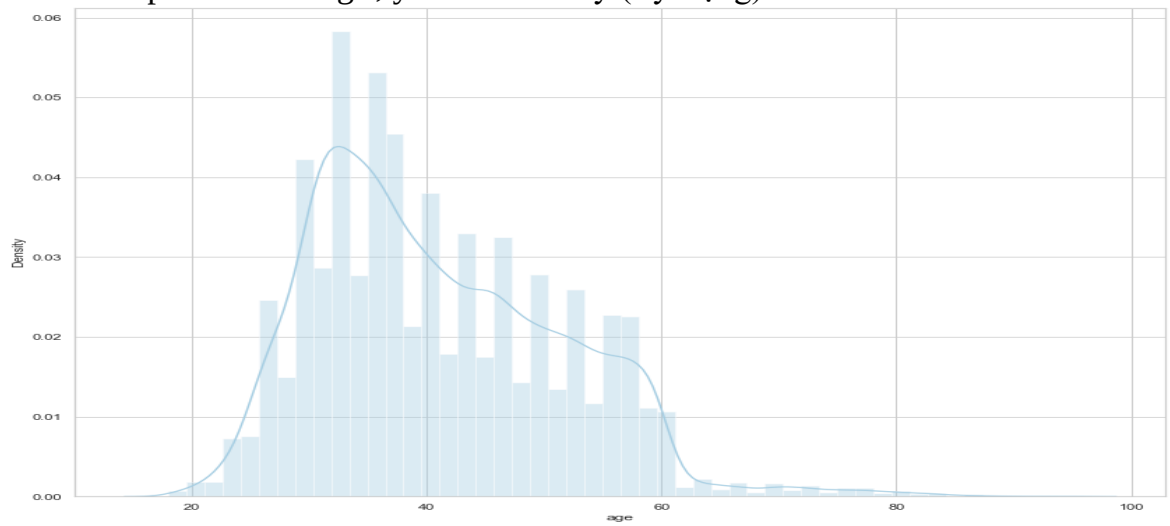


Hình 3.17: Tỷ lệ khách hàng có hoặc không đăng ký ở tuổi 38-39

Từ sơ đồ bên trên, chúng ta biết rằng đối với cả khách hàng đặt cọc hoặc không đăng ký tiền gửi có kỳ hạn, đều có độ tuổi trung bình khoảng 38–39. Và sơ đồ cho cả hai lớp trùng nhau khá nhiều, có nghĩa là độ tuổi không nhất thiết là tốt cho việc khách hàng nào sẽ đăng ký và khách hàng nào sẽ không.

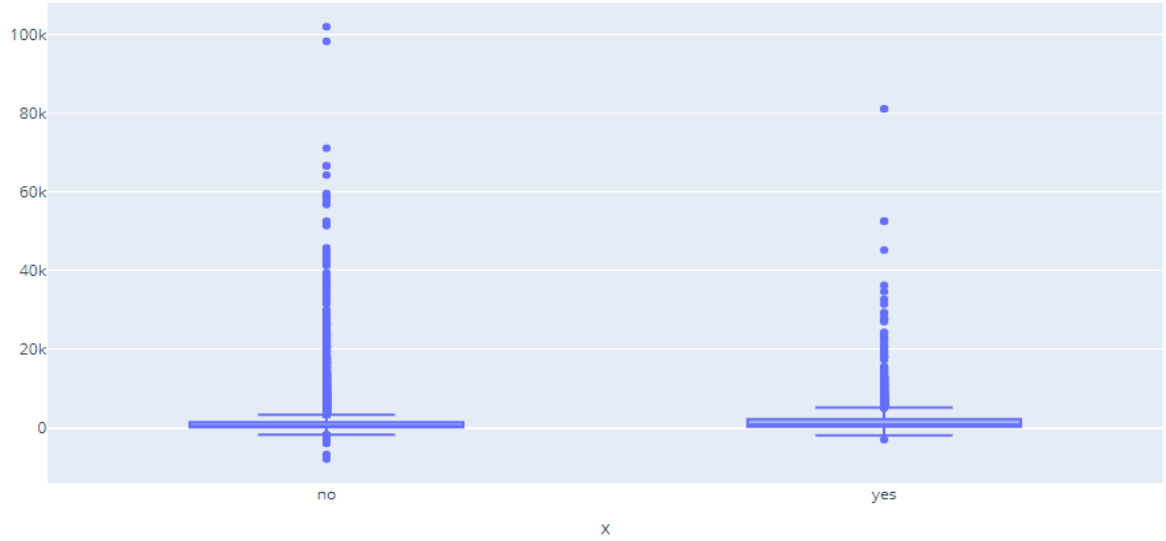
```
plt.figure(figsize = (14,12))
sns.distplot(banktelemarket["age"])
```

```
<AxesSubplot:xlabel='age', ylabel='Density (Tỷ trọng)'>
```



Hình 3.18: Biểu đồ tỷ trọng khách hàng có hoặc không đăng ký ở tuổi 38-39

```
fig = px.box(x = banktelemarket["y"], y = banktelemarket["balance"])
fig.show()
```

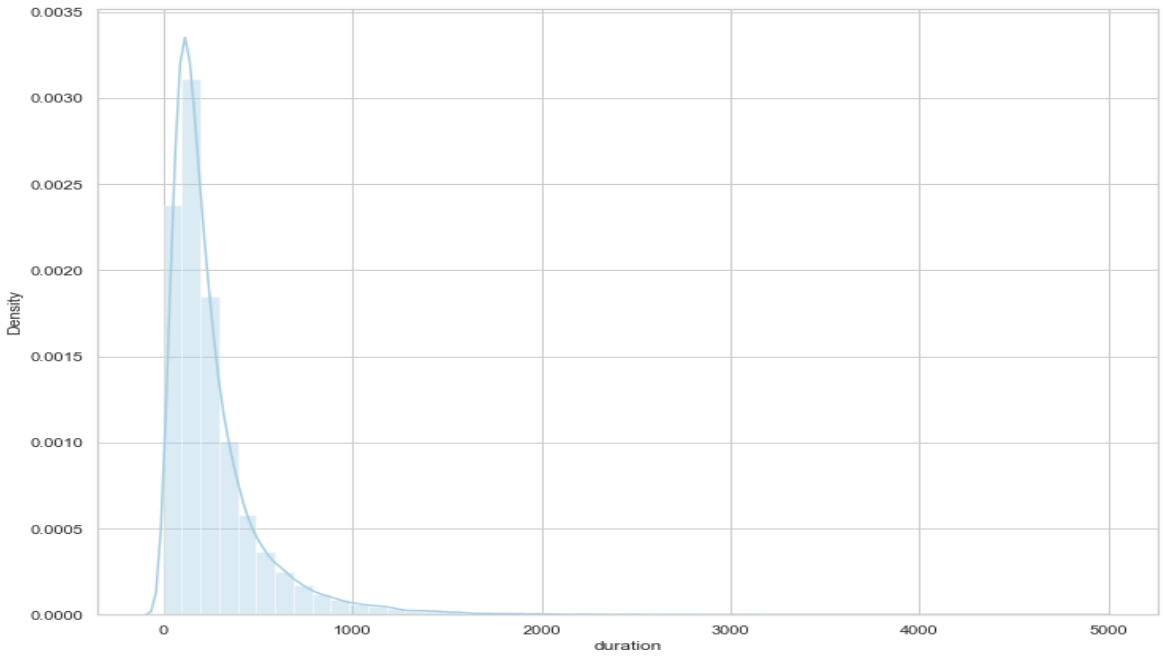


Hình 3.19**:** Biểu đồ tỷ trọng thuộc tính số dư của khách hàng

Tương tự như tính năng độ tuổi, ngay cả tính năng cân bằng cũng không đóng góp nhiều vào tính năng mục tiêu. Thuộc tính thời lượng ảnh hưởng nhiều đến thuộc tính mục tiêu (ví dụ: nếu thời lượng = 0 thì y = 'không'). Tuy nhiên, thời lượng không được biết trước khi thực hiện cuộc gọi. Ngoài ra, sau khi kết thúc cuộc gọi, biến đích y hiển nhiên đã được biết. Do đó, đầu vào này chỉ nên được đưa vào cho mục đích chuẩn và nên bị loại bỏ nếu mục đích là có một mô hình dự đoán thực tế.

Chúng ta biết rằng chúng ta sẽ không thể đưa tính năng này vào các mô hình cuối cùng của mình, vì rõ ràng là chúng ta muốn tạo ra một mô hình dự đoán thực tế có thể được sử dụng. Tuy nhiên, chúng ta chắc chắn sẽ triển khai một mô hình cơ bản với tính năng thời lượng chỉ để xem mức độ ảnh hưởng của tính năng này. Vì vậy, cùng với đó, chúng ta hãy xem xét sơ đồ hộp của tính năng này.

```
plt.figure(figsize = (14,12))
sns.distplot(banktelemarket["duration"])
<AxesSubplot:xlabel='duration', ylabel='Density (Tỷ trọng)'>
```

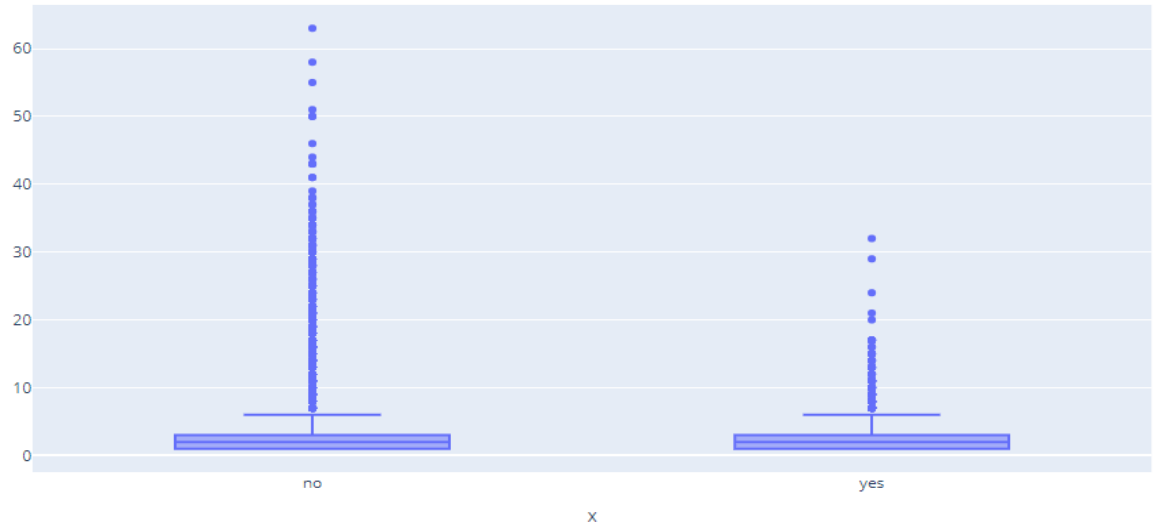


Hình 3.20: Biểu đồ tỷ trọng thuộc tính thời lượng của khách hàng

Từ biểu đồ trên, chúng ta có thể thấy rằng, thời lượng (thời gian tiếp xúc cuối cùng) của một khách hàng có thể hữu ích cho việc dự đoán biến mục tiêu. Nó được mong đợi bởi vì nó đã được đề cập trong tổng quan dữ liệu rằng trường này ảnh hưởng lớn đến biến mục tiêu và chỉ nên được sử dụng cho mục đích chuẩn.

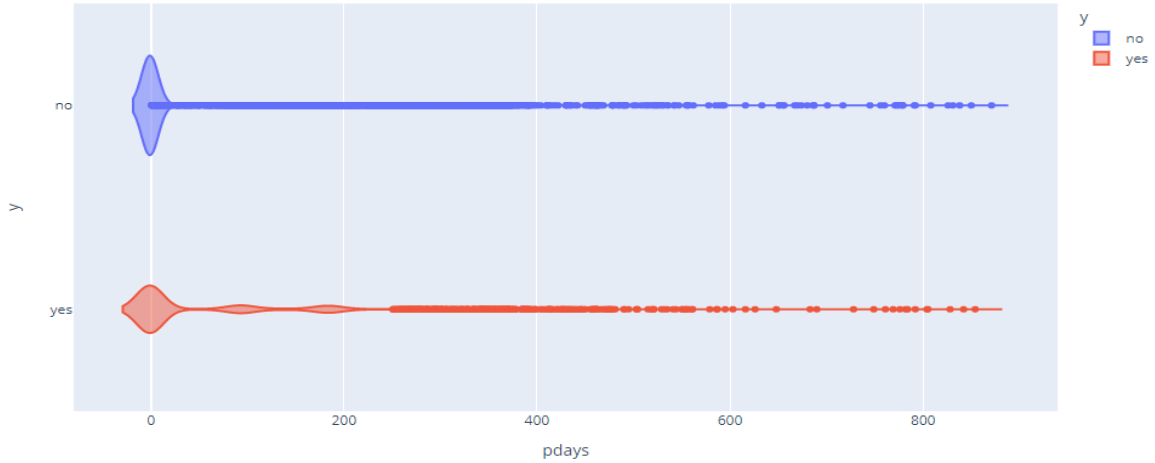
fig = px.box(x = banktelemarket["y"], y = banktelemarket["campaign"])

fig.show()



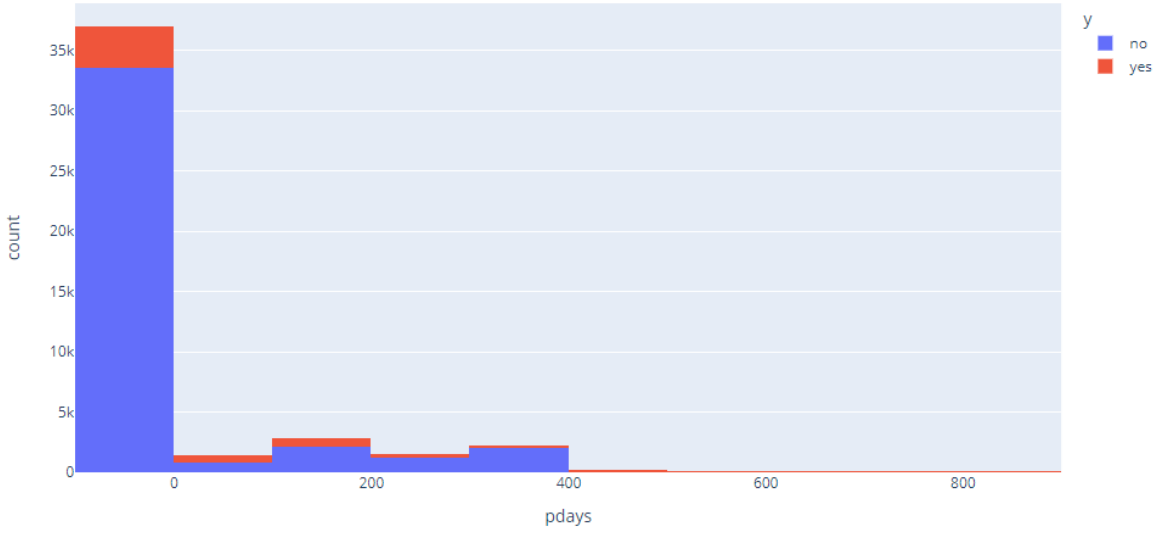
Hình 3.21: Biểu đồ tỷ trọng số lượng liên hệ của đợt

```
fig = px.violin(banktelemarket, x = banktelemarket["pdays"],
                y = banktelemarket["y"], color = banktelemarket["y"])
fig.show()
```



Hình 3.22: Biểu đồ tỷ trọng số lượng liên hệ của đợt

```
fig = px.histogram(banktelemarket, x="pdays",
                   color = banktelemarket["y"], nbins = 15)
fig.show()
```



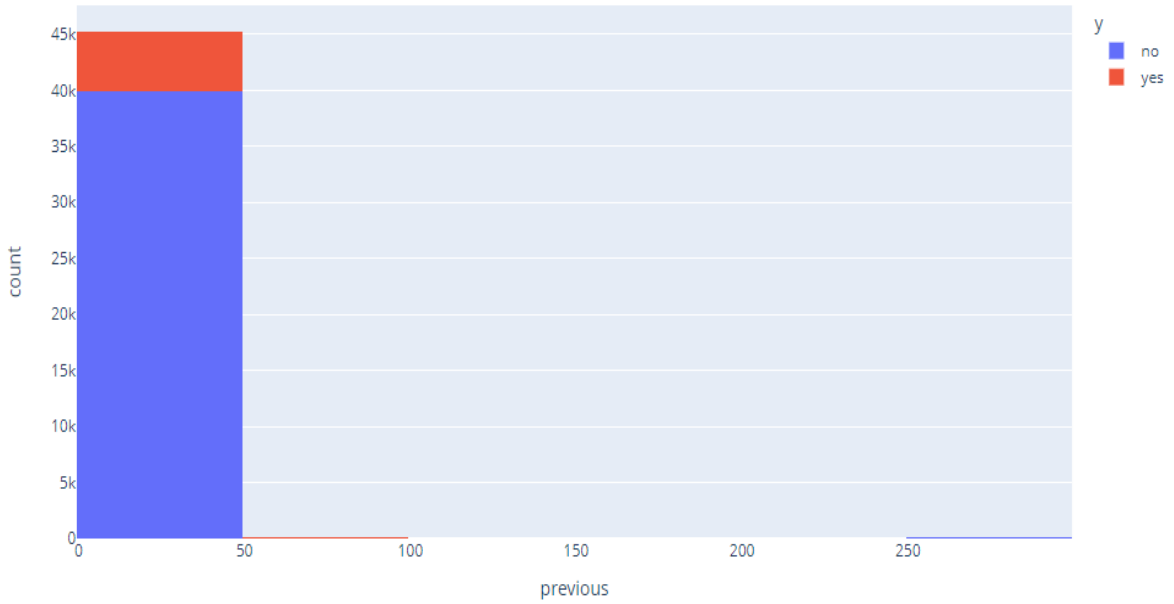
Hình 3.23: Số ngày trôi qua sau khi khách hàng được liên hệ lần cuối

Chúng ta có thể thấy rằng số ngày trôi qua sau khi khách hàng được liên hệ lần cuối từ một đợt trước đó (số .99 có nghĩa là khách hàng chưa được liên hệ trước đó) là từ 3 đến 6 tháng thì khả năng cao là khách hàng đăng ký tiền gửi có kỳ hạn.

khách hàng được liên hệ nhiều ngày trước hoặc hoàn toàn không liên lạc thì khách hàng đăng ký tiền gửi có kỳ hạn sẽ ít hơn.

fig = px.histogram(banktelemarket, x="previous", color = "y", nbins = 10)

fig.show()



Hình 3.24: Số lượng địa chỉ liên hệ được thực hiện trước đợt này

Số lượng liên hệ được thực hiện trước đợt này và cho khách hàng này hoặc tính năng trước đó không phải tốt về biến mục tiêu.

Bước 3: Tương quan các tính năng

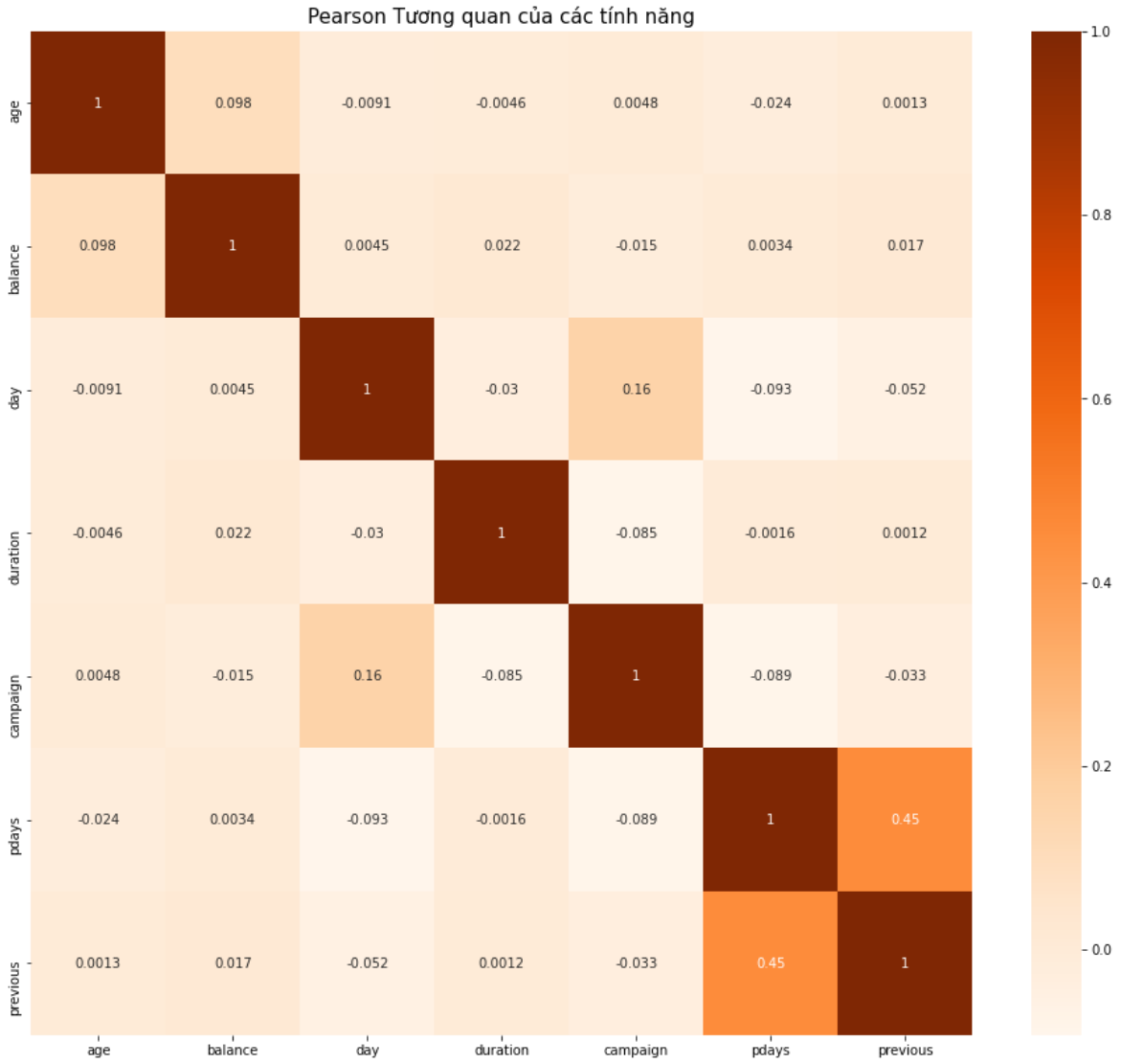
plt.figure(figsize = (16,14))

corr = banktelemarket.corr()

sns.heatmap(corr, annot = True, cmap = "Oranges")

plt.title("Pearson Tương quan của các tính năng", size = 15)

Text(0.5, 1.0, 'Pearson Tương quan của các tính năng')



Hình 3.25 – Mối tương quan đến biến mục tiêu

Pdays và tính năng trước đó có mức độ tương quan cao là 0,45. Kiểm tra xem có bất kỳ giá trị trùng lặp nào không.

```
banktelemarket[banktelemarket.duplicated()].sum()
age          0.0
job          0.0
marital      0.0
education    0.0
default      0.0
balance      0.0
housing      0.0
loan         0.0
contact      0.0
```

```
day          0.0
month        0.0
duration     0.0
campaign     0.0
pdays        0.0
previous     0.0
poutcome     0.0
y            0.0
dtype: float64
```

Bước 4: Xử lý trước dữ liệu

```
# Bỏ cột liên hệ vì nó không ảnh hưởng nhiều đến tính năng dự đoán mục tiêu
banktelemarket = banktelemarket.drop(columns = "contact", axis = 1)
banktelemarket.head()
from sklearn.preprocessing import LabelEncoder
labelEncoder = LabelEncoder()
banktelemarket['y'] = labelEncoder.fit_transform(banktelemarket['y'])
banktelemarket['marital'] =
labelEncoder.fit_transform(banktelemarket['marital'])
banktelemarket['education'] =
labelEncoder.fit_transform(banktelemarket['education'])
banktelemarket['default'] =
labelEncoder.fit_transform(banktelemarket['default'])
banktelemarket['housing'] =
labelEncoder.fit_transform(banktelemarket['housing'])
banktelemarket['loan'] = labelEncoder.fit_transform(banktelemarket['loan'])
banktelemarket['poutcome'] =
labelEncoder.fit_transform(banktelemarket['poutcome'])
banktelemarket["month"].value_counts().sort_values(ascending =
False).head(20)
may    13766
jul     6895
aug     6247
jun     5341
nov     3970
apr     2932
feb     2649
jan     1403
oct      738
sep      579
mar      477
dec      214
Name: month, dtype: int64
```

```
banktelemarket["job"].value_counts().sort_values(ascending = False).head(20)
blue-collar      9732
management       9458
technician       7597
admin.           5171
services         4154
retired          2264
self-employed    1579
entrepreneur     1487
unemployed       1303
housemaid        1240
student          938
unknown          288
Name: job, dtype: int64
# Nhận toàn bộ tập hợp các biến giả cho tất cả các cột phân loại
def one_hot_encode_top(df, col, top_6_labels):
    for label in top_6_labels:
        df[col+'_'+label] = np.where(df[col] == label,1,0)
banktelemarket_new = pd.read_csv('C:\\input\\bank-full.csv',
                                 sep=';', usecols = ['job','month'])
```

```
# Mã hóa cho cột công việc
top_6_labels = [x for x in banktelemarket["job"].value_counts().sort_values(ascending = False).head(6).index]
one_hot_encode_top(banktelemarket, "job", top_6_labels)
banktelemarket.head()
```

	age	job	marital	education	default	balance	housing	loan	day	month	duration	campaign	pdays	previous	poutcome	y	job_blue-collar
0	58	management	1	2	0	2143	1	0	5	may	261	1	-1	0	3	0	0
1	44	technician	2	1	0	29	1	0	5	may	151	1	-1	0	3	0	0
2	33	entrepreneur	1	1	0	2	1	1	5	may	76	1	-1	0	3	0	0
3	47	blue-collar	1	3	0	1506	1	0	5	may	92	1	-1	0	3	0	1
4	33	unknown	2	3	0	1	0	0	5	may	198	1	-1	0	3	0	0

```
# Mã hóa cho cột tháng
top_6_labels = [x for x in banktelemarket["month"].value_counts().sort_values(ascending = False).head(6).index]
one_hot_encode_top(banktelemarket, "month", top_6_labels)
banktelemarket.head()
```

	age	job	marital	education	default	balance	housing	loan	day	month	duration	campaign	pdays	previous	poutcome	y	job_blue-collar
0	58	management	1	2	0	2143	1	0	5	may	261	1	-1	0	3	0	0
1	44	technician	2	1	0	29	1	0	5	may	151	1	-1	0	3	0	0
2	33	entrepreneur	1	1	0	2	1	1	5	may	76	1	-1	0	3	0	0
3	47	blue-collar	1	3	0	1506	1	0	5	may	92	1	-1	0	3	0	1
4	33	unknown	2	3	0	1	0	0	5	may	198	1	-1	0	3	0	0

```
# Bỏ cột phân loại công việc ban đầu và tháng
banktelemarket.drop(columns = ["job", "month"], axis =1, inplace = True)
banktelemarket.head()
```

	age	marital	education	default	balance	housing	loan	day	duration	campaign	pdays	previous	poutcome	y	job_blue-collar
0	58	1	2	0	2143	1	0	5	261	1	-1	0	3	0	0
1	44	2	1	0	29	1	0	5	151	1	-1	0	3	0	0
2	33	1	1	0	2	1	1	5	76	1	-1	0	3	0	0
3	47	1	3	0	1506	1	0	5	92	1	-1	0	3	0	1
4	33	2	3	0	1	0	0	5	198	1	-1	0	3	0	0

```
banktelemarket.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 45211 entries, 0 to 45210
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   age               45211 non-null  int64
 1   marital           45211 non-null  int32
 2   education         45211 non-null  int32
 3   default           45211 non-null  int32
 4   balance           45211 non-null  int64
 5   housing           45211 non-null  int32
 6   loan              45211 non-null  int32
 7   day               45211 non-null  int64
 8   duration          45211 non-null  int64
 9   campaign          45211 non-null  int64
 10  pdays             45211 non-null  int64
 11  previous          45211 non-null  int64
 12  poutcome          45211 non-null  int32
 13  y                 45211 non-null  int32
 14  job_blue-collar   45211 non-null  int32
 15  job_management    45211 non-null  int32
 16  job_technician    45211 non-null  int32
 17  job_admin.        45211 non-null  int32
 18  job_services      45211 non-null  int32
 19  job_retired       45211 non-null  int32
 20  month_may         45211 non-null  int32
 21  month_jul         45211 non-null  int32
 22  month_aug         45211 non-null  int32
 23  month_jun         45211 non-null  int32
 24  month_nov         45211 non-null  int32
 25  month_apr         45211 non-null  int32
dtypes: int32(19), int64(7)
```

Tất cả các cột phân loại được chuyển đổi thành cột số

```
# Sắp xếp lại các cột
banktelemarket = banktelemarket[["age", "balance", "day", "campaign"
                                 , "duration", "pdays", "previous"
                                 , "marital", "education",
                "default", "housing", "loan", "poutcome", "job_management"
                                 , "job_blue-collar", "job_technician",
                "job_admin.", "job_services", "job_retired", "month_may"
                                 , "month_aug", "month_jul", "month_jun"
                                 , "month_nov", "month_apr", "y"]]
X = banktelemarket.iloc[:, :25]
y = banktelemarket.iloc[:, -1]
```

Tách tập dữ liệu thành huấn luyện và thử nghiệm

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
print("X_train: ", X_train.shape)
print("X_test: ", X_test.shape)
print("y_train: ", y_train.shape)
print("y_test: ", y_test.shape)
```

```
X_train:  (36168, 25)
X_test:  (9043, 25)
y_train:  (36168,)
y_test:  (9043,)
```

```
clf = DecisionTreeClassifier(random_state=0)
clf.fit(X_train, y_train)
y_pred_train = clf.predict(X_train)
y_pred_test = clf.predict(X_test)
acc_train = accuracy_score(y_train, y_pred_train)
acc_test = accuracy_score(y_test, y_pred_test)
print("Độ chính xác của dữ liệu đạo tạo: ", acc_train)
print("Độ chính xác của dữ liệu kiểm tra: ", acc_test)
```

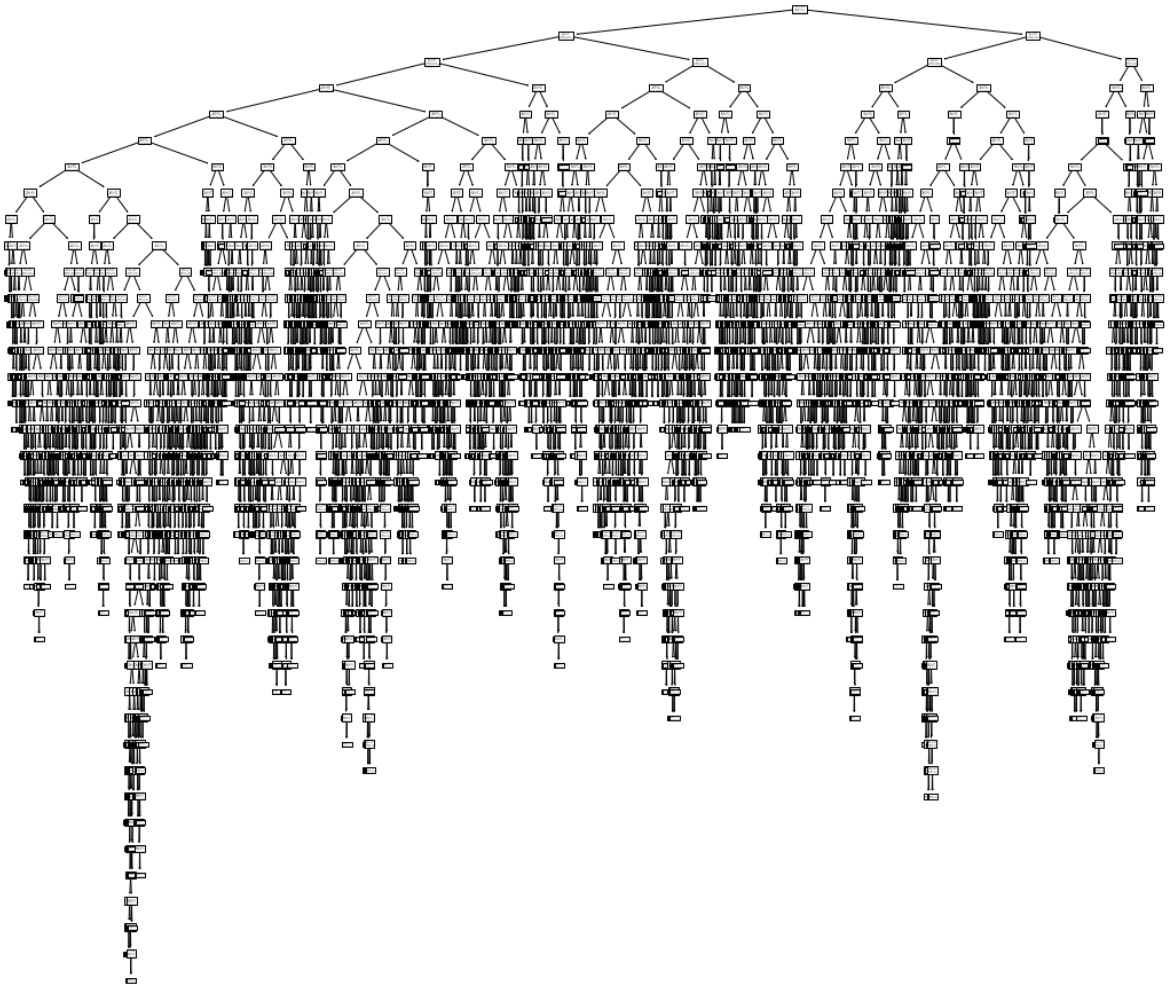
```
Độ chính xác của dữ liệu đạo tạo:  1.0
Độ chính xác của dữ liệu kiểm tra:  0.8673006745549043
```

Có thể thấy cây quyết định

```
plt.figure(figsize = (18, 16))

tree.plot_tree(clf)

plt.show()
```



Hình 3.26: Hình dạng cây quyết định

Hoạt động sau khi cắt tỉa

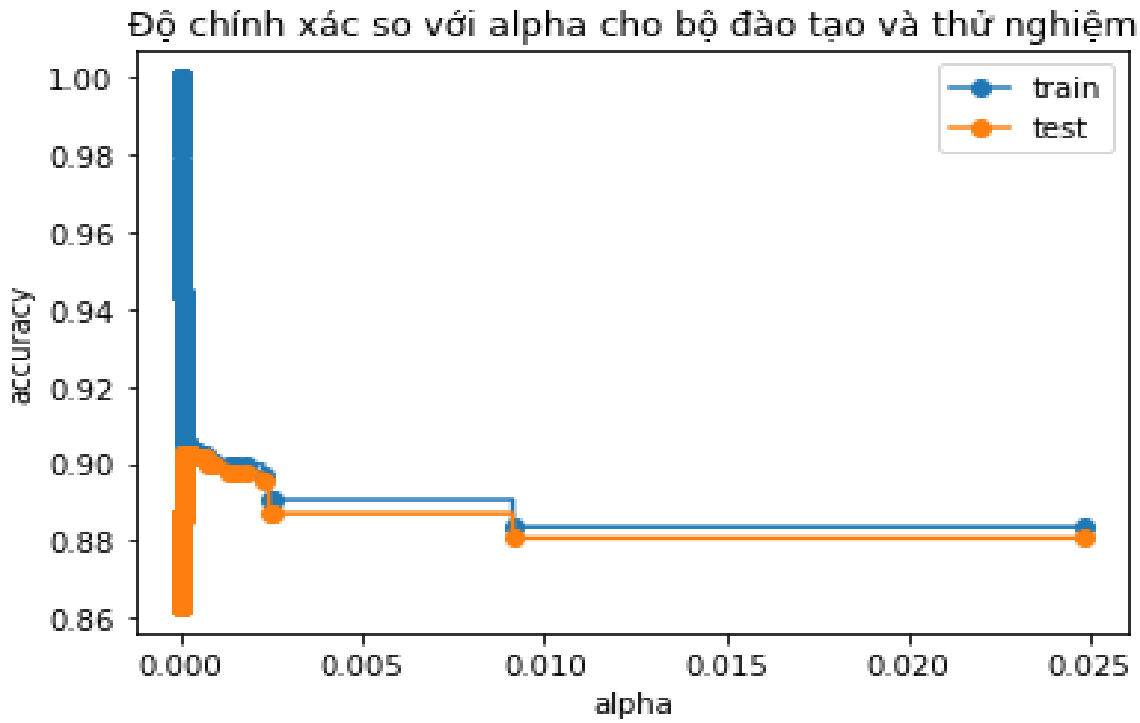
```
# Sử dụng kỹ thuật cost_complexity_pruning để cắt tỉa các nhánh của cây quyết định
path=clf.cost_complexity_pruning_path(X_train,y_train)
# biến đường dẫn cung cấp ccp_alphas và impurities
ccp_alphas,impurities=path.ccp_alphas,path.impurities
print("Giá trị alpha ccp :",ccp_alphas)
print()
print("Tạp chất cây quyết định :",impurities)
```

Giá trị alpha ccp : [0.00000000e+00 1.10502298e-05 1.10508868e-05 ... 2.93801166e-03
9.06406604e-03 2.57927731e-02]

Tạp chất cây quyết định : [0.00000000e+00 5.52511489e-05 1.10505583e-04 ... 1.62249556e-01
1.80377688e-01 2.06170461e-01]

```
# Lấy ccp_alphas làm một trong các tham số trong DecisionTreeClassifier ()
clfs=[]    #sẽ lưu trữ tất cả các mô hình tại đây
for ccp_alpha in ccp_alphas:
    clf=DecisionTreeClassifier(random_state=0,ccp_alpha=ccp_alpha)
    clf.fit(X_train,y_train)
    clfs.append(clf)
print("Nút cuối cùng trong cây Quyết định mới là {} và ccp_alpha cho nút cuối cùng là {}"
      .format(clfs[-1].tree_.node_count,ccp_alphas[-1]))
```

```
# Hình dung điểm độ chính xác cho tập huấn luyện và thử nghiệm.
train_scores = [clf.score(X_train, y_train) for clf in clfs]
test_scores = [clf.score(X_test, y_test) for clf in clfs]
fig, ax = plt.subplots()
ax.set_xlabel("alpha")
ax.set_ylabel("accuracy")
ax.set_title("Độ chính xác so với alpha cho bộ đào tạo và thử nghiệm")
ax.plot(ccp_alphas, train_scores, marker='o', label="train",drawstyle="steps-post")
ax.plot(ccp_alphas, test_scores, marker='o', label="test",drawstyle="steps-post")
ax.legend()
plt.show()
```

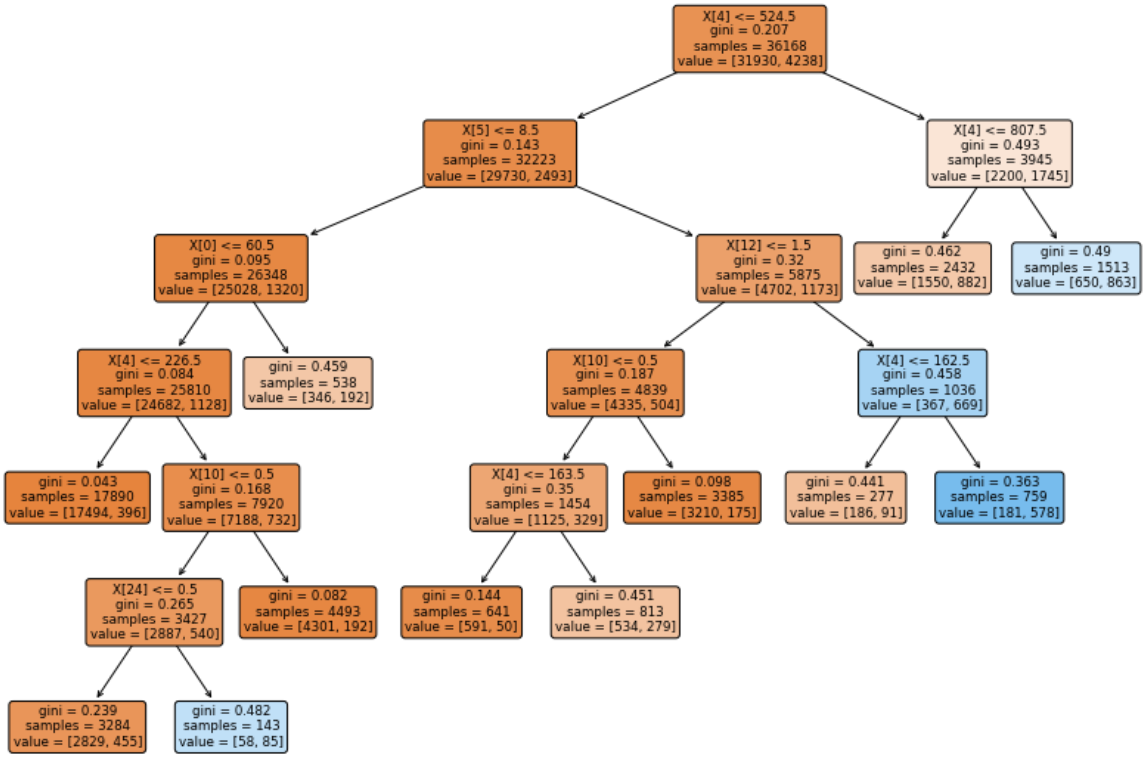


Hình 3.27: Độ chính xác so với alpha cho bộ đào tạo và thử nghiệm

Nếu chúng ta đánh đổi độ chệch và phương sai cân bằng, chúng ta sẽ chọn điểm đó sẽ có độ chệch thấp (sai số huấn luyện thấp) và phương sai thấp (sai số thử nghiệm thấp). Ở đây chúng ta lấy điểm đó ở giá trị alpha = 0,001.

3.2.2. Kết quả mô hình dự đoán

```
clf=DecisionTreeClassifier(random_state=0,ccp_alpha=0.001)
clf.fit(X_train,y_train)
plt.figure(figsize=(14,10))
tree.plot_tree(clf,rounded=True,filled=True)
plt.show()
```



Hình 3.28: Kết quả mô hình cây quyết định

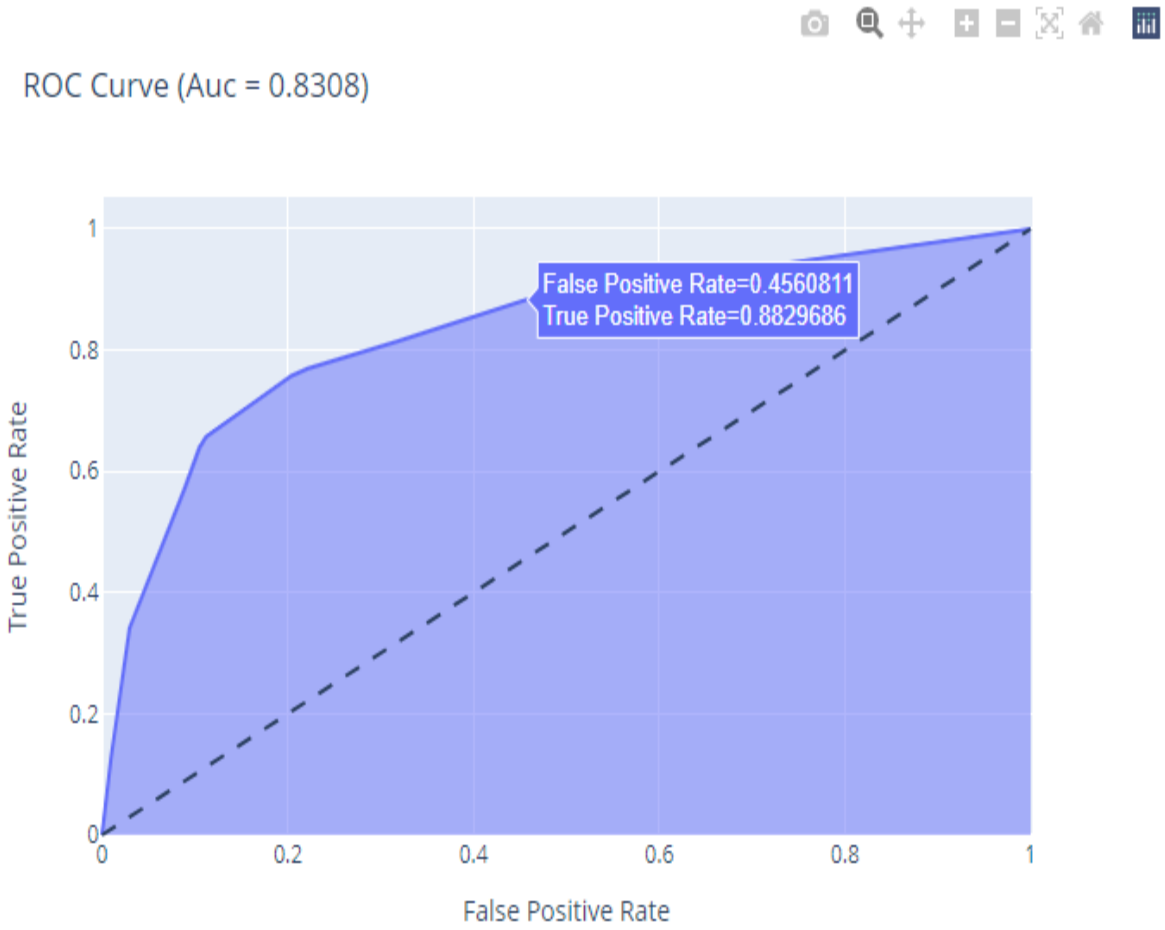
Ở đây chúng tôi có thể cắt tỉa cây phát triển vô tận. Hãy kiểm tra lại điểm độ chính xác

```
accuracy_score(y_test,clf.predict(X_test))
```

```
0.8994802609753401
```

```
from sklearn.metrics import roc_curve, auc
y_score = clf.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_score)
```

```
fig = px.area(
x=fpr, y=tpr,
title=f'ROC Curve (Auc = {auc(fpr, tpr):.4f})',
labels=dict(x = 'False Positive Rate', y = 'True Positive Rate'),
width = 700, height = 500)
fig.add_shape(
type = 'line', line = dict(dash='dash'),
x0=0, x1=1, y0=0, y1=1)
```

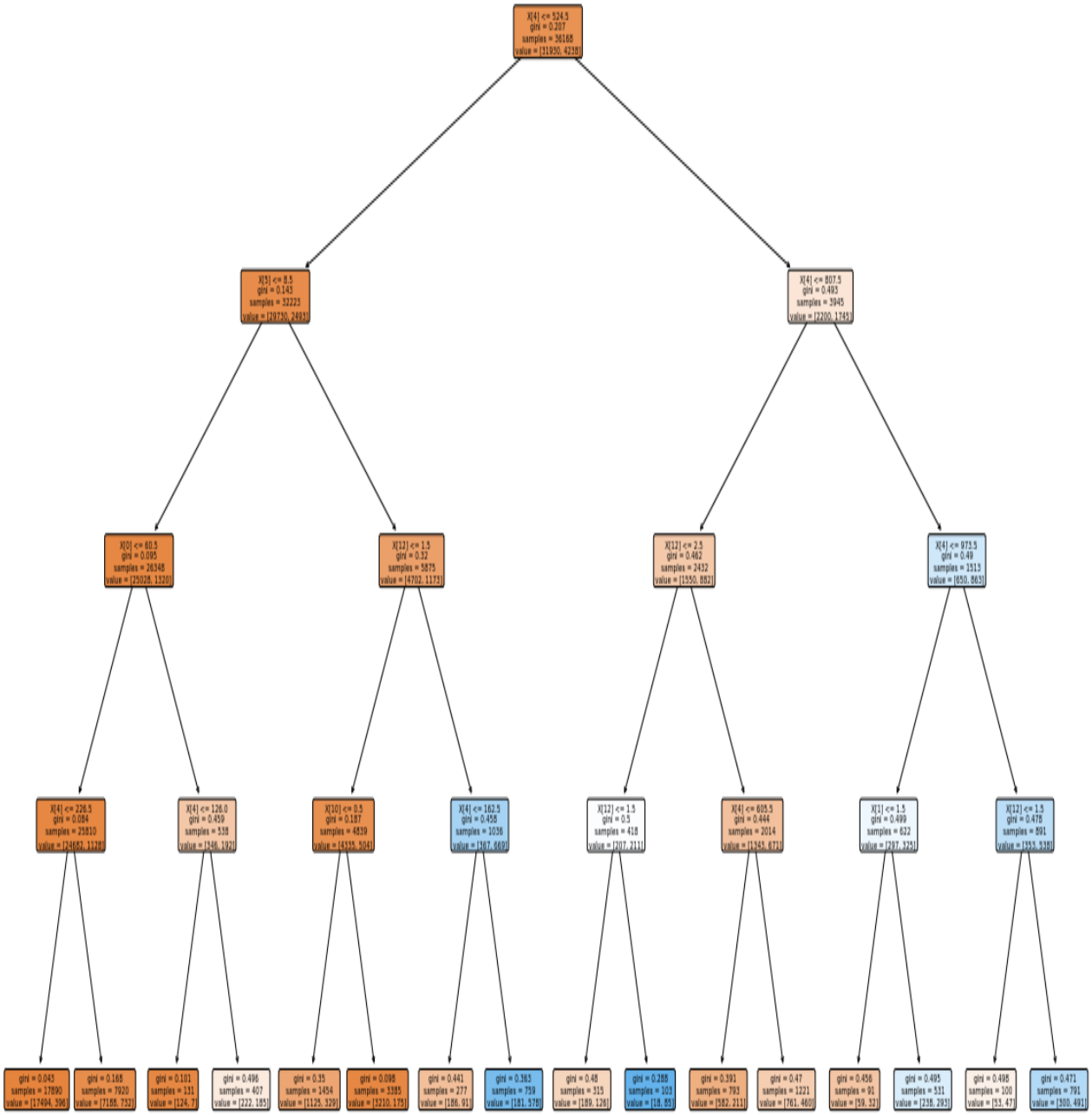


Hình 3.29: Kết quả mô hình độ chính xác cây quyết định

auc = 0,83 có nghĩa là có 83% cơ hội để mô hình của chúng tôi có thể phân biệt giữa lớp tích cực và lớp tiêu cực.

Cắt tỉa lại

```
# Sử dụng điều chỉnh siêu tham số
from sklearn.model_selection import GridSearchCV
#from sklearn.grid_search import GridSearchCV
grid_param={"criterion":["gini","entropy"],
            "splitter":["best","random"],
            "max_depth":range(5,15,1),
            "min_samples_leaf":range(5,15,1),
            "min_samples_split":range(5,15,1)
           }
grid_search=GridSearchCV(estimator=clf,param_grid=grid_param,cv=5,n_jobs=-1)
grid_search.fit(X_train,y_train)
GridSearchCV(cv=5,
             estimator=DecisionTreeClassifier(ccp_alpha=0.001, random_state=0),
             n_jobs=-1,
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': range(5, 15),
                         'min_samples_leaf': range(5, 15),
                         'min_samples_split': range(5, 15),
                         'splitter': ['best', 'random']})
print(grid_search.best_params_)
{'criterion': 'entropy', 'max_depth': 11, 'min_samples_leaf': 13, 'min_samples_split':
5, 'splitter': 'random'}
clf=DecisionTreeClassifier(criterion = 'gini', max_depth = 4,
                           min_samples_leaf = 1,
                           min_samples_split = 2,
                           splitter = 'best')
clf.fit(X_train,y_train)
plt.figure(figsize=(20,12))
tree.plot_tree(clf,rounded=True,filled=True)
plt.show()
```



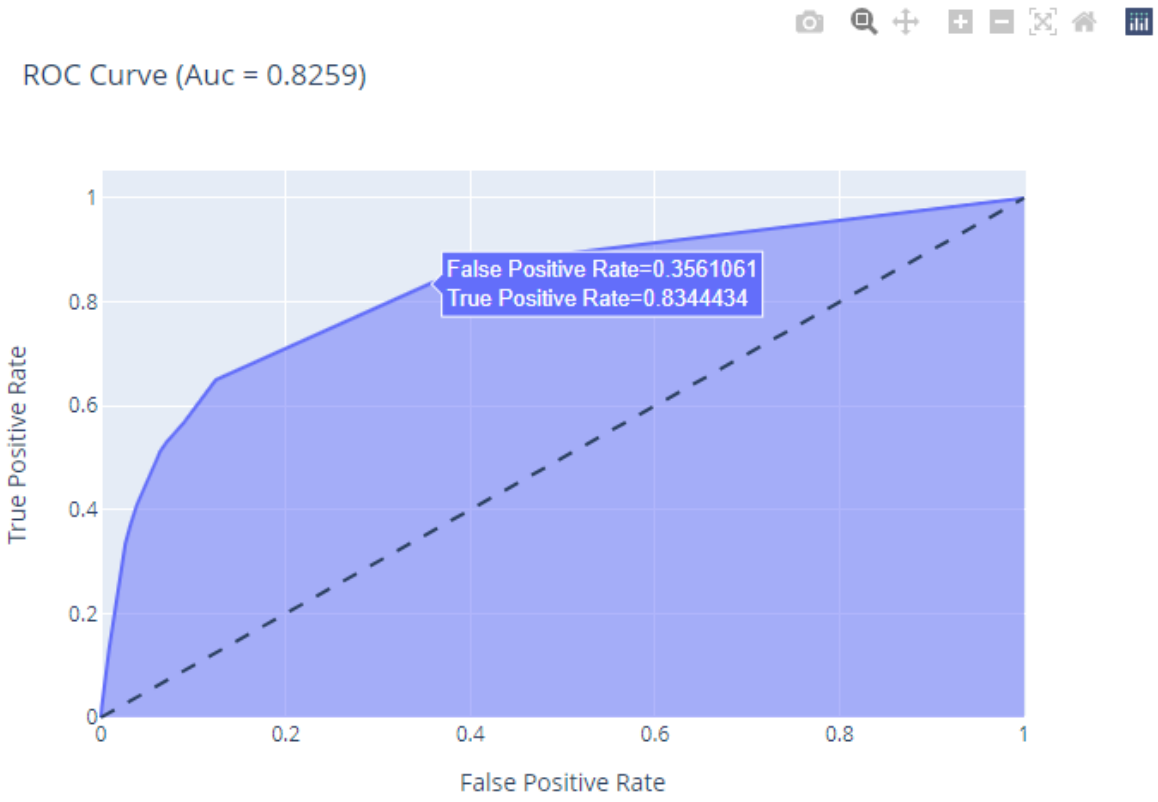
Hình 3.30: Kết quả mô hình cây quyết định sau khi cắt tỉa

```
y_pred = clf.predict(X_test)
accuracy_score(y_test,clf.predict(X_test))
```

```
0.8987061815769103
```

```
y_score = clf.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_score)

fig = px.area(
x=fpr, y=tpr,
title=f'ROC Curve (Auc = {auc(fpr, tpr):.4f})',
labels=dict(x = 'False Positive Rate', y = 'True Positive Rate'),
width = 700, height = 500)
fig.add_shape(
type = 'line', line = dict(dash='dash'),
x0=0, x1=1, y0=0, y1=1)
```



Hình 3.31: Kết quả mô hình độ chính xác cây quyết định 82%

auc = 0,82 có nghĩa là có 82% cơ hội để mô hình của chúng ta có thể phân biệt giữa lớp tích cực và lớp tiêu cực.

3.3. So sánh mô hình Hồi quy Logistic và Cây quyết định (DT)

Hồi quy logistic là một trong những kỹ thuật máy học được sử dụng nhiều nhất. Ưu điểm chính của nó là kết quả rõ ràng và khả năng giải thích mối quan hệ giữa các đối tượng phụ thuộc địa lý và độc lập một cách đơn giản. Tuy nhiên, nó cũng có một số nhược điểm chính là khả năng giải quyết các vấn đề phi tuyến tính còn hạn chế.

Cây quyết định thực hiện một nhiệm vụ rất giống nhau, chia nhỏ dữ liệu thành các nút để đạt được sự phân biệt tối đa giữa dương và âm, các nút của cây quyết định chọn nhiều tính năng cùng một lúc.

Biết rằng cây quyết định tốt trong việc xác định các mối quan hệ phi tuyến tính giữa các đối tượng phụ thuộc và độc lập, chúng ta có thể chuyển đổi đầu ra của cây quyết định (các nút) thành một biến phân loại và sau đó triển khai nó trong một hồi quy logistic, bằng cách chuyển đổi từng loại (các nút) thành các biến giả. [5]

Mô hình hồi quy logistic và cây quyết định trong mô hình trình bày là gần như tương đồng với nhau. Nhưng mô hình hồi quy logistic tốt hơn chút ít không chênh lệch nhiều so với mô hình cây quyết định.

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
clf = LogisticRegression(random_state=0)
clf.fit(X_train, Y_train)
y_pred_train = clf.predict(X_train)
y_pred_test = clf.predict(X_test)
acc_train = accuracy_score(Y_train, y_pred_train)
acc_test = accuracy_score(Y_test, y_pred_test)
print("Điểm chính xác của dữ liệu đào tạo: ", acc_train)
print("Điểm chính xác của dữ liệu kiểm tra: ", acc_test)
results = pd.DataFrame([['Hồi quy logistic', acc_test]],
                columns = ['Mô hình', 'Độ chính xác'])
```

```
Điểm chính xác của dữ liệu đào tạo:  0.8934693651846937
Điểm chính xác của dữ liệu kiểm tra:  0.8905230565077961
```

```
results
```

	Mô hình	Độ chính xác
0	Hồi quy logistic	0.890523

```
## Decision Tree
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion='entropy', random_state=0)
classifier.fit(X_train, y_train)

#Predicting the best set result
y_pred = clf.predict(X_test)
acc = accuracy_score(y_test, y_pred)

y_pred_ = clf.predict(X_train)
acc_train = accuracy_score(y_train, y_pred_)

accuracy_score(y_train,clf.predict(X_train))

results = pd.DataFrame([['Decision Tree (Cây Quyết Định)', acc]],
               columns = ['Mô hình', 'Độ chính xác'])
print("Điểm chính xác của dữ liệu kiểm tra: ", acc_train)
print("Điểm chính xác của dữ liệu kiểm tra: ", acc)
```

```
Điểm chính xác của dữ liệu kiểm tra:  0.8889073213890732
Điểm chính xác của dữ liệu kiểm tra:  0.8857679973460135
```

```
results
```

	Mô hình	Độ chính xác
0	Decision Tree (Cây Quyết Định)	0.885768

So sánh Mô hình Hồi quy Logistic và Cây quyết định (DT)

Bảng 3.2: So sánh 2 mô hình

Mô hình	Độ chính xác acc
Hồi quy Logistic	**0.890523**
Cây quyết định (DT)	**0.885768**

III. KẾT LUẬN

Trong ngành ngân hàng, tối ưu hóa nhằm mục tiêu cho tiếp thị qua điện thoại là một vấn đề then chốt, dưới áp lực ngày càng tăng nhằm tăng lợi nhuận và giảm chi phí. Với mỗi ngân hàng đều có những thế mạnh riêng và những sản phẩm mang tính truyền thống như cho vay và huy động tiền gửi thì mỗi ngân hàng đều có những chiến lược về sản phẩm riêng cũng như về thị trường và những chiến lược marketing cho những sản phẩm mang tính truyền thống để có sự khác biệt nhằm giữ chân khách hàng cũ và lôi kéo khách hàng mới. Do đó, hoạt động tiếp thị ngân hàng của em chỉ xoay quanh về sản phẩm tiền gửi đối với những khách hàng cũ và khách hàng mới đầy tiềm năng để tiếp thị.

Kết quả thử nghiệm của em trên tập dữ liệu bank-full.csv với 45.211 khách hàng dựa vào các thuộc tính có độ tương quan cao như công việc, tình trạng hôn nhân, nhà ở và các khoản vay của khách hàng để xây dựng 2 mô hình hồi quy logistic và cây quyết định dựa trên 80% dữ liệu đào tạo và 20% dữ liệu kiểm tra. Kết quả mô hình dự đoán khách hàng tiềm năng của 2 mô hình chính xác đến 89% và 88%. Đây là kết quả dự đoán rất khả quan đối với việc tiếp thị sản phẩm tiền gửi ngân hàng.

Độ chính xác của Mô hình Hồi quy Logistic là 89% và Cây Quyết định là 88% tức là gần 90% rất khả quan.

Mô hình hồi quy logistic và cây quyết định gần như tương đồng với nhau. Nhưng mô hình hồi quy logistic tốt hơn chút ít không chênh lệch nhiều so với mô hình cây quyết định.

Việc dùng mô hình dự đoán tập dữ liệu lớn với kết quả dự đoán của 2 mô hình trên giúp cho việc tiếp thị qua điện thoại cũng đỡ mất thời gian vì chỉ cần dựa vào kết quả dự đoán khách hàng tiềm năng để liên lạc qua điện thoại để tiếp thị sản phẩm tiền gửi có hiệu quả hơn.

Bên cạnh đó, kết quả nghiên cứu hoàn toàn có thể sử dụng được ở Việt Nam vì tất cả các trường dữ liệu của bộ dữ liệu đều là những tiêu chí để ngân hàng dựa vào đó để liên lạc với khách hàng.

Hồi quy logistic là một trong những kỹ thuật máy học được sử dụng nhiều nhất. Ưu điểm chính của nó là kết quả rõ ràng và khả năng giải thích mối quan hệ giữa các đối tượng phụ thuộc địa lý và độc lập một cách đơn giản. Tuy nhiên, nó cũng có một số nhược điểm chính là khả năng giải quyết các vấn đề phi tuyến tính còn hạn chế.

IV. DANH MỤC TÀI LIỆU THAM KHẢO

1. Moro, Sérgio, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. Decision Support Systems 62 (2014): 22-31. Available:https://www.academia.edu/6412064/A_Data_Driven_Approach_to_Predict_the_Success_of_Bank_Telemarketing.
2. http://bis.net.vn/forums/t/484.asp
3. https://ichi.pro/vi/tong-quan-ve-mo-hinh-cay-quyet-dinh-42316283748679
4. https://www.kaggle.com/nhunguyen1906/logistic-regression-bank-marketing
5. https://www.kaggle.com/madhuribh/bank-marketing-campaign-using-decision-tree
6. https://machinelearningcoban.com/2017/01/27/logisticregression
7. https://rpubs.com/alfandash/lbb-classification-2
8. https://vncoder.vn/bai-hoc/lay-du-lieu-443