BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH
KHOA CÔNG NGHỆ THÔNG TIN

TIỂU LUẬN MÔN HỌC

PHÂN TÍCH VÀ DỰ ĐOÁN HÀNH VI MUA SẮM CỦA KHÁCH HÀNG

Giảng viên hướng dẫn: ThS. SỬ NHẬT HẠ
Sinh viên thực hiện: CHU DOÃN ĐỨC
MSSV: 2000003917
Chuyên ngành: KHOA HỌC DỮ LIỆU
Môn học: CÔNG NGHỆ KHOA HỌC DỮ
Khóa: 2020

Tp.HCM, tháng 08 năm 2023

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH
KHOA CÔNG NGHỆ THÔNG TIN

TIỂU LUẬN MÔN HỌC

PHÂN TÍCH VÀ DỰ ĐOÁN HÀNH VI MUA SẮM CỦA KHÁCH HÀNG

Giảng viên hướng dẫn: ThS. SỬ NHẬT HẠ
Sinh viên thực hiện: CHU DOÃN ĐỨC
MSSV: 2000003917
Chuyên ngành: KHOA HỌC DỮ LIỆU
Môn học: CÔNG NGHỆ KHOA HỌC DỮ
Khóa: 2020

Tp.HCM, tháng 08 năm 2023

LỜI CẢM ƠN

Em xin bày tỏ lòng kính trọng và biết ơn các giảng viên ở bộ môn khoa công nghệ thông tin đã giúp đỡ và tạo điều kiện cho chúng em học hỏi nhiều điều mới trong hướng đi chuyên ngành mình đã chọn trong ngành khoa học dữ liệu. Nhờ sự cố gắng của các thầy, cô mà sinh viên chúng em có thể tự tin trang bị những kiến thức cần có để chuẩn bị cho kì thực tập của mình sắp tới.

Cảm ơn Thầy ThS. Sử Nhật Hạ đã tận tình hướng dạy trong môn Công nghệ Khoa Học Dữ Liệu để cho tụi em có cái nhìn tốt hơn về môn học này, cùng với nhiều bài tập và ví dụ thực tế cho ứng dụng của môn này trong cuộc sống. Nhờ đó mà em biết áp dụng môn học này có thể giải quyết một số bài toán thực tế bên ngoài như nào sau khi đi làm.

Ngoài ra cảm ơn các bạn nhóm khác đã cùng hỗ trợ mình khi gặp một số vấn đề khó giải quyết. Sản phẩm mình hoàn thành cũng có sự giúp đỡ của các bạn.

Sinh viên thực hiện

Chu Doãn Đức

TRƯỜNG ĐH NGUYỄN TẤT THÀNH KỲ THI KẾT THÚC HỌC PHẦN

TRUNG TÂM KHẢO THÍ **HỌC KỲ 3 NĂM HỌC 2022-2023**

PHIẾU CHẤM THI TIỂU LUẬN / ĐỒ ÁN

Môn thi: Công nghệ Khoa Học Dữ Liệu Lớp: 20DTH1D

Nhóm sinh viên thực hiện: 4

1. Nguyễn Hoàng Hiếu Tham gia đóng góp : 100%
2. Chu Doãn Đức Tham gia đóng góp : 100%
3. Trần Quốc Minh Tham gia đóng góp : 100%
4. Lê Đức Duy Tham gia đóng góp : 100%
5. Nguyễn Tiến Lợi Tham gia đóng góp : 100%

Ngày thi: 30/08/2023 Phòng thi: L.401

Đề tài tiểu luận / báo cáo của sinh viên: **Phân tích và Dự đoán hành vi mua sắm của khách hàng**

Phần đánh giá của giảng viên (căn cứ trên thang rubrics của môn học):

Tiêu chí (theo CĐR HP)	**Đánh giá của GV**	**Điểm tối đa**	**Điểm đạt được**
Cấu trúc báo cáo			
Nội dung			
Các nội dung thành phần			
Lập luận			
Kết luận			
Trình bày			
TỔNG ĐIỂM			

Giảng viên chấm thi

(Ký, ghi rõ họ và tên)

NHẬN XÉT GIÁO VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

TPHCM, Ngày …… tháng …… năm

Giáo viên nhận xét

(Ký, ghi rõ họ và tên)

MỤC LỤC

DANH MỤC HÌNH

LỜI MỞ ĐẦU

Bài tiểu luận này tập trung vào việc khai thác tiềm năng của bộ dữ liệu chứa hành vi mua sắm của khách hàng có tên là "Online Shoppers Intention" để dự đoán và phân tích hành vi mua sắm của khách hàng. Đặc biệt, chúng tôi quyết định sử dụng ba trong những thuật toán phân loại phổ biến nhất, bao gồm Decision Tree, Logistic Regression và Random Forest. Điều này giúp chúng tôi tạo ra một cái nhìn sâu hơn về cách mô hình hóa và dự đoán hành vi mua sắm có thể cung cấp thông tin hữu ích cho các doanh nghiệp trong việc tối ưu hóa chiến lược kinh doanh.

Với sự kết hợp giữa sự mạnh mẽ của học máy và quy trình phân tích dữ liệu, chúng ta có cơ hội tiếp cận những thông tin ẩn sau mỗi tương tác trực tuyến. Việc hiểu rõ yếu tố nào đang thúc đẩy hoặc ngăn cản sự hoàn thành của giao dịch có thể cung cấp cho các doanh nghiệp thông tin chiến lược cần thiết để cải thiện trải nghiệm mua sắm, tối ưu hóa quy trình bán hàng tăng cường sự tương tác.

Dự án này mở ra cơ hội tìm hiểu về cách áp dụng ba thuật toán quan trọng để dự đoán và phân tích hành vi mua sắm của khách hàng. Với 3 thuật toán trên là những công cụ mạnh mẽ trong việc phân loại và dự đoán dựa trên bộ dữ liệu. Bằng cách thực hiện sự kết hợp này, ta hy vọng có thể tìm ra các mô hình dự đoán chính xác nhất.

CHƯƠNG I: TỔNG QUAN ĐỀ TÀI

1. Giới thiệu

Trong thời kỳ mà thương mại điện tử đang trỗi dậy mạnh mẽ, việc hiểu rõ hành vi mua sắm của khách hàng trực tuyến không chỉ là chìa khóa thành công cho các doanh nghiệp, mà còn mang tính chất toàn cầu về việc cung cấp trải nghiệm mua sắm tốt nhất. Hành vi mua sắm phản ánh sự phức tạp của quyết định, ảnh hưởng cá nhân và yếu tố tâm lý trong quá trinhg khách hàng tham gia vào việc mua sắm qua nền tảng trực tuyến. Để khám phá những khía cạnh này, việc áp dụng các phương pháp dự đoán và phân tích trở nên vô cùng quan trọng.

2. Lý thuyết và nghiên cứu liên quan

Trong lĩnh vực này, nhiều nghiên cứu đã tập trung vào việc phân tích hành vi mua sắm và áp dụng các phương pháp dự đoán. Các tài liệu và nghiên cứu trước đây cung cấp một nền tảng quan trọng để tiếp cận đề tài này.

Trong một nghiên cứu của **Nguyen et al.(2019)**, các nhà nghiên cứu đã sử dụng dữ liệu từ một trang web thương mại điện tử để dự đoán hành vi mua sắm của khách hàng, Họ đã áp dụng các thuật toán học máy như Random Forest v à SVM để xây dựng mô hình dự đoán. Kết quả cho thấy khả năng dự đoán tương đối cao, đồng thời chỉ ra mối tương quan giữa các biến đầu vào và việc hoàn thành mua sắm.

Nghiên cứu **Li et al.(2020)** tập trung vào việc ứng dụng thuật toán Decision Tree để phân loại hành vi mua sắm của khách hàng. Họ đã xem xét tác động của yếu tố như thời gian truy cập, số lượng sản phẩm xem và loại trình duyệt. Kết quả cho thấy việc áp dụng Decision Tree có thể giúp phân loại hành vi mua sắm một cách chính xác.

Cũng trong cùng hướng nghiên cứu **Smith et al.(2021)** đã áp dụng Logistic Regression để dự đoán xác suất khách hàng thực hiện giao dịch. Họ đã phân tích cả yếu tố kỹ thuật (như trình duyệt và hệ điều hành) và yếu tố hành vi (như thời gian dành trong trang web) để xây dựng mô hình dự đoán.

3. Vấn đề cần giải quyết và giải pháp đề xuất

Mặc dù đã có nhiều nghiên cứu về hành vi mua sắm trực tuyến và áp dụng học máy trong lĩnh vực này, việc hiểu rõ hơn về cách áp dụng các thuật toán như Decision Treem Random Forest và Logistic Regression để dự đoán hành vi mua sắm vẫn còn nhiều thách thức. Chúng ta cần xác định cách tối ưu hóa việc chọn các biến đầu vào, xử lý dữ liệu, xây dựng mô hình dự đoán chính xác hơn. Đồng thời, v việc phân tích và giải thích mô hình cũng là một phần quan trọng để có thể áp dụng thông tin từ mô hình vào quyết định kinh doanh.

Với mục tiêu giải quyết những thách thức, chúng tôi đề xuất sử dụng bộ dữ liệu "online_shoppers_intention.csv" để thực hiện phân tích hành vi mua sắm của khách hàng. Chúng tôi sẽ áp dụng ba thuật toán Decision Tree, Random Forest và Logistic Regression để dự đoán khả năng hoàn thành mua sắm dựa trên các yếu tố liên quan. Đồng thời, chúng tôi cũng sẽ tiến hành phân tích mô hình để hiểu rõ hơn về sự tương quan giữa các biến và giải thích quyết định của mô hình.

CHƯƠNG II: CƠ SỞ LÝ THUYẾT

1. Hồi quy Logistic (Logistic Regression)

1.1. Hồi quy Logistic là gì ?

Hồi quy logistic là một kỹ thuật phân tích dữ liệu sử dụng toán học để tìm ra mối quan hệ giữa hai yếu tố dữ liệu. Sau đó, kỹ thuật này sử dụng mối quan hệ đã tìm được để dự đoán giá trị của những yếu tố đó dựa trên yếu tố còn lại. Dự đoán thường cho ra một số kết quả hữu hạn, như có hoặc không.

1.2. Sự quan trọng của hồi quy Logistic

Hồi quy logistic là một kỹ thuật quan trọng trong lĩnh vực trí tuệ nhân tạo và máy học (AI/ML). Mô hình ML là các chương trình phần mềm có thể được đào tạo để thực hiện các tác vụ xử lý dữ liệu phức tạp mà không cần sự can thiệp của con người. Mô hình ML được xây dựng bằng hồi quy logistic có thể giúp các tổ chức thu được thông tin chuyên sâu hữu ích từ dữ liệu kinh doanh của mình. Họ có thể sử dụng những thông tin chuyên sâu này để phân tích dự đoán nhằm giảm chi phí hoạt động, tăng độ hiệu quả và đổi chỉnh quy mô nhanh hơn.

Một số lợi ích của việc sử dụng hồi quy logistic so với các kỹ thuật ML khác :

- Tính đơn giản: Các mô hình hồi quy logistic ít phức tạp về mặt toán học hơn các phương pháp ML khác. Do đó, bạn có thể triển khai chúng ngay cả khi đội ngũ của bạn không ai có chuyên môn sâu về ML.
- Tốc độ: Các mô hình hồi quy logistic có thể xử lý khối lượng lớn dữ liệu ở tốc độ cao bởi chúng cần ít khả năng điện toán hơn, chẳng hạn như bộ nhớ và sức mạnh xử lý. Điều này khiến các mô hình hồi quy logistic trở nên lý tưởng đối với những tổ chức đang bắt đầu với các dự án ML để đạt được một số thành tựu nhanh chóng.
- Sự linh hoạt: Bạn có thể sử dụng hồi quy logistic để tìm đáp án cho các câu hỏi có hai hoặc nhiều kết quả hữu hạn. Bạn cũng có thể sử dụng phương pháp này để

xử lý trước dữ liệu. Ví dụ: bạn có thể sắp xếp dữ liệu với một phạm vi giá trị lớn, chẳng hạn như giao dịch ngân hàng, thành một phạm vi giá trị hữu hạn, nhỏ hơn nhờ hồi quy logistic. Sau đó, bạn có thể xử lý tập dữ liệu nhỏ hơn này với các kỹ thuật ML khác để phân tích chính xác hơn.

- Khả năng hiển thị: Phân tích hồi quy logistic cung cấp cho nhà phát triển khả năng nhìn nhận các quy trình phần mềm nội bộ rõ hơn so với các kỹ thuật phân tích dữ liệu khác. Khắc phục sự cố và sửa lỗi cũng trở nên dễ dàng hơn do các phép toán ít phức tạp hơn.

2. Cây quyết định (Decision Tree)

2.1. Cây quyết định (Decision Tree) là gì?

Cây quyết định (Decision Tree) là một cây phân cấp có cấu trúc được dùng để phân lớp các đối tượng dựa vào dãy các luật (series of rules). Khi cho dữ liệu về các đối tượng gồm các thuộc tính cùng với lớp (classes) của nó, cây quyết định sẽ sinh ra các luật để dự đoán lớp của các đối tượng chưa biết (unseen data).

Decision Trees gồm 3 phần chính: 1 node gốc (root node), những node lá (leaf nodes) và các nhánh của nó (branches). Node gốc là điểm bắt đầu của cây quyết định và cả hai node gốc và node chứa câu hỏi hoặc tiêu chí để được trả lời. Nhánh biểu diễn các kết quả của kiểm tra trên nút. Ví dụ câu hỏi ở node đầu tiên yêu cầu câu trả lời là “yes” hoặc là “no” thì sẽ có 1 node con chịu trách nhiệm cho phản hồi là “yes”, 1 node là “no”.

2.2. Xây dựng một cây quyết định

Có một vài thuật toán để tạo một cây quyết định, chúng ta sẽ nói về 2 trong số chúng:

1) CART (Classification and Regression Trees) → dùng Gini Index (Classification) để kiểm tra.
2) ID3 (Iterative Dichotomiser 3) → dùng Entropy function và Information gain để kiểm tra.

2.3. Ưu/Nhược điểm của cây quyết định Decision Tree

a. Ưu điểm

Cây quyết định dễ hiểu. Người ta có thể hiểu mô hình cây quyết định sau khi được giải thích ngắn.

Việc chuẩn bị dữ liệu cho một cây quyết định là cơ bản hoặc không cần thiết. Các kỹ thuật khác thường đòi hỏi chuẩn hóa dữ liệu, cần tạo các biến phụ (dummy variable) và loại bỏ các giá trị rỗng.

Cây quyết định có thể xử lý cả dữ liệu có giá trị bằng số và dữ liệu có giá trị là tên thể loại. Các kỹ thuật khác thường chuyên để phân tích các bộ dữ liệu chỉ gồm một loại biến. Chẳng hạn, các luật quan hệ chỉ có thể dùng cho các biến tên, trong khi mạng nơ-ron chỉ có thể dùng cho các biến có giá trị bằng số.

Cây quyết định là một mô hình hộp trắng. Nếu có thể quan sát một tình huống cho trước trong một mô hình, thì có thể dễ dàng giải thích điều kiện đó bằng logic Boolean. Mạng nơ-ron là một ví dụ về mô hình hộp đen, do lời giải thích cho kết quả quá phức tạp để có thể hiểu được.

Có thể thẩm định một mô hình bằng các kiểm tra thống kê. Điều này làm cho ta có thể tin tưởng vào mô hình.

Cây quyết định có thể xử lý tốt một lượng dữ liệu lớn trong thời gian ngắn. Có thể dùng máy tính cá nhân để phân tích các lượng dữ liệu lớn trong một thời gian đủ ngắn để cho phép các nhà chiến lược đưa ra quyết định dựa trên phân tích của cây quyết định.

b. Nhược điểm

Khó giải quyết được những vấn đề có dữ liệu phụ thuộc thời gian liên tục

Dễ xảy ra lỗi khi có quá nhiều lớp chi phí tính toán để xây dựng mô hình cây quyết định cao.

3. Rừng ngẫu nhiên (Random Forest)

3.1. Rừng ngẫu nhiên là gì?

Rừng ngẫu nhiên là một thuật toán học có giám sát. Như tên gọi của nó, Rừng ngẫu nhiên sử dụng các cây (tree) để làm nền tảng. Rừng ngẫu nhiên là một tập hợp của các Decision Tree, mà mỗi cây được chọn theo một thuật toán dựa vào ngẫu nhiên. Decision Tree là tên đại diện cho một nhóm thuật toán phát triển dựa trên Cây quyết định. Ở đó, mỗi Node của cây sẽ là các thuộc tính, và các nhánh là giá trị lựa chọn của thuộc tính đó. Bằng cách đi theo các giá trị thuộc tính trên cây, Cây quyết định sẽ cho ta biết giá trị dự đoán.Nhóm thuật toán cây quyết định có một điểm mạnh đó là có thể sử dụng cho cả bài toán Phân loại (Classification) và Hồi quy (Regression).

Random Forest algorithm có thể sử dụng cho cả bài toán Classification và RegressionRandom Forest làm việc được với dữ liệu thiếu giá trịKhi Forest có nhiều cây hơn, chúng ta có thể tránh được việc Overfitting với tập dữ liệuCó thể tạo mô hình cho các giá trị phân loại.

3.2. Xây dựng thuật toán Rừng ngẫu nhiên

Giả sử bộ dữ liệu của mình có n dữ liệu (sample) và mỗi dữ liệu có d thuộc tính (feature).

Để xây dựng mỗi cây quyết định mình sẽ làm như sau:

1) Lấy ngẫu nhiên n dữ liệu từ bộ dữ liệu với kĩ thuật Bootstrapping, hay còn gọi là random sampling with replacement. Tức khi mình sample được 1 dữ liệu thì mình không bỏ dữ liệu đấy ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục sample cho tới khi sample đủ n dữ liệu. Khi dùng kĩ thuật này thì tập n dữ liệu mới của mình có thể có những dữ liệu bị trùng nhau.
2) Sau khi sample được n dữ liệu từ bước 1 thì mình chọn ngẫu nhiên ở k thuộc tính ($k < n$). Giờ mình được bộ dữ liệu mới gồm n dữ liệu và mỗi dữ liệu có k thuộc tính.

3) Dùng thuật toán Decision Tree để xây dựng cây quyết định với bộ dữ liệu ở bước 2.

Do quá trình xây dựng mỗi cây quyết định đều có yếu tố ngẫu nhiên (random) nên kết quả là các cây quyết định trong thuật toán Random Forest có thể khác nhau.

Thuật toán Random Forest sẽ bao gồm nhiều cây quyết định, mỗi cây được xây dựng dùng thuật toán Decision Tree trên tập dữ liệu khác nhau và dùng tập thuộc tính khác nhau. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định.

Khi dùng thuật toán Random Forest, mình hay để ý các thuộc tính như: số lượng cây quyết định sẽ xây dựng, số lượng thuộc tính dùng để xây dựng cây. Ngoài ra, vẫn có các thuộc tính của thuật toán Decision Tree để xây dựng cây như độ sâu tối đa, số phần tử tối thiểu trong 1 node để có thể tách.

3.3. Tại sao thuật toán Rừng ngẫu nhiên lại tốt?

Trong thuật toán Decision Tree, khi xây dựng cây quyết định nếu để độ sâu tùy ý thì cây sẽ phân loại đúng hết các dữ liệu trong tập training dẫn đến mô hình có thể dự đoán tệ trên tập validation/test, khi đó mô hình bị overfitting, hay nói cách khác là mô hình có high variance.

Thuật toán Random Forest gồm nhiều cây quyết định, mỗi cây quyết định đều có những yếu tố ngẫu nhiên:

- Lấy ngẫu nhiên dữ liệu để xây dựng cây quyết định.
- Lấy ngẫu nhiên các thuộc tính để xây dựng cây quyết định.

Do mỗi cây quyết định trong thuật toán Random Forest không dùng tất cả dữ liệu training, cũng như không dùng tất cả các thuộc tính của dữ liệu để xây dựng cây nên mỗi cây có thể sẽ dự đoán không tốt, khi đó mỗi mô hình cây quyết định không bị overfitting mà có thể bị underfitting, hay nói cách khác là mô hình có high bias. Tuy nhiên, kết quả

cuối cùng của thuật toán Random Forest lại tổng hợp từ nhiều cây quyết định, thế nên thông tin từ các cây sẽ bổ sung thông tin cho nhau, dẫn đến mô hình có low bias và low variance, hay mô hình có kết quả dự đoán tốt.

CHƯƠNG III: MÔ HÌNH THỰC NGHIỆM

1. Tổng quan bộ dữ liệu

Bộ dữ liệu "online_shoppers_intention.csv" là một tập hợp dữ liệu về hành vi mua sắm trực tuyến của khách hàng trên một trang web thương mại điện tử. Bộ dữ liệu này chứa thông tin chi tiết về việc tương tác của khách hàng với trang web, cũng như các thông tin liên quan đến hành vi mua sắm.

Dữ liệu bào gồm nhiều biến quan trọng như:

1) Administrative: Số lượng trang quản trị viên (administrative) được truy cập bởi người dùng trước khi thực hiện mua hàng.
2) Administrative_Duration: Thời gian trung bình mà người dùng đã truy cập vào các trang quản trị viên trước khi thực hiện mua hàng.
3) Informational: Số lượng trang thông tin (informational) được truy cập bởi người dùng trước khi thực hiện mua hàng.
4) Informational_Duration: Thời gian trung bình mà người dùng đã truy cập vào các trang thông tin trước khi thực hiện mua hàng.
5) ProductRelated: Số lượng trang liên quan đến sản phẩm (product-related) được truy cập bởi người dùng trước khi thực hiện mua hàng.
6) ProductRelated_Duration: Thời gian trung bình mà người dùng đã truy cập vào các trang liên quan đến sản phẩm trước khi thực hiện mua hàng.
7) BounceRates: Tỷ lệ người dùng rời khỏi trang web sau khi chỉ xem một trang duy nhất (không tương tác thêm).
8) ExitRates: Tỷ lệ người dùng rời khỏi trang web sau khi xem trang hiện tại (bao gồm cả trang hiện tại và các trang trước đó).

9) PageValues: Giá trị trung bình của các trang được truy cập bởi người dùng, tính theo giá trị cuối cùng (giá trị mục tiêu) của trang.
10) SpecialDay: Chỉ số ngày đặc biệt gần ngày mua hàng (ví dụ: ngày lễ, sự kiện đặc biệt).
11) Month: Tháng trong năm khi người dùng thực hiện mua hàng.
12) OperatingSystems: Hệ điều hành được sử dụng bởi người dùng.
13) Browser: Trình duyệt web được sử dụng bởi người dùng.
14) Region: Vùng địa lý của người dùng.
15) TrafficType: Loại lưu lượng truy cập (traffic type) được sử dụng bởi người dùng.
16) VisitorType: Loại khách truy cập (visitor type) của người dùng (New Visitor, Returning Visitor, Other).
17) Weekend: Biến nhị phân (0 hoặc 1) cho biết liệu ngày mua hàng có rơi vào cuối tuần hay không.
18) Revenue: Biến nhị phân (0 hoặc 1) cho biết liệu người dùng đã thực hiện mua hàng hay không (0: Không mua hàng, 1: Mua hàng).

Bộ dữ liệu có thể được lấy từ nhiều nguồn khác nhau, nhưng một trong những nguồn phổ biến và đáng tin cậy là từ các kho lưu trữ dữ liệu trực tuyến và các trang web chia sẻ dữ liệu liên quan đến học máy và phân tích dữ liệu. Các trang web như Kaggle, UCI Machine Learning Repository và Data.gov cung cấp nhiều bộ dữ liệu cho cộng đồng nghiên cứu sử dụng.

2. Thực nghiệm

2.1. Môi trường thực thi

Môi trường thực thi Google Colab, là một nền tảng học máy và phân tích đữ liệu trực tuyến được cung cấp bởi Google. Được xây dựng trên nền tảng Google Drive, Google Colab cung cấp môi trường phát triển học máy mạnh mẽ và tiện lợi cho các nhà nghiên cứu, khoa học đữ liệu và những người quan tâm đến việc làm việc với dữ liệu lớn và thuật toán học máy phức tạp.

Một trong những ưu điểm nổi bật của Google Colab là tích hợp sẵn với Jupyter Notebook, cho phép tạo và chia sẻ các tệp notebook có chứa mã, văn bản, biểu đồ và hình ảnh. Có thể viết mã và chạy mã Python trực tiếp trên trình duyệt mà không cần cài đặt bất kỳ phần mềm nào trên máy tính cá nhân. Điều này giúp tiết kiệm thời gian và loại bỏ rào cản trong việc trải nghiệm và học tập về học máy.

Môi trường Google Colab cung cấp môi trường phát triển Python hoàn chỉnh, tích hợp sẵn các thư viện phổ biến như Numpy, Pandas, Matplotlib, Sklearn,…. Có thể sử dụng các thư viện này để thực hiện phân tích dữ liệu, xây dựng và đào tạo mô hình học máy một cách hiệu quả.

2.2. Import thư viện và Load dữ liệu

Nhập các thư viện và module cần thiết cho quá trình thực hiện dự đoán và phân tích

```
# Import các thư viện và module cần thiết
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import pickle

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,f1_score,recall_score,precision_score,classification_report, confusion_matrix

from sklearn.preprocessing import LabelEncoder
from imblearn.over_sampling import SMOTE
from sklearn.preprocessing import MinMaxScaler

warnings.filterwarnings("ignore")
plt.style.use('ggplot')
```

Thực hiện đọc dữ liệu từ Google Drive và in ra 5 dòng đầu tiên của bộ dữ liệu :

```
# Đọc dữ liệu từ csv
df = pd.read_csv('/drive/My Drive/CongNgheKHDL/PredictingOnlineShopperIntention/data/online_shoppers_intention.csv')
df.head()
```

	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration
0	0	0.0	0	0.0	1	0.000000
1	0	0.0	0	0.0	2	64.000000
2	0	0.0	0	0.0	1	0.000000
3	0	0.0	0	0.0	2	2.666667
4	0	0.0	0	0.0	10	627.500000

Kiểm tra thông tin tập dữ liệu tổng quan gồm những cột gì và kiểu dữ liệu tương ứng từng cột. Trong đó có 4 kiểu dữ liệu bao gồm: int64, float64, bool, object.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12330 entries, 0 to 12329
Data columns (total 18 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Administrative           12330 non-null  int64
 1   Administrative_Duration  12330 non-null  float64
 2   Informational            12330 non-null  int64
 3   Informational_Duration   12330 non-null  float64
 4   ProductRelated           12330 non-null  int64
 5   ProductRelated_Duration  12330 non-null  float64
 6   BounceRates              12330 non-null  float64
 7   ExitRates                12330 non-null  float64
 8   PageValues               12330 non-null  float64
 9   SpecialDay               12330 non-null  float64
 10  Month                    12330 non-null  object
 11  OperatingSystems         12330 non-null  int64
 12  Browser                  12330 non-null  int64
 13  Region                   12330 non-null  int64
 14  TrafficType              12330 non-null  int64
 15  VisitorType              12330 non-null  object
 16  Weekend                  12330 non-null  bool
 17  Revenue                  12330 non-null  bool
dtypes: bool(2), float64(7), int64(7), object(2)
memory usage: 1.5+ MB
```

Xem chi tiết trong mỗi cột có giá trị null (thiếu) hay không:

```python
# Kiểm tra số lượng giá trị thiếu trong từng cột của DataFrame
missing_values = df.isnull().sum()
missing_values
```

```
Administrative             0
Administrative_Duration    0
Informational              0
Informational_Duration     0
ProductRelated             0
ProductRelated_Duration    0
BounceRates                0
ExitRates                  0
PageValues                 0
SpecialDay                 0
Month                      0
OperatingSystems           0
Browser                    0
Region                     0
TrafficType                0
VisitorType                0
Weekend                    0
Revenue                    0
dtype: int64
```

Xem các thống kê mô tả của các cột có kiểu dữ liệu là int và float:

	count	mean	std	min	25%	50%	75%	max
Administrative	12330.0	2.315166	3.321784	0.0	0.000000	1.000000	4.000000	27.000000
Administrative_Duration	12330.0	80.818611	176.779107	0.0	0.000000	7.500000	93.256250	3398.750000
Informational	12330.0	0.503569	1.270156	0.0	0.000000	0.000000	0.000000	24.000000
Informational_Duration	12330.0	34.472398	140.749294	0.0	0.000000	0.000000	0.000000	2549.375000
ProductRelated	12330.0	31.731468	44.475503	0.0	7.000000	18.000000	38.000000	705.000000
ProductRelated_Duration	12330.0	1194.746220	1913.669288	0.0	184.137500	598.936905	1464.157214	63973.522230
BounceRates	12330.0	0.022191	0.048488	0.0	0.000000	0.003112	0.016813	0.200000
ExitRates	12330.0	0.043073	0.048597	0.0	0.014286	0.025156	0.050000	0.200000
PageValues	12330.0	5.889258	18.568437	0.0	0.000000	0.000000	0.000000	361.763742
SpecialDay	12330.0	0.061427	0.198917	0.0	0.000000	0.000000	0.000000	1.000000
OperatingSystems	12330.0	2.124006	0.911325	1.0	2.000000	2.000000	3.000000	8.000000
Browser	12330.0	2.357097	1.717277	1.0	2.000000	2.000000	2.000000	13.000000
Region	12330.0	3.147364	2.401591	1.0	1.000000	3.000000	4.000000	9.000000
TrafficType	12330.0	4.069586	4.025169	1.0	2.000000	2.000000	4.000000	20.000000

Tiếp tục xem thống kê mô tả của 2 kiểu dữ liệu còn lại:

```
1   df.describe(include = 'object')
```

	Month	VisitorType
count	12330	12330
unique	10	3
top	May	Returning_Visitor
freq	3364	10551

```
1   df.describe(include = 'bool')
```

	Weekend	Revenue
count	12330	12330
unique	2	2
top	False	False
freq	9462	10422

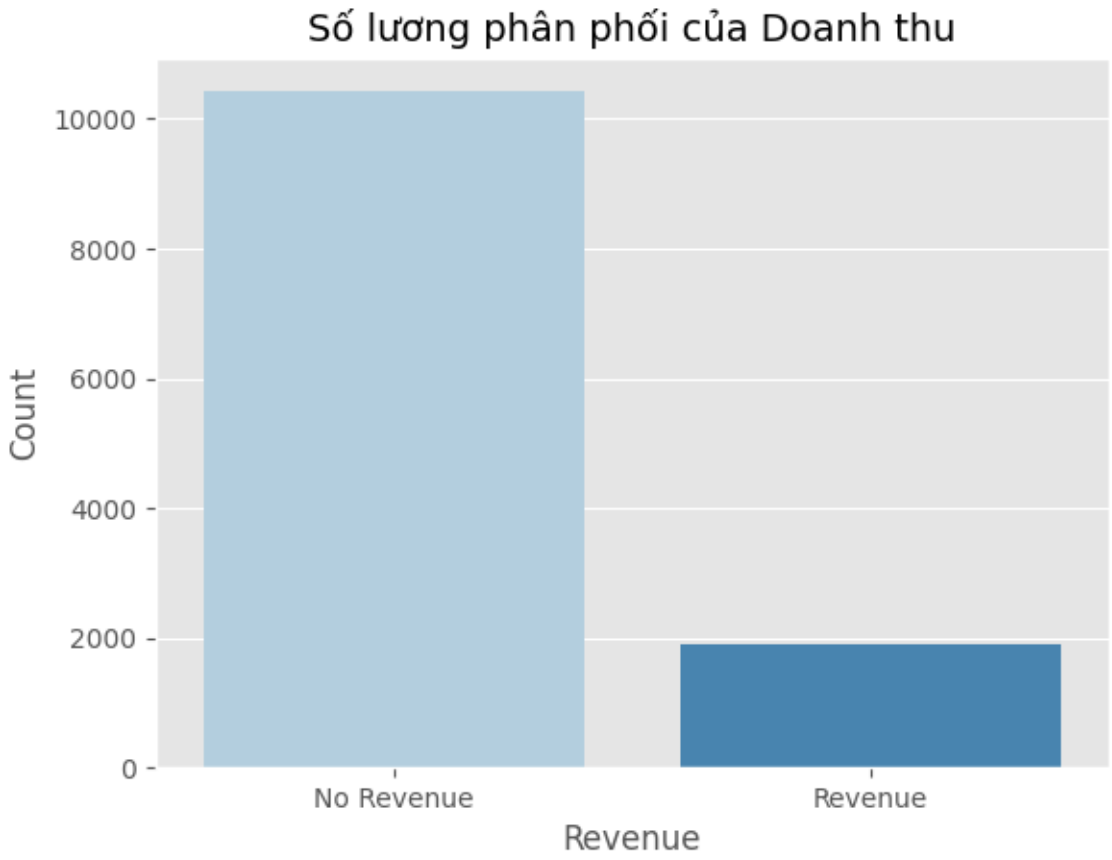
Đếm giá trị duy nhất trong mỗi cột. Trong đó cột dự đoán (Revenue) có 2 giá trị duy nhất là True và False,tương ứng với True là mua hàng còn 0 là không mua hàng.

```
# Đếm số lượng giá trị duy nhất trong mỗi cột và sắp xếp theo thứ tự tăng dần
unique_counts = df.nunique().sort_values(ascending=True)
unique_counts
```

```
Revenue                     2
Weekend                     2
VisitorType                 3
SpecialDay                  6
OperatingSystems            8
Region                      9
Month                      10
Browser                    13
Informational              17
TrafficType                20
Administrative             27
ProductRelated            311
Informational_Duration   1258
BounceRates              1872
PageValues               2704
Administrative_Duration  3335
ExitRates                4777
ProductRelated_Duration  9551
dtype: int64
```

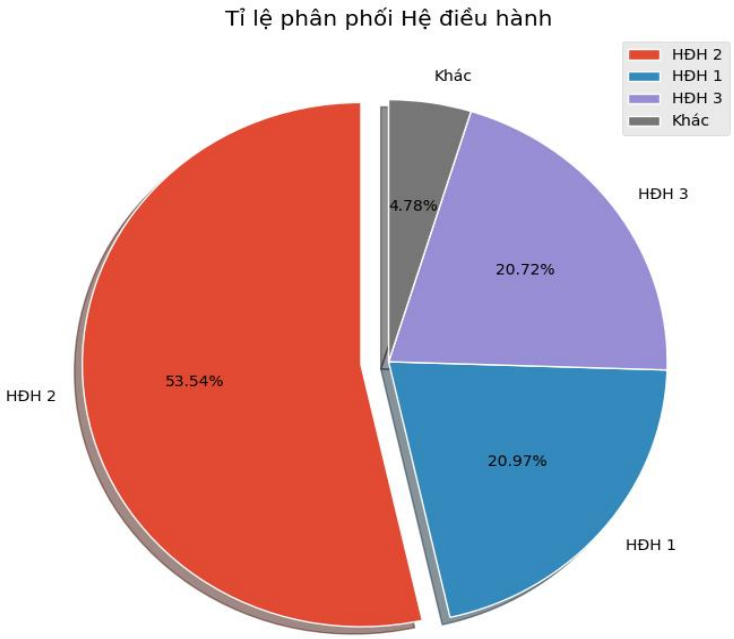
2.3. Phân tích dữ liệu khám phá (Exploratory Data Analysis)

Biểu đồ Histogram thể hiện phân phối giữa hai biến ‘Doanh Thu’ (Revenue), trong đó số lượng người không mua hàng (No Revenue = False) cao gấp 9 lần số lượng người mua hàng (Revenue = True).

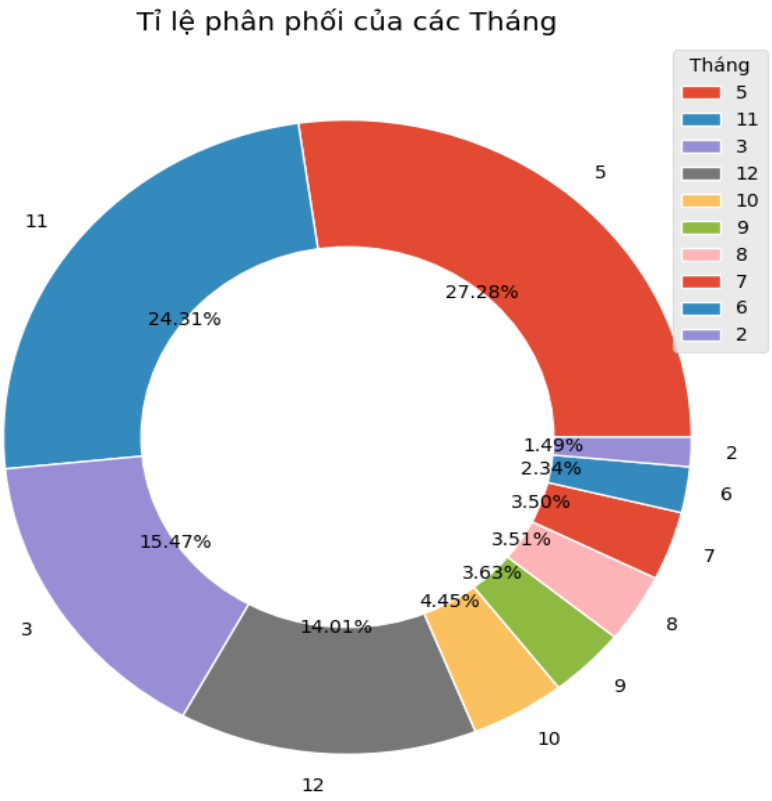


Hình 1 : Phân bố giá trị của biên Revenue (Dùng để dự đoán)

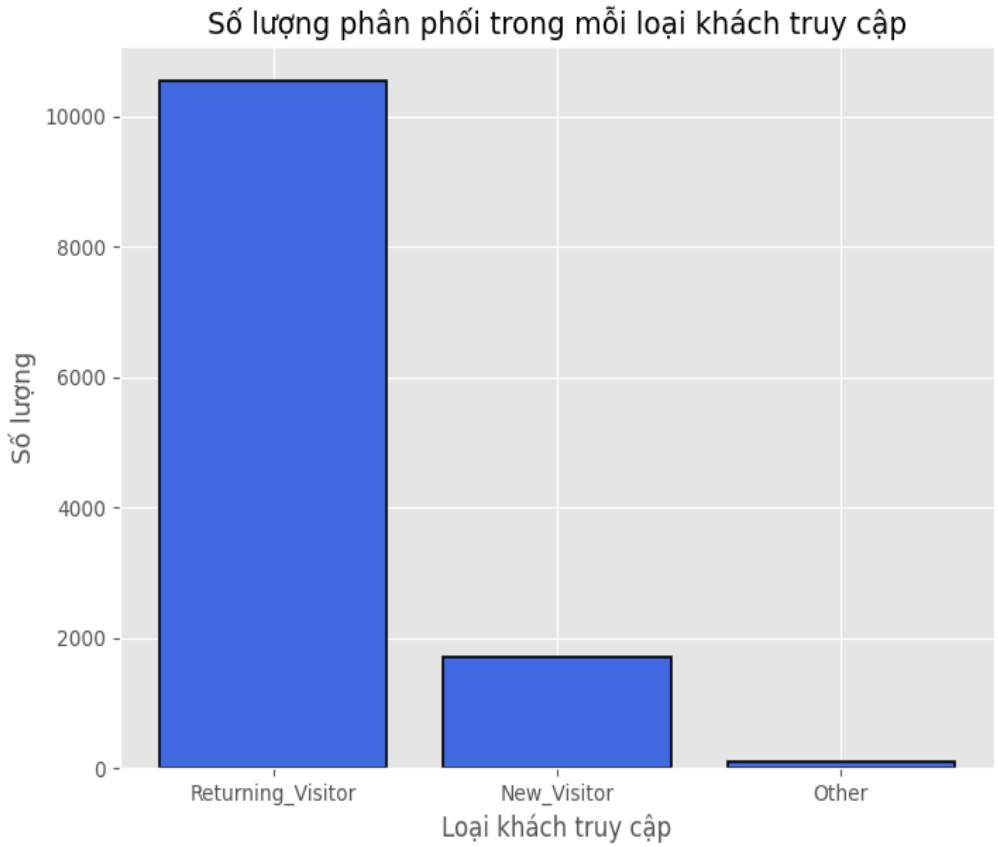
Biểu đồ Pie Chart thể hiện tỉ lệ phân phối của biến 'operating_system'. Tỷ lệ người dùng sử dụng hệ điều hành 2 (HĐH 2) chiếm đa số với tỷ lệ 53.34%. Tiếp theo là hệ điều hành 1 (HĐH 1) và hệ điều hành 3 (HĐH 3), lần lượt chiếm 20.97% và 20.72%. Các hệ điều hành khách chiếm một phần rất nhỏ với tỷ lệ 4.78%.



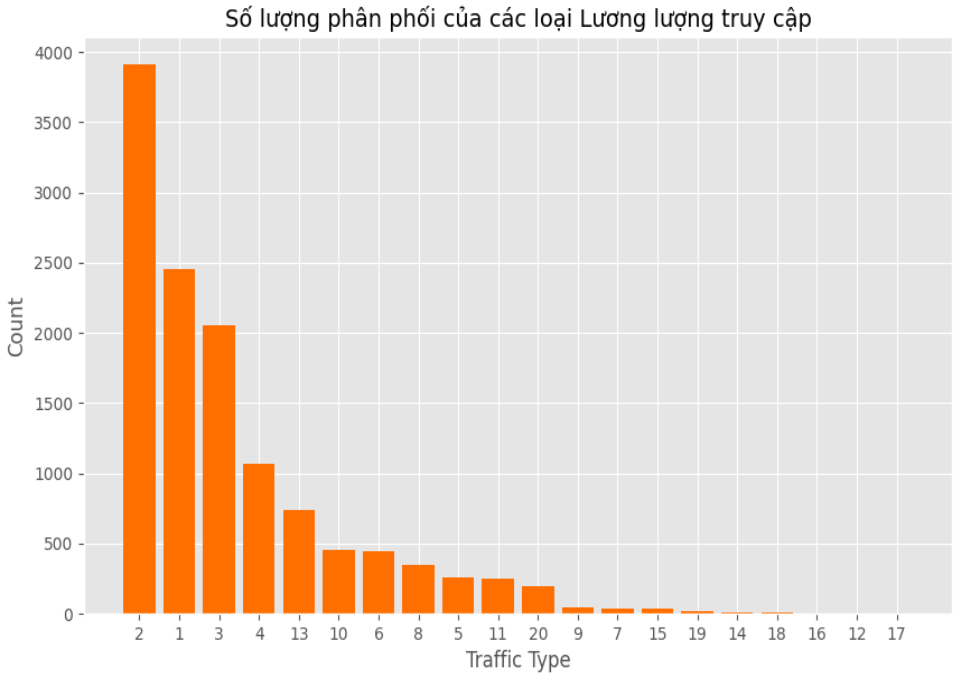
Biểu đồ Donut Chart thể hiện tỉ lệ phân phối của ‘Month’ trong bộ dữ liệu. Biểu đồ cho thấy rằng tháng 5 và tháng 11 là hai tháng có tỉ lệ cao nhất, lần lượt là 27.78% và 24.31%. Tháng 3 và tháng 12 là hai tháng tiếp theo có tỉ lệ cao, với lần lượt 15.37% và 14.01%. Các tháng còn lại có tỉ lệ thấp hơn



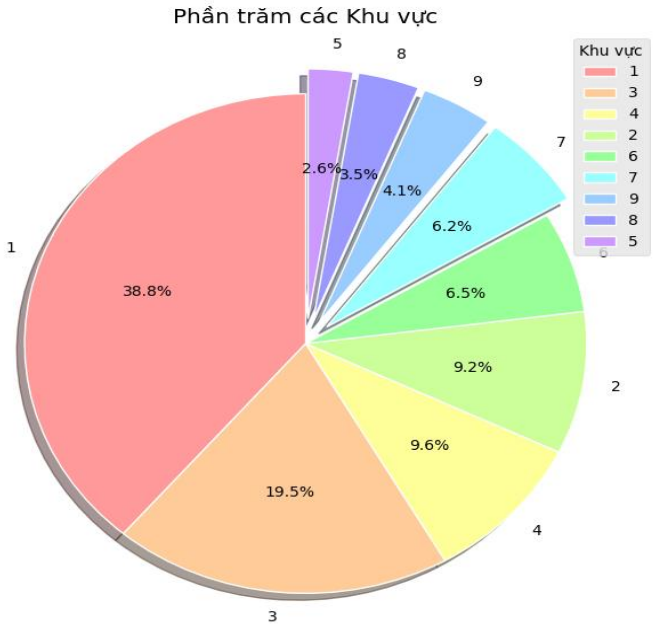
Bar Chart thể hiện số lượng phân phối của biến ‘Loại khách’ (VistorType). Có thể dễ dàng thấy rằng, số lượng vị khách trở lại (Return_visitor) chiếm đa số với hơn 10.000 người, xếp sau đó là vị khách mới (New_visitor) và vị khách khác (Other) với số lượng phân phối rất thấp dưới 2.000 người.



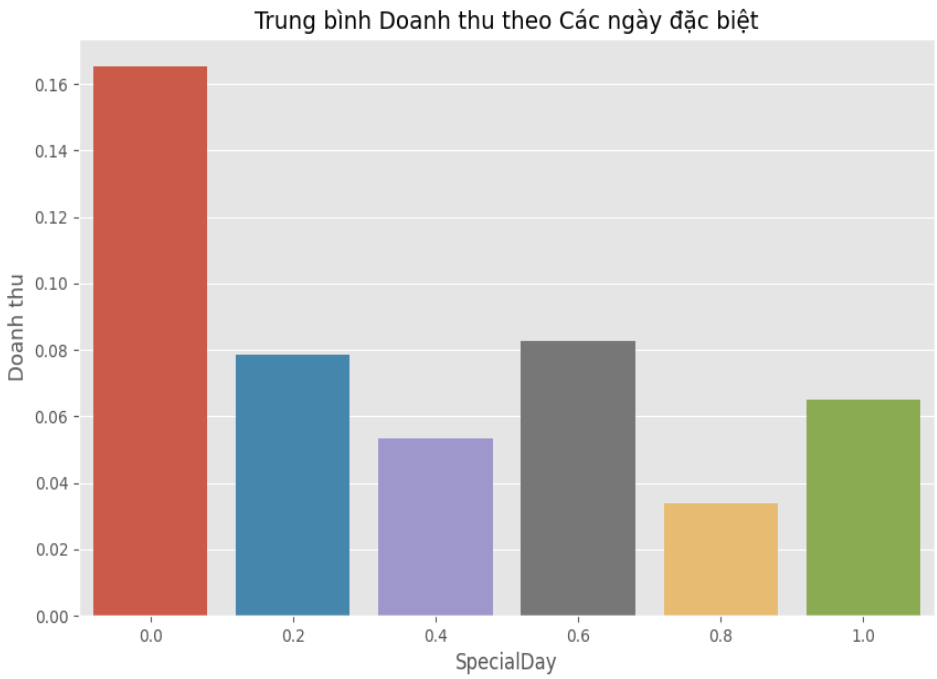
Biểu đồ Bar Chart thể hiện số lượng phân phối của biến 'Loại truy cập' (TrafficType). Dễ dàng nhận thấy rằng, top 4 loại truy cập phổ biến là : Loại truy cập 2 với gần 4.000 người sau đó là Loại truy cập 1 và 3 với hơn 2.000 người và Loại truy cập 4 với trên 1.000 người. Các loại truy cập còn lại có phân phối thấp, dưới 1.000 người



Biểu đồ Pie Chart thể hiện phần trăm của biến ‘Khu vực’ (Region). Được thể hiện rõ ràng rằng, khu vực 1 và 3 chiếm phần trăm cao nhất trong bộ dữ liệu. Trong khi đó, phần trăm thấp nhất được chia cho các khu vực 5, 8 và 9, mỗi khu vực chiếm ít hơn 5% tổng phần trăm.



Vẽ biểu đồ Bar Chart thể hiện Trung bình doanh thu theo Ngày đặc biệt (Weekend). Có thể thấy rằng trong các ngày không đặc biệt (SpecialDay = 0.0) có giá trị trung bình của doanh thu là cao nhất (hơn 0.16). Tuy nhiên, trong các ngày đặc biệt (SpecialDay = 0.2, 0.4, 0.6, 0.8, 1) giá trị trung bình doanh thu giảm xuống. Có thể giả định rằng các ngày đặc biệt như ngày lễ hoặc sự kiện đặc biệt có thể ảnh hưởng đến hành vi mua sắm khách hàng và doanh thu. Giá trị trung bình của doanh thu có thể giảm do sự tăng cường hoạt động mua sắm trong các ngày này hoặc sự giảm cấp của giá trị giao dịch trong các ngày quan trọng.



2.4. Tiền xử lý dữ liệu (Data Preprocessing)

Trong tập dữ liệu, có những trường hợp xuất hiện giá trị trùng lặp, điều này có thể gây ảnh hưởng không mong muốn đến hiệu suất của mô hình máy học. Vì vậy, trước khi áp dụng mô hình máy học, việc tìm và loại bỏ các giá trị trùng lặp là cần thiết. Trong tập dữ liệu hiện tại, đã được xác định tồn tại 125 dòng dữ liệu có giá trị trùng lặp. Sau khi tiến hành quá trình loại bỏ, kích thước của tập dữ liệu đã được rút gọn từ 12,330 dòng ban đầu xuống còn 12,205 dòng, giúp tối ưu hóa chất lượng và độ chính xác của mô hình

máy học.

```
print("Số lượng dòng dữ liệu trùng lặp:", df_copy.duplicated().sum())
```

```
Số lượng dòng dữ liệu trùng lặp: 125
```

```
# Kiểm tra và xóa bỏ các dữ liệu trùng lặp
df_copy.drop_duplicates(inplace=True)

# In thông tin về số lượng dòng sau khi xóa bỏ các dữ liệu trùng lặp
print("Số lượng dòng sau khi xóa bỏ các dữ liệu trùng lặp:", len(df_copy))
```

```
Số lượng dòng sau khi xóa bỏ các dữ liệu trùng lặp: 12205
```

Đối với các biến có giá trị liên tục (continuous variables) như 'Administrative', 'Administrative_Duration', 'Informational', 'Informational_Duration', 'ProductRelated', 'BounceRates', 'ExitRates', 'PageValues', 'ProductRelated_Duration', chúng ta cần áp dụng phương pháp chuẩn hóa Min-Max Scaler. Mục tiêu là đưa các giá trị dữ liệu liên tục về khoảng [0, 1], giúp mô hình có khả năng tổng quát hóa tốt hơn và học hiệu quả hơn từ dữ liệu.

```
# Sử dụng MinMaxScaler để chuẩn hóa các biến liên tục trong DataFrame df_copy
sc = MinMaxScaler()
df_copy[continuous_vars] = sc.fit_transform(df_copy[continuous_vars])

# In ra các giá trị chuẩn hóa của các biến liên tục
df_copy[continuous_vars]
```

	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	BounceRates	ExitRates
0	0.000000	0.000000	0.0	0.0	0.001418	1.000000	1.000000
1	0.000000	0.000000	0.0	0.0	0.002837	0.000000	0.500000
2	0.000000	0.000000	0.0	0.0	0.001418	1.000000	1.000000
3	0.000000	0.000000	0.0	0.0	0.002837	0.250000	0.700000
4	0.000000	0.000000	0.0	0.0	0.014184	0.100000	0.250000

Sử dụng phương pháp mã hóa One-Hot Encoding cho cột ‘VistiorType’ để tạo ra các cột tương ứng với từng giá trị riêng biệt trong cột ban đầu. Trong cột mới này giá trị 1 sẽ được gán nếu giá trị tương ứng xuất hiện, ngược lại giá trị 0 sẽ được gán nếu không có giá trị đó.

```python
# Sử dụng phương pháp One-Hot Encoding để mã hóa biến VisitorType trong DataFrame df_copy
visitortype_encode = pd.get_dummies(df_copy['VisitorType'])

# In ra DataFrame mã hóa của biến VisitorType
visitortype_encode
```

	New_Visitor	Other	Returning_Visitor
0	0	0	1
1	0	0	1
2	0	0	1
3	0	0	1
4	0	0	1

Do sự mất cân bằng giữa hai lớp 0 và 1 trong biến dự đoán 'Revenue', nếu không thực hiện việc cân bằng dữ liệu, mô hình huấn luyện có khả năng học và dự đoán chủ yếu theo lớp 0 hơn là lớp 1. Hiệu quả dự đoán trên tập dữ liệu mới có thể bị ảnh hưởng bởi sự mất cân bằng này.

```python
# Tạo biến X (Features) và y (Target) từ DataFrame df_copy
X = df_copy.drop('Revenue', axis=1)
y = df_copy['Revenue']
```

```python
# In ra số lượng mẫu 'Revenue' trước khi sử dụng kỹ thuật Oversampling SMOTE
print("Số lượng mẫu trước oversampling:")
print(pd.Series(y).value_counts())
```

```
Số lượng mẫu trước oversampling:
0    10297
1     1908
Name: Revenue, dtype: int64
```

Trước khi đưa dữ liệu vào mô hình máy học, thực hiện cân bằng mẫu lớp thiểu số (lớp 1) bằng phương pháp Oversampling, cụ thể là sử dụng phương pháp SMOTE (Synthetic Minority Over-sampling Technique). Phương pháp này tạo ra các mẫu lớp thiểu số bằng cách tạo ra các điểm dữ liệu mới dựa trên việc kết hợp giữa các máu gần nhau trong lớp đó, SMOTE giúp tăng cường số lượng mẫu trong lớp thiểu số, làm cho mô hình học tốt hơn và dự đoán cân bằng hơn giữa các lớp trong mô hình máy học.

```python
# Áp dụng kỹ thuật Oversampling SMOTE cho dữ liệu
smote = SMOTE(sampling_strategy='minority')
X_resampled, y_resampled = smote.fit_resample(X, y)
```

```python
# In ra số lượng mẫu 'Revenue' sau khi sử dụng kỹ thuật Oversampling SMOTE
print("Số lượng mẫu sau oversampling:")
print(pd.Series(y_resampled).value_counts())
```

```
Số lượng mẫu sau oversampling:
0    10297
1    10297
Name: Revenue, dtype: int64
```

Dữ liệu được chia thành hai tập: tập huấn luyện (Training) chiếm 60% và tập kiểm tra (Testing) chiếm 40%. Tập huấn luyện được sử dụng để huấn luyện mô hình máy học, còn tập kiểm tra được sử dụng để đánh giá mô hình sau khi hoàn thành quá trình huấn luyện. Khi sử dụng hàm train_test_split với tham số ‘stratify = y_resampled’, tham số này đảm bảo phân phối các lớp trong tập huấn luyện và tập thử nghiệm tương tự nhau, tránh tình trạng mất cân bằng lớp.

```python
# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=.4, random_state=42, stratify=y_resampled)

# In thông tin về số dòng, số cột của từng tập dữ liệu
print("Số dòng, số cột của X_train:", X_train.shape)
print("Số dòng, số cột của X_test:", X_test.shape)
print("Số dòng, số cột của y_train:", y_train.shape)
print("Số dòng, số cột của y_test:", y_test.shape)
```

```
Số dòng, số cột của X_train: (12356, 19)
Số dòng, số cột của X_test: (8238, 19)
Số dòng, số cột của y_train: (12356,)
Số dòng, số cột của y_test: (8238,)
```

2.5. Dự đoán mô hình học máy (Predict Machine Learning)

2.5.1. Hồi quy Logistic (Logistic Regression)

Mô hình được khởi tạo từ thư viện sklearn. Sau đó, mô hình được huấn luyện bằng tập dữ liệu (Training) và sau đó sử dụng phương thức ‘predict()’ để dự đoán trên tập kiểm tra. Kết quả cho thấy với 20 dữ liệu đầu tiên từ tập X_test, mô hình đã dự đoán đúng 17 trên tổng số 20 điểm dữ liệu. Mô hình thể hiện khả năng dự đoán rất tốt với chỉ 3 điểm dữ liệu bị dự đoán sai.

```
# Tạo một mô hình Logistic Regression mới
lr_model = LogisticRegression()

# Huấn luyện mô hình trên tập huấn luyện
lr_model.fit(X_train, y_train)

```

```
▾ LogisticRegression
LogisticRegression()
```

```
# Sử dụng mô hình Logistic Regression để dự đoán nhãn của dữ liệu kiểm tra
lr_pred = lr_model.predict(X_test)

# In ra 20 dự đoán đầu tiên
print(lr_pred[0:20])
```

```
[0 1 0 1 0 1 0 1 0 0 0 0 1 0 1 0 0 0 1 1]
```

```
print(y_test[:20].values)
```

```
[0 1 0 1 0 1 0 0 0 1 0 0 1 0 1 0 0 0 0 1]
```

In ra bảng phân loại báo cáo (classification report) trên tập dữ liệu kiểm tra (Testing) ta có thể thấy rằng:

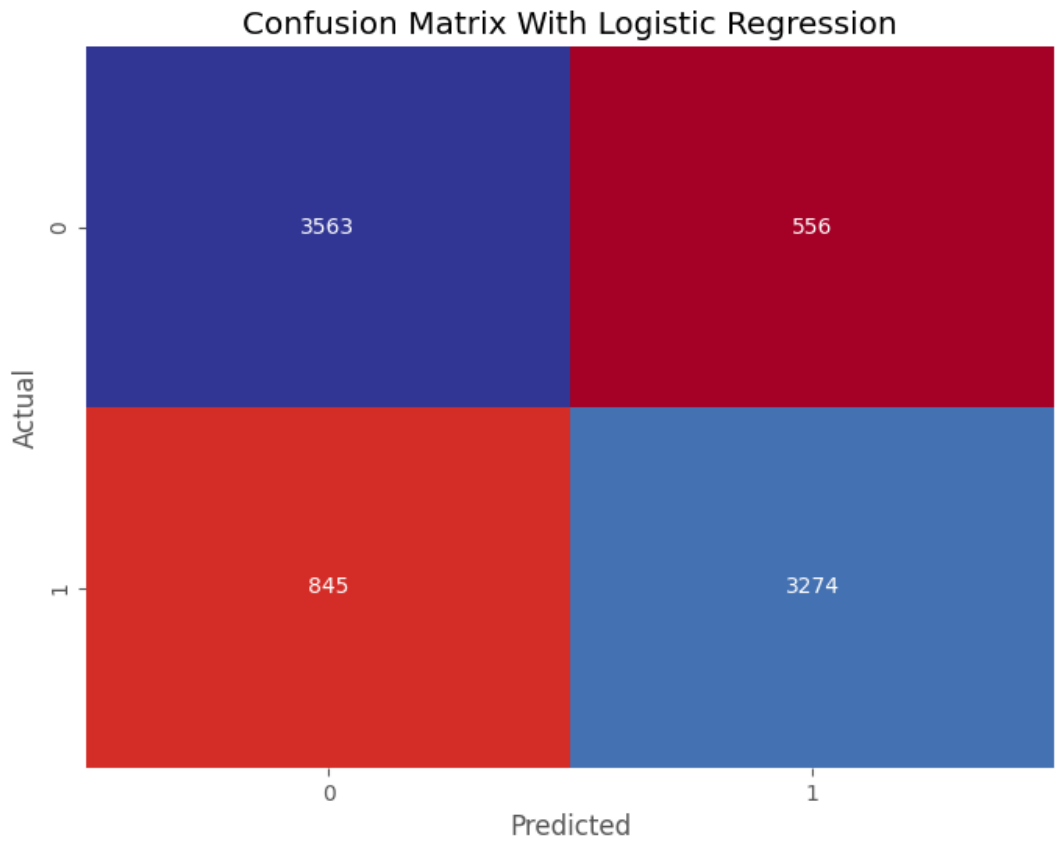
- Accuracy: mô hình có độ chính xác khoảng 82.99%. Điều này cho thấy mô hình đưa ra dự đoán chính xác hầu hết các trường hợp.
- Precision và Recall: Đối với lớp 0 (negative), độ chính xác (precision) là 0.81 và độ phủ (recall) là 0.87.Đối với lớp 1 (positive), độ chính xác (precision) là 0.85 và độ phủ (recall) là 0.79. Mô hình có khả năng tìm ra các trường hợp positive không quá tốt (recall thấp), nhưng khi đưa ra dự đoán positive thì khả năng chính xác khá tốt (precision cao)
- F1-score : F1-score cho lớp 0 là 0.84 và cho lớp 1 là 0.82. F1-score là một số liệu kết hợp giữa precision và recall, và trong trường hợp này, nó cho thấy mô hình có hiệu suất tương đối cân bằng giữa việc đưa ra dự đoán chính xác và việc tìm ra các trường hợp positive.

```python
# In ra các giá trị đánh giá hiệu suất của mô hình Logistic Regression
print(f"Accuracy của mô hình Logistic: {lr_accuracy}\n")
print(classification_report(y_test, lr_pred))
```

```
Accuracy của mô hình Logistic: 0.8299344501092498

              precision    recall  f1-score   support

           0       0.81      0.87      0.84      4119
           1       0.85      0.79      0.82      4119

    accuracy                           0.83      8238
   macro avg       0.83      0.83      0.83      8238
weighted avg       0.83      0.83      0.83      8238
```

Mô hình Logistic Regression có hiệu suất tương đối tốt trong việc dự đoán hành vi mua sắm của khách hàng trên tập dữ liệu kiểm tra. Tuy nhiên, việc tìm ra các trường hợp positive (khách hàng thực sự mua sắm) có thể được cải thiện để tăng cường khả năng dự đoán các hành vi mua sắm quan trọng.

Vẽ ma trận nhầm lẫn (confusion matrix) trên tập dữ liệu kiểm tra (Testing) cho mô hình Logistic Regression : Trong đó, đối với lớp 0, mô hình đã dự đoán chính xác 3563 người thực sự thuộc lớp 0 và sai 556 người bằng cách dự đoán là lớp 1 nhưng thực tế thuộc lớp 0. Đối với lớp 1, mô hình dự đoán chính xác 3274 người thực sự thuộc lớp 1 và sai 845 người bằng cách dự đoán là lớp 0 nhưng thực tế thuộc lớp 1.



Hình 2 : Kết quả dự đoán (Ma trận hỗn loạn) trên tập test của Mô hình Logistic Regression

2.5.2. Cây quyết định (Decision Tree)

Sau khi khởi tạo mô hình từ thư viện sklearn, ta tiến hành huấn luyện mô hình bằng tập dữ liệu huấn luyện (Training). Sau đó, sử dụng phương thức 'predict()' để thực hiện dự đoán trên tập kiểm tra. Khi xem xét 20 dữ liệu đầu tiên từ tập X_test, có thể thấy mô hình đã dự đoán chính xác 18 trên 20 điểm dữ liệu. Điều này cho thấy mô hình có khả năng dự đoán rất tốt, chỉ mắc sai sót trong 2 điểm dữ liệu.

```
09]  1  # Khởi tạo mô hình Decision Tree và huấn luyện trên tập huấn luyện
     2  dt_model = DecisionTreeClassifier()
     3  dt_model.fit(X_train, y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

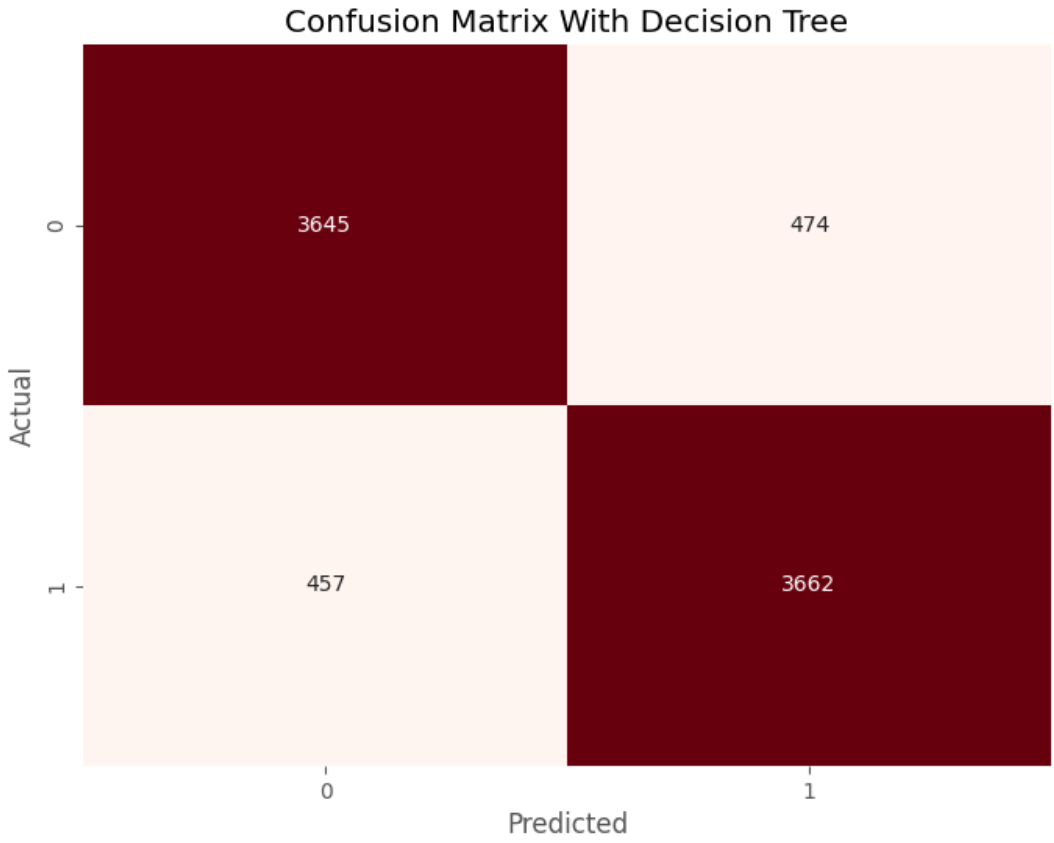
```
     1  # Dự đoán nhãn cho tập dữ liệu kiểm tra bằng mô hình Decision Tree
     2  dt_pred = dt_model.predict(X_test)
     3
     4  # Hiển thị 20 dự đoán đầu tiên
     5  print(dt_pred[0:20])
```

```
[0 1 0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 1]
```

```
11]  1  print(y_test[:20].values)
```

```
[0 1 0 1 0 1 0 0 0 1 0 0 1 0 1 0 0 0 0 1]
```

Vẽ ma trận nhầm lẫn (confusion matrix) trên tập dữ liệu kiểm tra (Testing) cho mô hình Decision Tree : Trong đó, đối với lớp 0, mô hình đã dự đoán chính xác 3645 người thực sự thuộc lớp 0 và sai 474 người bằng cách dự đoán là lớp 1 nhưng thực tế thuộc lớp 0. Đối với lớp 1, mô hình dự đoán chính xác 3662 người thực sự thuộc lớp 1 và sai 457 người bằng cách dự đoán là lớp 0 nhưng thực tế thuộc lớp 1.



Hình 3 : Kết quả dự đoán (Ma trận hỗn loạn) trên tập test của Mô hình Decision Tree

In ra bảng phân loại báo cáo (classification report) trên tập dữ liệu kiểm tra (Testing) ta có thể thấy rằng :

- Accuracy: mô hình có độ chính xác khoảng 88.70%. Điều này cho thấy mô hình đưa ra dự đoán chính xác cho hầu hết các trường hợp.
- Precision và Recall: đối với cả lớp 0 (negative) và lớp 1 (positive), độ chính xác (precision) và độ phủ (recall) đều ở mức rất cao, xấp xỉ 89%. Điều này cho thấy mô hình có khả năng đưa ra dự đoán chính xác và tìm ra các trường hợp positive một cách hiệu quả.
- F1-score: F1-score cho cả lớp 0 và lớp 1 đều ở mức rất cao, xấp xỉ 0.89%. Điều này cho thấy mô hình có hiệu suất tương đương cả về việc đưa ra dự đoán chính xác và việc tìm ra các trường hợp positive.

```python
# In ra giá trị accuracy của mô hình Decision Tree
print(f"Accuracy của mô hình Decision Tree: {dt_accuracy}\n")

# In ra classification report của mô hình Decision Tree
print(classification_report(y_test, dt_pred))
```

```
Accuracy của mô hình Decision Tree: 0.8869871327992231

              precision    recall  f1-score   support

           0       0.89      0.88      0.89      4119
           1       0.89      0.89      0.89      4119

    accuracy                           0.89      8238
   macro avg       0.89      0.89      0.89      8238
weighted avg       0.89      0.89      0.89      8238
```

Mô hình Decision Tree có hiệu suất rất tốt trong việc dự đoán hành vi mua sắm của khách hàng trên tập dữ liệu kiểm tra. Mô hình này có khả năng đưa ra dự đoán chính xác và tìm ra các trường hợp positive một cách hiệu quả. Nếu các số liệu đánh giá cao là ưu tiên của bạn, mô hình Decision Tree có thể là một lựa chọn tốt.

2.5.3. Rừng ngẫu nhiên (Random Forest)

Mô hình được khởi tạo từ thư viện sklearn. Sau đó, mô hình được huấn luyện bằng tập dữ liệu (Training), và sau đó sử dụng phương thức “predict()” để thực hiện dự đoán trên tập kiểm tra. Khi xem xét 20 dữ liệu đầu tiên từ tập X_test, có thể thấy mô hình đã dự đoán đúng tất cả 20 điểm dữ liệu. Mô hình thể hiện khả năng dự đoán rất tốt, không mắc phải bất kỳ sai sót nào trong 20 điểm dữ liệu đầu tiên.

```
# Tạo một mô hình Random Forest và huấn luyện trên tập huấn luyện
rf_model = RandomForestClassifier()
rf_model.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier()
```

```
# Sử dụng mô hình Logistic Regression để dự đoán nhãn của dữ liệu kiểm tra
rf_pred = rf_model.predict(X_test)

# In ra 20 dự đoán đầu tiên
print(rf_pred[0:20])
```

```
[0 1 0 1 0 1 0 0 0 1 0 0 1 0 1 0 0 0 0 1]
```

```
print(y_test[:20].values)
```

```
[0 1 0 1 0 1 0 0 0 1 0 0 1 0 1 0 0 0 0 1]
```

In ra bảng phân loại báo cáo (classification report) trên tập dữ liệu kiểm tra (Testing) ta có thể thấy rằng :

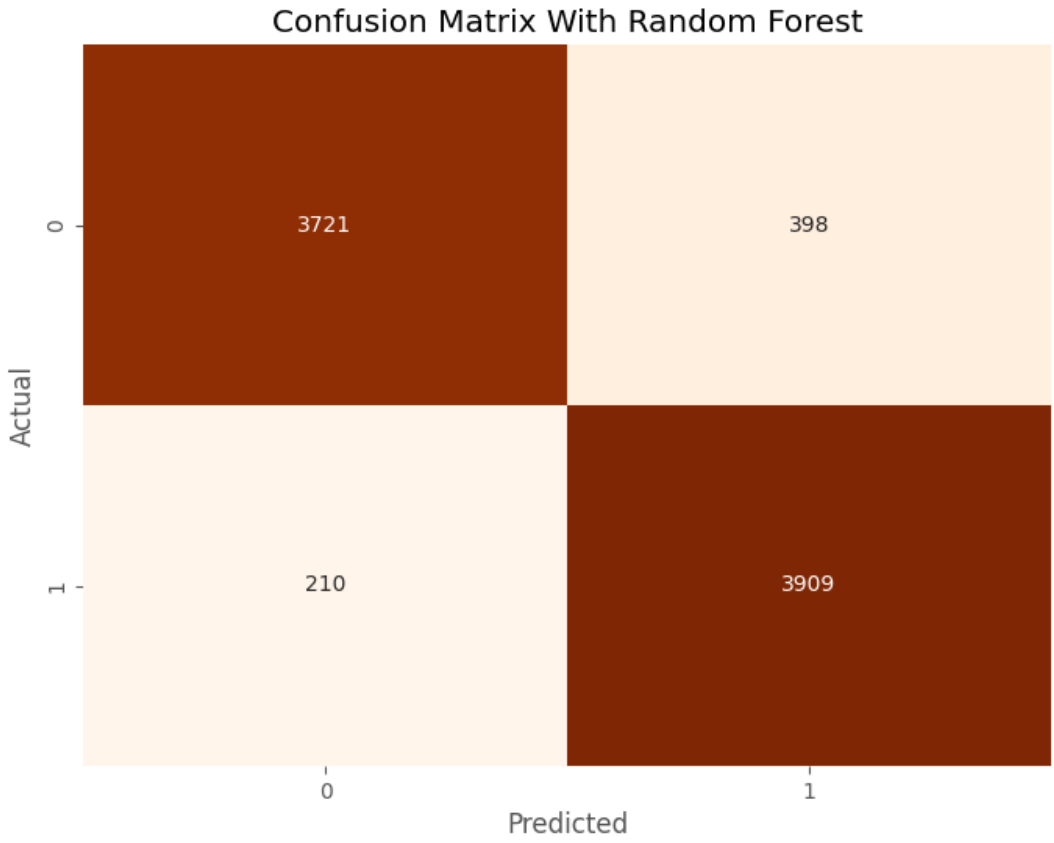
- Accuracy: Mô hình có độ chính xác (accuracy) khoảng 92.62%. Điều này cho thấy mô hình đưa ra dự đoán chính xác cho hầu hết các trường hợp.
- Precision và Recall: Đối với cả lớp 0 (negative) và lớp 1 (positive), độ chính xác (precision) và độ phủ (recall) đều ở mức cao, lần lượt là 0.95 và 0.90 cho lớp 0, và 0.91 và 0.95 cho lớp 1. Điều này cho thấy mô hình có khả năng đưa ra dự đoán chính xác và tìm ra các trường hợp positive một cách hiệu quả.
- F1-score: F1-score cho cả lớp 0 và lớp 1 đều ở mức cao, xấp xỉ 0.92 và 0.93 tương ứng. Điều này cho thấy mô hình có hiệu suất tương đương cả về việc đưa ra dự đoán chính xác và việc tìm ra các trường hợp positive.

```python
# In ra các giá trị đánh giá hiệu suất của mô hình Logistic Regression
print(f"Accuracy của mô hình Random Forest: {rf_accuracy}\n")
print(classification_report(y_test, rf_pred))
```

```
Accuracy của mô hình Random Forest: 0.926195678562758

              precision    recall  f1-score   support

           0       0.95      0.90      0.92      4119
           1       0.91      0.95      0.93      4119

    accuracy                           0.93      8238
   macro avg       0.93      0.93      0.93      8238
weighted avg       0.93      0.93      0.93      8238
```

Mô hình Random Forest có hiệu suất rất tốt trong việc dự đoán hành vi mua sắm của khách hàng trên tập dữ liệu kiểm tra. Mô hình này có khả năng đưa ra dự đoán chính xác và tìm ra các trường hợp positive một cách hiệu quả, vượt qua cả mô hình Decision Tree và Logistic Regression về độ chính xác. Nếu độ chính xác là ưu tiên của bạn, mô hình Random Forest có thể là lựa chọn tốt nhất trong trường hợp này.

Vẽ ma trận nhầm lẫn (confusion matrix) trên tập dữ liệu kiểm tra (Testing) đối với mô hình Random Forest : trong đó, đối với lớp 0, mô hình dự đoán chính xác 3721 người thực sự thuộc lớp 0 và sai 398 người bằng cách dự đoán là lớp 1 nhưng thực tế thuộc lớp 0. Đối với lớp 1, mô hình dự đoán chính xác 3909 người thực sự thuộc lớp 1 và sai 210 người bằng cách dự đoán là lớp 0 nhưng thực tế thuộc lớp 1.



Hình 4 : Kết quả dự đoán (Ma trận hỗn loạn) trên tập test của Mô hình Random Forest

2.6. So Sánh và lựa chọn mô hình dự đoán tốt nhất

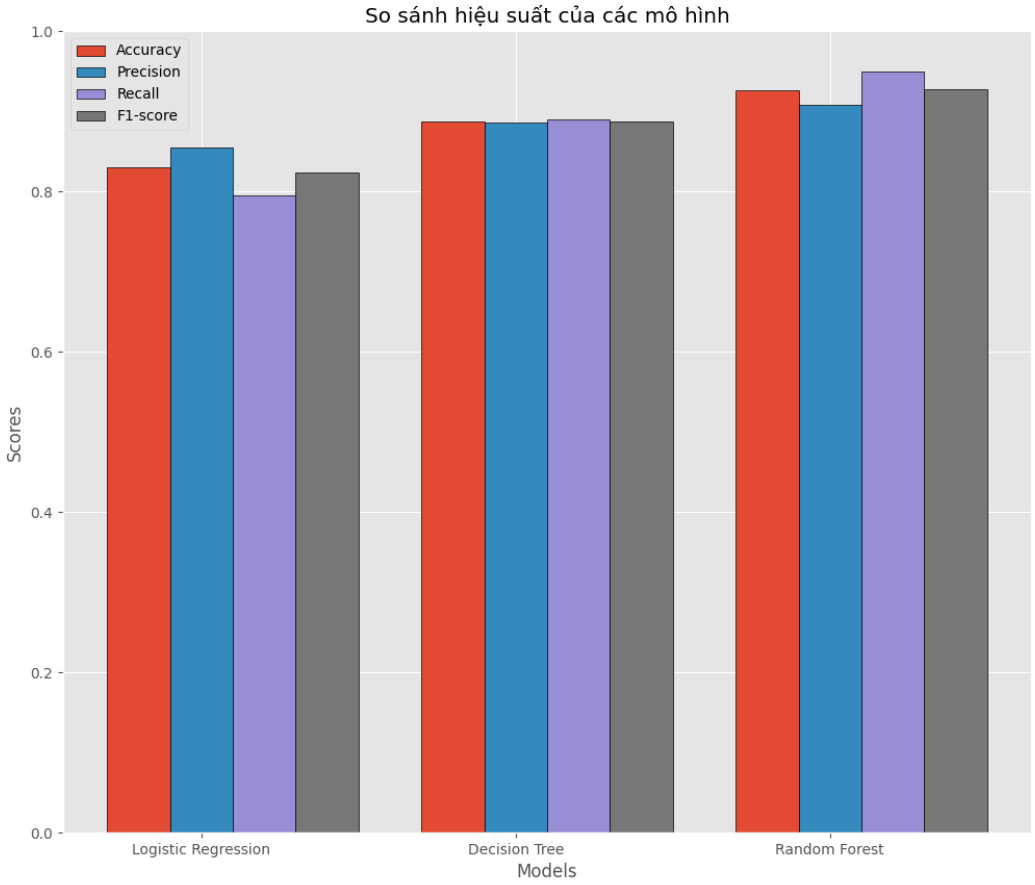
Dựa trên bảng đánh giá hiệu suất của ba mô hình: Random Forest, Decision Tree và Logistic Regression, ta có thể thấy rõ sự khác biệt trong hiệu suất của chúng. Dưới đây là một số nhận xét và so sánh giữa ba mô hình:

Model	Precision	Recall	F1-Score	Accuracy
Logistic Regression	0.854830	0.794853	0.823751	0.829934
Decision Tree	0.885397	0.889051	0.887220	0.886987
Random Forest	0.907592	0.949017	0.927842	0.926196

- Accuracy: Random Forest có accuracy cao nhất (92.62%), Decision Tree đứng thứ hai (88.70%), và Logistic Regression có accuracy thấp nhất (82.99%).

- Precision và Recall: Random Forest có precision và recall đều rất cao, cho thấy mô hình này đồng thời đưa ra dự đoán chính xác và tìm ra các trường hợp positive một cách hiệu quả. Decision Tree cũng có precision và recall tương đối cân bằng, trong khi Logistic Regression có precision cao hơn nhưng recall thấp hơn.

- F1-score: Random Forest có F1-score cao nhất (92.78%), Decision Tree đứng thứ hai (88.72%), và Logistic Regression có F1-score thấp nhất (82.38%). F1-score thể hiện sự cân bằng giữa precision và recall, và cho thấy hiệu suất tổng thể của mô hình.



Hình 5 : Biểu đồ so sánh hiệu suất của 3 mô hình

Dựa trên các số liệu này, có thể kết luận rằng mô hình Random Forest có hiệu suất tốt nhất trong ba mô hình này, với accuracy, precision, recall và F1-score đều ở mức cao. Decision Tree cũng có hiệu suất tốt, nhưng thấp hơn so với Random Forest. Logistic Regression có hiệu suất thấp hơn cả hai mô hình còn lại.

CHƯƠNG IV: KẾT LUẬN VÀ KIẾN NGHỊ

1. Kết luận chung và kết quả đạt được

Kết luận chung:

Hiệu suất của các mô hình: Ba mô hình đã được đào tạo và đánh giá trên tập dữ liệu kiểm tra. Kết quả đạt được thể hiện qua các chỉ số như accuracy, precision, recall và F1-score. Random Forest đạt hiệu suất cao nhất với accuracy 92.62%, precision 90.76%, recall 94.90% và F1-score 92.78%. Decision Tree có hiệu suất ổn định với accuracy 88.70%, precision 88.54%, recall 88.91% và F1-score 88.72%. Logistic Regression có hiệu suất thấp hơn với accuracy 82.99%, precision 85.48%, recall 79.49% và F1-score 82.38%.

Tính diễn giải: Decision Tree là mô hình dễ diễn giải, cho phép ta hiểu rõ các quyết định dựa trên các đặc trưng. Điều này có ý nghĩa trong việc tìm hiểu tại sao mô hình đưa ra các dự đoán cụ thể.

Đóng góp và Kết quả đạt được:

Trong quá trình nghiên cứu, tôi đã tìm hiểu và áp dụng các mô hình machine learning phổ biến để dự đoán hành vi mua sắm của khách hàng. Kết quả thực nghiệm cho thấy Random Forest có hiệu suất tốt nhất trong bộ ba mô hình được thử nghiệm.

Tôi đã tiến hành phân tích sâu hơn về các chỉ số đánh giá mô hình, giúp hiểu rõ hơn về khả năng dự đoán và tìm ra các trường hợp positive của từng mô hình.

2. Kiến nghị và hướng phát triển

Trong tương lai, có thể thử nghiệm các mô hình machine learning khác, như Gradient Boosting và Support Vector Machines, để xem liệu có thể cải thiện hiệu suất so với các mô hình đã thử nghiệm.

Tăng cường xử lý dữ liệu trước khi đưa vào mô hình, bao gồm việc loại bỏ nhiễu và chuẩn hóa đặc trưng.

Nghiên cứu thêm về tối ưu hóa siêu tham số của các mô hình, giúp tăng cường hiệu suất dự đoán.

Mở rộng phạm vi nghiên cứu để xem xét các yếu tố bổ sung như thông tin địa lý, sự kiện đặc biệt, và thuộc tính khác để cải thiện dự đoán hành vi mua sắm.

3. Tổng kết

Nghiên cứu này đã giúp hiểu rõ hơn về việc ứng dụng các mô hình machine learning để dự đoán hành vi mua sắm của khách hàng. Kết quả cho thấy Random Forest là lựa chọn hiệu suất tốt nhất trong số các mô hình thử nghiệm. Tuy nhiên, việc lựa chọn mô hình cần dựa trên mục tiêu cụ thể và yếu tố thực tiễn của từng dự án. Công trình này cũng mở ra những cơ hội phát triển trong tương lai để cải thiện hiệu suất dự đoán và khám phá sâu hơn về hành vi mua sắm của khách hàng.

LINK TIỂU LUẬN (CODE + BÁO CÁO)

Google Drive:

https://drive.google.com/drive/folders/1DN3FWYx7gxyduSiT55t4iQBMMu-EJxfC?usp=sharing

TÀI LIỆU THAM KHẢO

[1] https://core.ac.uk/download/pdf/328025049.pdf

[2] https://haralick.org/DV/Handbook_of_Data_Visualization.pdf

[3] https://www.javatpoint.com/logistic-regression-in-machine-learning

[4] https://machinelearningcoban.com/tabml_book/ch_model/decision_tree.html

[5] https://machinelearningcoban.com/tabml_book/ch_model/random_forest.html