**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC CMC**

**BÁO CÁO BÀI TẬP LỚN**

HỌC PHẦN: TRÍ TUỆ NHÂN TẠO

Tên đề tài : **ỨNG DỤNG HỌC MÁY TRONG DỰ ĐOÁN KHÁCH HÀNG HỦY ĐẶT PHÒNG**

Nhóm sinh viên: Nguyễn Văn Thiện

Lê Minh Hiếu

Trần Mai Trung

**HÀ NỘI, tháng 5 – 2024**

**TÓM TẮT ĐỀ TÀI**

Học máy là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống "học" tự động từ dữ liệu để giải quyết những vấn đề cụ thể. Việc sử dụng học máy có thể giúp nghiên cứu các quy luật khách quan của dữ liệu dựa trên mối quan hệ giữa chúng nhằm mục đích dự đoán hoặc phân loại dữ liệu. Gần đây, Học máy được nghiên cứu, ứng dụng rộng rãi trong nhiều lĩnh vực sản xuất cũng như kinh doanh giúp giải quyết được nhiều bài toán thực tế trước đây chưa có lời giải.

Hiện tại trên thế giới nói chung, Việt Nam nói riêng việc quản lý đặt phòng và hủy đặt phòng khách sạn đang là vấn đề khó khăn cho các chủ khách sạn, khu nghỉ dưỡng. Việc một khách hàng hoặc một nhóm khách hàng huỷ đặt phòng một cách đột ngột sẽ đem lại thiệt hại lớn cho nhà kinh doanh đặc biệt trong mùa cao điểm du lịch. Bài toán “Ứng dụng học máy trong dự đoán khách hàng hủy đặt phòng” sẽ dựa trên các thông tin như ID khách hàng, khách đã từng lưu trú ở khách sạn vào cuối tuần, trong tuần, … dự đoán khả năng liệu khách hàng đó có hủy đặt phòng hay không? Từ đó giúp chủ khách sạn có những chiến lược kinh doanh phù hợp để sẵn sàng ứng phó nếu trường hợp này xảy ra.

Để thực hiện bài toán này, em sử dụng ngôn ngữ lập trình Python, các thư viện hỗ trợ trực quan hoá dữ liệu, cài đặt một số mô hình học máy giải quyết bài toán.

**CÁC MỤC TIÊU CHÍNH**

* Tìm hiểu bài toán, phân tích bộ dữ liệu, trực quan hóa dữ liệu.
* Tìm hiểu và cài đặt một số thuật toán Học máy để thực hiện bài toán.
* So sánh kết quả giữa các phương pháp học máy, đánh giá, xem xét tính hiệu quả giữa các thuật toán học máy.

**KẾT QUẢ DỰ KIẾN**

* Lựa chọn được mô hình học máy phù hợp cho bài toán đề ra.
* Xây dựng được trang Web đơn giản để nhập dữ liệu và chạy chương trình.
* Hoàn thiện báo cáo

MỤC LỤC

[DANH MỤC CÁC HÌNH ẢNH ii](#_Toc166619365)

[DANH MỤC BẢNG BIỂU iii](#_Toc166619366)

[DANH MỤC CÁC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ iv](#_Toc166619367)

[BỐ CỤC NỘI DUNG v](#_Toc166619368)

[2.1.1 Mở đầu về học máy 3](#_Toc166619369)

[2.1.2 Một số thuật toán học máy 8](#_Toc166619370)

[2.1.3 Các phương pháp đánh giá mô hình 20](#_Toc166619371)

[2.2.1 Xử lý dữ liệu khuyết 27](#_Toc166619372)

[2.2.2 Chuẩn hóa dữ liệu 29](#_Toc166619373)

[3.1.1 Các thư viện hỗ trợ 31](#_Toc166619374)

[3.2.1 WSGI 34](#_Toc166619375)

[3.2.2 Werkzeug 35](#_Toc166619376)

[3.2.3 Jinja2 35](#_Toc166619377)

[3.3.1 Mô tả bộ dữ liệu 37](#_Toc166619378)

[3.3.2 Trực quan hóa dữ liệu 39](#_Toc166619379)

[3.4.1 Tiền xử lý dữ liệu 46](#_Toc166619380)

[3.4.2 Chia tập dữ liệu 49](#_Toc166619381)

[3.4.3 Xây dựng và huấn luyện mô hình học máy 49](#_Toc166619382)

[TÀI LIỆU THAM KHẢO 64](#_Toc166619383)

DANH MỤC CÁC HÌNH ẢNH

[Hình 2.1 Mô hình học có giám sát – Supervised Learning 6](#_Toc138510352)

[Hình 2.2 Mô hình học không giám sát – Unsupervised Learning 7](#_Toc138510353)

[Hình 2.3 Ví dụ về Data trong Học máy 9](#_Toc138510354)

[Hình 2.4 Ví dụ về cây quyết định 12](#_Toc138510355)

[Hình 2.5 Đồ thị entropy 14](#_Toc138510356)

[Hình 2.6 Mô hình Randomforest 16](#_Toc138510357)

[Hình 2.7 Quá trình đưa ra quyết định của RF 17](#_Toc138510358)

[Hình 2.8 Minh họa cho quá trình Logistic Regression đưa ra phân loại 19](#_Toc138510359)

[Hình 2.9 Support vector machine 20](#_Toc138510360)

[Hình 2.10 Ví dụ dữ liệu gồm hai lớp (xanh lục và lam) 21](#_Toc138510361)

[Hình 2.11 Các ranh giới có thể phân loại hai lớp dữ liệu 21](#_Toc138510362)

[Hình 2.12 Siêu phẳng tối ưu nhất 22](#_Toc138510363)

[Hình 2.13 Biểu diễn Sstotal 25](#_Toc138510364)

[Hình 2.14 Biểu diễn Ssres 25](#_Toc138510365)

[Hình 2.15 Tiền xử lý dữ liệu cho học máy 29](#_Toc138510366)

[Hình 3.1: Ngôn ngữ lập trình python 34](#_Toc138510367)

[Hình 3.2: Một vài biểu đồ được tạo bởi seaborn 38](#_Toc138510368)

[Hình 3.3: Các điểm mạnh của Python Flask 40](#_Toc138510369)

[Hình 3.4: Cấu trúc của dữ liệu 44](#_Toc138510370)

[Hình 3.5: Phân bố chất trạng thái hủy đặt phòng 46](#_Toc138510371)

[Hình 3.6 Heatmap biểu hiện mức độ tương quan giữa các thuộc tính 47](#_Toc138510372)

[Hình 3.7 Phân phối mức giá trung bình của phòng khách sạn 48](#_Toc138510373)

[Hình 3.8 Phân phối giá phòng đối với booking\_status 48](#_Toc138510374)

[Hình 3.9 Tỉ lệ phân bố các phương thức đặt phòng 49](#_Toc138510375)

[Hình 3.10 Mối quan hệ giữa phương thức đặt phòng và trạng thái hủy đặt phòng 50](#_Toc138510376)

[Hình 3.11 Mối quan hệ giữa khách quen và trạng thái hủy đặt phòng hay không 51](#_Toc138510377)

[Hình 3.12 Các bước giải quyết bài toán 51](#_Toc138510378)

[Hình 3.13 Kiểm tra kiểu data của các trường thông tin 53](#_Toc138510379)

[Hình 3.14 Tìm tham số max\_depth tối ưu cho DT 57](#_Toc138510380)

[Hình 3.15 Tìm tham số n\_estimators tối ưu cho RF 59](#_Toc138510381)

[Hình 4.1: Accuracy của các mô hình học máy 64](#_Toc138510382)

[Hình 4.2 Report classification của LR 64](#_Toc138510383)

[Hình 4.3 Report classification của DT 65](#_Toc138510384)

[Hình 4.4 Report classification của RF 65](#_Toc138510385)

[Hình 4.5 Report classification của SVM 65](#_Toc138510386)

[Hình 4.6 Giao diện chính ban đầu của web 66](#_Toc138510387)

[Hình 4.7 Web trả về kết quả dự đoán khách hàng 67](#_Toc138510388)

DANH MỤC BẢNG BIỂU

[Bảng 2.1 Ví dụ về confusion matrix 26](#_Toc138510298)

[Bảng 3.1: Các trường thông tin trong tập dữ liệu 44](#_Toc138510299)

[Bảng 4.1: Độ chính xác của LR với tiêu chí accuracy 61](#_Toc138510300)

[Bảng 4.2: Độ chính xác của DT với tiêu chí accuracy 61](#_Toc138510301)

[Bảng 4.3: Độ chính xác của RF với tiêu chí accuracy 62](#_Toc138510302)

[Bảng 4.4 Độ chính xác của SVM với tiêu chí accuracy 63](#_Toc138510303)

[Bảng 4.5: Kết quả trung bình phân lớp của bốn thuật toán 64](#_Toc138510304)

DANH MỤC CÁC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Thuật ngữ** | **Giải thích** |
| DT | Decision Tree | Thuật toán học máy cây quyết định |
| FN | False negative | Các tập âm tính giả |
| FP | False positive | Các tập dương tính giả |
| LR | Logistic Regression | Thuật toán học máy hồi quy lôgic |
| MSE | Mean Squared Error | Sai số toàn phương trung bình |
| RF | Random Forest | Thuật toán học máy rừng ngẫu nhiên |
| RMSE | Root Mean Squared Error | Lỗi trung bình bình phương gốc |
| TN | True negative | Các tập âm tính thật |
| TP | True positive | Các tập dương tính thật |
| SVM | Support Vector Machine | Thuật toán giám sát |
| WSGI | Web Server Gateway Interface | Giao diện cổng vào máy chủ web |

BỐ CỤC NỘI DUNG

**Chương 1**: Giới thiệu tổng quan về bài toán

Tổng quan về nội dung nghiên cứu, mục tiêu đề tài, phương pháp nghiên cứu

**Chương 2**: Cơ sở lý thuyết

Trình bày về các thuật toán cơ sở, các phương pháp đánh giá mô hình, các mô hình học máy Decision Tree, Random Forest, Logistic Regression và Support Vector Machine. Lý thuyết cơ sở về xây dựng web bằng Flask.

**Chương 3**: Ứng dụng phương pháp và xây dựng mô hình, đánh giá kết quả mô hình

Trình bày các công cụ sử dụng và tiến hành xây dựng mô hình, đánh giá kết quả mô hình, từ đó có thể chọn ra mô hình phù hợp với bài toán để xây dựng web.

**Chương 4**: Kết luận

# GIỚI THIỆU

## Giới thiệu tổng quan

Hoạt động kinh doanh khách sạn đã thực sự trở thành một ngành kinh doanh ở Việt Nam vào những năm đầu của thập niên 90. Kinh doanh khách sạn là một trong những lĩnh vực kinh doanh thu hút đông đảo các nhà đầu tư. Đây là một ngành kinh doanh mang tính cạnh tranh cao, tuy nhiên nếu quản lý tốt, doanh nghiệp khách sạn có thể đạt được lợi nhuận cao và ổn định.

## Mục tiêu nghiên cứu

Mục tiêu chính của bài tập lớn là đề xuất các phương pháp tiếp cận phù hợp cho bài toán quản lý phòng khách sạn, nhằm nâng cao độ chính xác mô hình phân lớp. Cụ thể là qua quá trình tìm hiểu, em đã lựa chọn được ba phương pháp tương đối phù hợp cho bài toán này, đó là Decision Tree, Random Forest và Logistic Regression. Do đó, mục tiêu đề ra của bài tập lớn là:

* Hiểu về phương pháp Random Forest, Logistic Regression và Decision Tree .
* Phân tích tổng quan về bộ dữ liệu, bài toán phân lớp và trích chọn đặc trưng.
* Thiết kế thực nghiệm trên Python xử lý dữ liệu và xây dựng mô hình
* So sánh kết quả thu được từ ba phương pháp.
* Chọn ra phương pháp phù hợp nhất để xây dựng web cơ bản giúp đánh giá khách hàng có hủy phòng hay không

## Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu của đề tài là bộ dữ liệu bao gồm các trường thông tin hữu ích cho việc dự đoán khách hàng có hủy phòng hay không như số người lớn, số trẻ em hay thời gian đặt phòng, …. Bộ dữ liệu này được thu thập thủ công từ các thực nghiệm. Từ đó giúp chúng ta có thể xây dựng các mô hình học máy áp dụng cho bài toán quản lý khách sạn.

Phạm vi nghiên cứu của đồ án tốt nghiệp:

* Tìm hiểu về bộ dữ liệu
* Tìm hiểu về phương pháp Decision Tree, Random Forest và Logistic Regression
* Chạy thực nghiệm với Python sử dụng máy tính cá nhân.
* Xây dựng web cơ bản sử dụng framework Flask

## Phương pháp nghiên cứu

Phương pháp nghiên cứu khi thực hiện bài tập lớn là tìm hiểu cơ sở lý thuyết chung về các phương pháp học máy, cụ thể là ba phương pháp Decision Tree, Random Forest và Logistic Regression, lựa chọn các thuộc tính trong bài toán phân loại. Nhằm chứng minh tính hiệu quả của các mô hình học máy ứng dụng vào bài toán quản lý phòng khách sạn, đồ án tốt nghiệp sẽ thực nghiệm trên bộ dữ liệu được lấy từ thực tiễn.

# CƠ SỞ LÝ THUYẾT

## Học máy

### Mở đầu về học máy

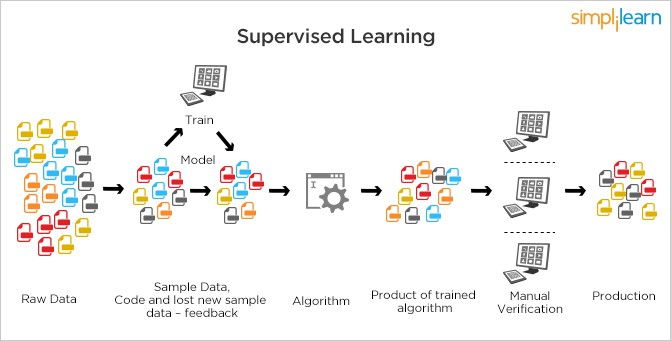
#### Giới thiệu

Machine Learning đã trở thành xu hướng công nghệ đang được quan tâm trên toàn thế giới trong gần một thập kỷ qua. Trong khoa học, có hàng ngàn bài báo được đăng về chủ đề này mỗi năm. Các công ty lớn như Google, Facebook, Microsoft và các công ty đang khởi nghiệp đều đang đầu tư vào Machine Learning. Đến nay, đã có rất nhiều ứng dụng sử dụng Machine Learning trong các lĩnh vực của cuộc sống, từ khoa học máy tính đến các ngành ít liên quan hơn như vật lý, hóa học, y học, chính trị. Ví dụ như xe tự hành của Google và Tesla, hệ thống tự tag khuôn mặt trong ảnh của Facebook, trợ lý ảo Siri của Apple, hệ thống gợi ý sản phẩm của Amazon, hệ thống gợi ý phim của Netflix, máy chơi cờ vây AlphaGo của Google DeepMind đều là các ví dụ rõ ràng cho sức mạnh của machine learning so với các phương pháp cổ điển.

#### Một số khái niệm trong học máy

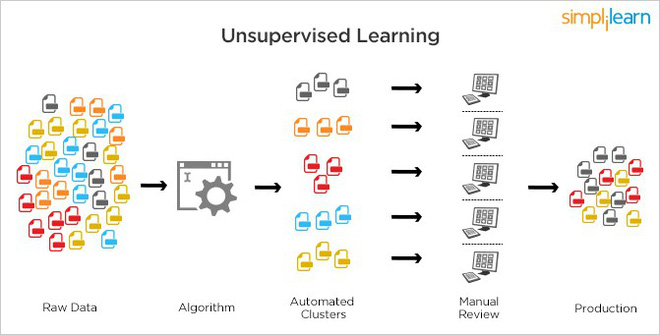
Trong học máy có một số khái niệm như sau:

***Học có giám sát*** (Supervised Learning): Học có giám sát (supervised learning) là một khía cạnh quan trọng trong học máy và được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau. Trong học có giám sát, máy tính sẽ được cung cấp với một tập dữ liệu đã được giám sát, trong đó mỗi mẫu dữ liệu được gắn với một nhãn cụ thể. Máy tính sẽ học từ tập dữ liệu này và xây dựng mô hình dự báo nhãn cho các mẫu dữ liệu mới.



Hình . Mô hình học có giám sát – Supervised Learning

***Học không giám sát*** (Unsupervised Learning): Học không giám sát (unsupervised learning) là một khía cạnh quan trọng trong học máy và được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau. Trong học không giám sát, máy tính không được cung cấp với bất kỳ nhãn nào cho các mẫu dữ liệu và phải tự động phát hiện các quan hệ và đặc trưng của dữ liệu.



Hình . Mô hình học không giám sát – Unsupervised Learning

Học không giám sát có rất nhiều ứng dụng thực tế, bao gồm tìm nhóm dữ liệu giống nhau, khám phá đặc trưng của dữ liệu và nhiều ứng dụng khác. Nó được sử dụng rộng rãi trong lĩnh vực công nghệ thông tin để phân tích dữ liệu người dùng và trong ngành y tế để tìm ra đặc điểm của bệnh tật.

Trong học không giám sát, có hai loại mô hình chính được sử dụng: phân cụm và khảo sát đặc trưng. Mô hình phân cụm được dùng để tìm các nhóm tương tự trong dữ liệu, trong khi mô hình khảo sát đặc trưng được dùng để tìm các đặc trưng của dữ liệu.

Khi chúng ta chỉ có dataset gồm đặc trưng mà không được gán nhãn. Nhiệm vụ của học không giám sát là phát hiện ra các thông tin hữu ích từ tập dữ liệu chưa được phân lớp và chưa được gán nhãn. Có thể là tìm ra sự giống nhau hay khác nhau giữa các tập đối tượng để nhóm chúng lại với nhau. Ví dụ cho học không giám sát:

* Có hai loại âm thanh, chúng ta cần tách hai loại đó ra.
* Có một tập các tin tức, phân ra các loại khác nhau (mặc dù ta chưa biết chính xác có bao nhiêu loại khác nhau)

Học không giám sát khó và phức tạp hơn Học có giám sát. Trong thực tế, có rất nhiều bài toán cần nghiên cứu và phát triển của học không giám sát, tuy nhiên ứng dụng thực tế của Học không giám sát đang rất khiêm tốn.

***Dữ liệu*** (Data): Dữ liệu là một yếu tố quan trọng trong học máy và có khả năng định hình được một mô hình học máy như thế nào. Dữ liệu có thể được sử dụng để huấn luyện (training), đánh giá (validation) và kiểm tra (testing) các mô hình học máy.

Dữ liệu trong học máy có thể được chia thành hai loại chính: dữ liệu giám sát (supervised data) và dữ liệu không giám sát (unsupervised data). Dữ liệu giám sát là dữ liệu mà mỗi mẫu được gắn với một nhãn cụ thể và được sử dụng trong học có giám sát, trong khi dữ liệu không giám sát là dữ liệu mà không có nhãn và được sử dụng trong học không giám sát.

Để có được dữ liệu tốt cho học máy, người dùng cần đảm bảo rằng dữ liệu đầu vào là đầy đủ, chính xác và liên quan đến vấn đề đang được giải quyết. Người dùng cũng cần phải chú ý đến việc lựa chọn tập dữ liệu phù hợp và định dạng dữ liệu đúng cách.

Dữ liệu cũng cần được định dạng đúng cách trước khi được sử dụng trong học máy. Điều này có nghĩa là người dùng cần chuẩn hóa các giá trị dữ liệu và chuyển đổi chúng thành dạng số nếu cần thiết. Người dùng cũng cần phải chú ý đến việc xử lý các giá trị thiếu và các giá trị bất thường (outliers).

Cuối cùng, người sử dụng cần chia dữ liệu thành hai phần chính: tập huấn luyện, tập và tập kiểm tra. Tập huấn luyện được sử dụng để xây dựng mô hình học máy, tập đánh giá được sử dụng để đánh giá hiệu quả của mô hình và tập kiểm tra được sử dụng để xác nhận hiệu quả của mô hình trên dữ liệu mới.

Học máy là một tiến trình cho phép máy tính tự động học và tiếp cận thông tin từ dữ liệu, tức là học từ kinh nghiệm. Vì vậy, việc khám phá dữ liệu là một bước quan trọng để hiểu rõ hơn về mô hình và cách tốt nhất để sử dụng chúng. Trong bài báo cáo này, dữ liệu được biểu diễn dưới dạng một bảng với nhiều hàng và cột, đó là một trong những cấu trúc dữ liệu cơ bản trong học máy.

***Thể hiện/ Dữ kiện*** (instance): Là một hàng trong bảng dữ liệu hay là một mẫu dữ liệu trong tập dữ liệu của một bài toán học máy. Mỗi instance được biểu diễn bởi một tập hợp các thuộc tính (còn gọi là features) và một nhãn (label).

Instance có thể được biểu diễn bằng các dạng dữ liệu khác nhau như số, chuỗi, ảnh, âm thanh hoặc video. Một số bài toán học máy yêu cầu người dùng phải định dạng lại dữ liệu đầu vào thành một định dạng cụ thể trước khi sử dụng trong mô hình học máy.

Instance là một khái niệm quan trọng trong lĩnh vực học máy vì nó là cơ sở để mô hình học máy học từ dữ liệu và tạo ra các kết quả dự báo chính xác. Mô hình học máy sẽ sử dụng các instance trong tập training để học các quan hệ giữa các thuộc tính.

***Đặc trưng*** (feature): là một đặc trưng của một instance trong tập dữ liệu học máy. Mỗi instance được biểu diễn bởi một tập hợp các feature và một nhãn (label), được hiểu đơn giản là một cột trong bảng dữ liệu. Nó là một thành phần của một dữ kiện. Một số đặc trưng là dữ liệu quan sát được và một số khác cần được dự đoán.

***Kiểu dữ liệu*** (data type): Mỗi đặc trưng trong dữ liệu đều có một kiểu dữ liệu xác định, bao gồm kiểu số nguyên, số thực, hoặc rời rạc. Trong thực tế, còn có các kiểu dữ liệu khác như ngày, tháng, chuỗi kí tự hoặc kiểu phức tạp khác, tuy nhiên, chúng sẽ được chuyển sang kiểu số nguyên, số thực hoặc rời rạc khi sử dụng các thuật toán học máy.

Việc lựa chọn data type phù hợp là rất quan trọng trong học máy vì nó có thể ảnh hưởng đến hiệu quả của mô hình học máy. Ví dụ, nếu một feature dữ liệu là số nguyên nhưng được lưu trữ dưới dạng chuỗi, mô hình học máy sẽ không thể tính toán được các phép toán số học trên feature này.

Việc chuyển đổi data type có thể được thực hiện bằng cách sử dụng các hàm của ngôn ngữ lập trình hoặc các thư viện hỗ trợ. Ví dụ, trong Python, người dùng có thể sử dụng hàm int() để chuyển một chuỗi thành số nguyên, hàm float() để chuyển một chuỗi thành số thực và hàm str() để chuyển một số thành chuỗi.

***Tập dữ liệu*** (Datasets): là một tập hợp các thể hiện/dữ kiện. Chúng có thể được sử dụng cho các mục đích khác nhau, như huấn luyện mô hình học máy, đánh giá hiệu năng của mô hình, hoặc kiểm tra hiệu năng của mô hình trên dữ liệu mới.

***Tập dữ liệu huấn luyện*** (Training dataset): là tập dữ liệu được sử dụng để huấn luyện mô hình máy học. Tập dữ liệu này cung cấp các ví dụ cần thiết cho mô hình để học các quan hệ giữa các đặc trưng và nhãn của dữ liệu. Nó là một thành phần không thể thiếu trong quá trình xây dựng mô hình học máy, vì nó giúp cho mô hình học từ dữ liệu.

Training dataset phải đạt đủ mức độ đa dạng để giúp mô hình học có khả năng generalize tốt cho các dữ liệu mới. Nếu training dataset không đủ đa dạng, mô hình có thể dễ dàng bị overfitting (quá khớp) và không có khả năng generalize tốt cho các dữ liệu mới.

Training dataset cũng phải được chọn một cách cân bằng giữa các nhãn, tránh trường hợp một số nhãn quá ít hoặc quá nhiều so với các nhãn khác. Nếu số lượng các mẫu dữ liệu của một nhãn quá ít, mô hình có thể không có khả năng học được các quan hệ giữa các đặc trưng và nhãn cho nhãn đó.

Để chọn một tập training dataset tốt, người dùng có thể thực hiện các bước sau:

* Tìm hiểu về bài toán: Trước khi chọn training dataset, người dùng nên tìm hiểu kỹ về bài toán mà mình muốn giải quyết, để có thể chọn được tập dữ liệu phù hợp.
* Chọn nguồn dữ liệu: Người dùng có thể chọn dữ liệu từ các nguồn khác nhau như các cơ sở dữ liệu, các tập dữ liệu mở hoặc tạo ra từ scratch.
* Đối chiếu với yêu cầu của bài toán: Người dùng nên đối chiếu tập dữ liệu với yêu cầu của bài toán để xác định xem tập dữ liệu có đủ đa dạng hay không.
* Đánh giá tính toàn vẹn của tập dữ liệu: Người dùng cần đánh giá xem tập dữ liệu có chứa đủ số lượng và loại dữ liệu khác nhau hay không.

***Tập dữ liệu kiểm tra*** (Testing dataset): Testing dataset là tập dữ liệu được sử dụng để đánh giá hiệu quả của mô hình học máy sau khi đã được huấn luyện (train) bằng tập dữ liệu khác (thường là tập training dataset). Testing dataset được sử dụng để xác định xem mô hình có khả năng generalize tốt cho các dữ liệu mới hay không.

Testing dataset phải đạt đủ mức độ đa dạng để giúp mô hình được đánh giá một cách chính xác về khả năng generalize. Nếu testing dataset không đủ đa dạng, mô hình có thể dễ dàng bị underfitting (quá ít khớp) và không được đánh giá một cách chính xác về khả năng generalize.

Tập dữ liệu huấn luyện có tác dụng tương tự như bài tập cho học sinh, còn tập dữ liệu kiểm tra tương tự như bài thi. Như một giáo viên giỏi, tập dữ liệu huấn luyện giúp mô hình học các quan hệ giữa các đặc trưng và nhãn của dữ liệu. Tập dữ liệu kiểm tra giúp kiểm tra hiệu năng của mô hình trên dữ liệu mới. Trong quá trình xây dựng mô hình học máy, chúng ta cần tách tập dữ liệu ban đầu ra thành các tập dữ liệu huấn luyện và kiểm tra.

Mục tiêu của một mô hình học máy (có giám sát) là xây dựng hàm f ánh xạ từ dữ kiện quan sát X(feature) tới nhãn y(label) (biểu thức 1):

|  |  |
| --- | --- |
|  | 2.1 |

### Một số thuật toán học máy

Trong thực tế, có rất nhiều phương pháp phân loại dữ liệu, mỗi phương pháp có những đặc điểm riêng và phù hợp với từng loại dữ liệu phân loại. Qua việc tìm hiểu những bài báo liên quan, em đã lựa chọn được những phương pháp phù hợp với dạng bài toán này, chính là Decision Tree, Random Forest, Logistic Regression và Support Vector Machine

#### Decision Tree

Cây quyết định là một thuật toán Học máy có giám sát dùng cho cả bài toán phân lớp (classification) và hồi quy (regression). Nó là một phương pháp phân loại rất hiệu quả và được sử dụng rộng rãi bởi nó dễ hiểu và có thể giải thích chính xác cách thức mô hình đưa ra kết luận. Thông qua các nhánh cây, chúng ta có thể dễ dàng theo dõi quá trình học và đưa ra dự đoán, chính xác với một chuỗi các câu lệnh if-then.

Cây quyết định là một cấu trúc dạng cây trong đó mỗi nút bên trong đại diện cho một câu hỏi hoặc kiểm tra tính năng, mỗi nhánh đại diện cho một câu trả lời cho câu hỏi đó và mỗi nút lá đại diện cho một nhãn lớp. Các quyết định được đưa ra bằng cách duyệt cây từ gốc đến lá, đi theo một đường được xác định bởi các câu trả lời cho các câu hỏi tại mỗi nút bên trong. Mỗi đường dẫn từ gốc đến lá đại diện cho một quy tắc phân loại.

Diagram

Description automatically generated

Hình . Ví dụ về cây quyết định

Mỗi một quyết định là một đường đi từ gốc tới lá, Độ dài của đường đi này được xác định bởi số các nhánh cây. Các nhánh này chính là kết quả của việc kiểm tra các đặc trưng (features). Như vậy, thứ tự các đặc trưng để kiểm tra sẽ tác động tới cấu trúc cụ thể của cây quyết định. Nói một cách khác, có hai yếu tố quan trọng trong quá trình xây dựng mô hình:

1. Đặc trưng nào được đưa ra kiểm tra?
2. Trật tự đặc trưng kiểm tra như thế nào?

Ta cũng cần lưu ý rằng, ngoài trật tự của các đặc trưng đem ra kiểm tra sẽ làm thay đổi hoàn toàn cây, thì có nhiều đặc trưng không có tác động tới quá trình xây dựng cây. Ta gọi chúng là những đặc trưng dư thừa.

Trong quá trình xây dựng cây quyết định, tìm kiếm các câu hỏi (đặc trưng) để dự đoán các thông tin liên quan. Các câu hỏi có thể là yes/no và có thể cho thông tin tốt hoặc không cho thông tin tốt. Ví dụ, nếu một câu hỏi yes/no có kết quả luôn chính xác khi trả lời và sai khi trả lời là "no", thì đây là một câu hỏi hay vì nó cho thông tin tốt. Để đánh giá "lượng thông tin" của các câu hỏi, người ta sử dụng entropy được giới thiệu bởi nhà toán học Shannon. Entropy dùng để đánh giá mức độ không chắc chắn của dữ liệu.

Giả sử chúng ta có tập dữ liệu S, mỗi phần tử trong S được gán nhãn vào một trong các lớp C1, C2, ..., Cn. Nếu tất cả các phần tử trong S chỉ thuộc vào một lớp Ci (1 < i <n) nào đó, thì mọi thứ sẽ được xác định rõ ràng, tức là không có sự không chắc chắn nữa; điều này sẽ thể hiện độ entropy sẽ thấp. Ngược lại, nếu các phần tử trong S mà trải khắp các lớp Ci, thì sẽ có sự không chắc chắn, tức là độ entropy sẽ cao.

Dưới góc nhìn toán học, nếu coi pi là tần suất dữ liệu được gán nhãn Ci trong tập S, ta tính:

pi = tần suất (Ci) = số lần xuất hiện (Ci) / tổng số hàng(S)

Sau đó, ta có công thức entropy như sau:

|  |
| --- |
| H(S) = -∑pi \* log*2*(pi) 2.2 |

Với quy ước 0\*log(0) = 0

Thông qua hình 2.3, chúng ta có thể nhận thấy entropy không âm, và nó tiến tới 0 khi pi tiến tới 0 hoặc 1. Điều này có nghĩa là entropy sẽ nhỏ khi pi tiến tới 0 hoặc 1 (khi hầu hết các điểm dữ liệu cùng thuộc vào một lớp). Và entropy sẽ lớn khi pi không gần 0 hoặc 1 (khi hầu hết các điểm dữ liệu trải đều ra lớp).

Để dựng cây quyết định, thuật toán cần thực hiện các bước sau đây:

Chart, line chart

Description automatically generated

Hình . Đồ thị entropy

***Bước 1****:* Tính entropy cho toàn bộ dữ liệu ban đầu, để xem đặc trưng nào tốt. Bằng cách chia tập dữ liệu dựa trên các giá trị có thể của mỗi thuộc tính và tính entropy trên hai tập dữ liệu con thu được. Để xác định thuộc tính nào tốt đề chọn, thuật toán cần tính độ lợi thông tin (Information Gain): IG(parent, children) = entropyparent — (entropychild1 X propchild1 entropychild2 X propchild2). Trong đó propchildi là tỉ lệ của các dữ liệu thuộc vào tập dữ liệu con i. Tính IG cho mọi đặc trưng và chọn đặc trưng nào cho IG cao nhất. Sau khi chọn được đặc trưng để tách tập dữ liệu ban đầu (gốc cây) thành hai tập con (2 nhánh cây) tương ứng với điều kiện True hay False.

***Bước 2****:* Với hai nhánh con mới sinh ra, chúng ta sẽ lần lượt kiểm tra xem mỗi nhánh đã đủ điều kiện là lá (1eaf) hay chưa. Nếu chưa, chúng ta tiền hành thuật toán như đã làm với nút gốc (Lặp lại Bước 1).

***Bước 3:*** Thuật toán sẽ dừng lại khi mọi nhánh đều có lá.

#### Random forest

Cây quyết định là một thuật toán máy học có thể giúp người ta dự đoán kết quả dựa trên các điều kiện cho trước. Tuy nhiên, cây quyết định có thể bị phương sai cao (tức overfitting), trong trường hợp nó được huấn luyện quá tốt trên dữ liệu huấn luyện và không thể giải quyết tốt trên dữ liệu mới.

Một cách để khắc phục vấn đề này là dùng mô hình Random Forest. Random Forest xây dựng và kết hợp nhiều cây quyết định với yếu tố ngẫu nhiên (random) với nhau, trong đó khả năng overfitting thấp hơn vì mỗi cây quyết định được đào tạo hoàn hảo trên dữ liệu mẫu cụ thể và kết quả cuối cùng không phụ thuộc vào một cây quyết định mà là nhiều quyết định cây. Trong trường hợp bài toán phân loại, kết quả cuối cùng được tính bằng cách sử dụng bộ phân loại đa số, trong trường hợp bài toán hồi quy, đầu ra cuối cùng là giá trị trung bình của tất cả các đầu ra.

Random Forest là một thuật toán supervised learning, có thể giải quyết cả bài toán hồi quy và phân loại. Nó sử dụng một cách giống như phiên bầu (voting) cho các cây quyết định độc lập để dự đoán kết quả. RF coi mỗi cây quyết định như một cử tri bỏ phiếu độc lập. Kết quả cuối cùng là câu trả lời nhận được nhiều bầu chọn nhất từ các cây quyết định.

Random Forest thường được sử dụng với các bài toán phức tạp về phân loại và hồi quy và được coi là một trong những thuật toán có hiệu suất tốt nhất trong máy học.

Chart, radar chart

Description automatically generated

Hình . Mô hình Randomforest

Random forests là một thuật toán máy học được coi là chính xác và mạnh mẽ vì số lượng cây quyết định tham gia vào quá trình dự đoán. Nó không bị vấn đề overfitting vì nó tính trung bình của tất cả các dự đoán, trong đó hủy bỏ những dự đoán sai. Random forests còn có khả năng xử lý giá trị còn thiếu bằng cách sử dụng giá trị trung bình hoặc tính toán mức trung bình gần kề. Nó còn có thể chọn ra các tính năng quan trọng nhất giúp cho việc phân loại.

Tuy nhiên, Random forests chậm hơn so với cây quyết định trong việc dự đoán vì nó cần phải sử dụng nhiều cây quyết định để hoàn thành quá trình dự đoán. Tất cả các cây trong rừng phải đưa ra dự đoán cho cùng một đầu vào và sau đó thực hiện bỏ phiếu trên đó. Quá trình này tốn thời gian hơn so với cây quyết định, một thuật toán dễ hiểu hơn và cho phép dễ dàng đưa ra quyết định bằng cách đi theo đường dẫn trong cây.

Diagram

Description automatically generated

Hình . Quá trình đưa ra quyết định của RF

Random Forest hoạt động theo các bước:

* ***Bước 1****:* Chọn mẫu ngẫu nhiên từ tập đã cho
* ***Bước 2****:* Thiết lập cây quyết định cho từng mẫu và nhận kết quả dự đoán từ mỗi quyết định cây.
* ***Bước 3****:* Bỏ phiếu cho mỗi kết quả dự đoán
* ***Bước 4****:* Chọn kết quả được dự đoán nhiều nhất là dự đoán cuối cùng

Random Forest là một thuật toán sử dụng nhiều cây quyết định để dự đoán, tuy nhiên do mỗi cây quyết định trong thuật toán không sử dụng toàn bộ dữ liệu training và tất cả các thuộc tính của dữ liệu để xây dựng, nên mỗi cây có thể sẽ dự đoán không tốt và gây ra high bias (quá chọn đơn giản). Tuy nhiên, vì kết quả cuối cùng của thuật toán Random Forest được tổng hợp từ nhiều cây quyết định nên thông tin từ từng cây sẽ bổ sung thông tin cho nhau, giúp cho mô hình có lowbias và low variance, hay mô hình có kết quả dự đoán tốt hơn. Tổng hợp từ nhiều cây quyết định này có thể giúp tránh được vấn đề underfitting, giúp dự đoán chính xác hơn. Nó cũng giúp cho thuật toán tránh được overfitting vì nó dùng nhiều cây quyết định với dữ liệu training khác nhau để dự đoán, điều này làm cho kết quả cuối cùng của thuật toán trở nên chính xác và có ít sai số hơn.

#### Logistic regression

Logistic regression là một phương pháp học máy thuộc nhóm phân loại (classification), được sử dụng để dự đoán xác suất của một sự kiện xảy ra dựa trên các đặc trưng (features) đầu vào. Logistic regression được sử dụng rộng rãi trong nhiều lĩnh vực như y tế, kinh tế, marketing và công nghệ thông tin.

Phương pháp hồi quy logistic là mô hình hồi quy dùng để dự đoán các giá trị rời rạc của một biến đầu ra y dựa trên một vectơ đầu vào X. Điều này tương đương với phân loại các đầu vào X vào các nhóm y tương ứng.

Trong hồi quy tuyến tính chúng ta dựa vào một hàm hồi quy giả thuyết θ để dự báo biến mục tiêu liên tục y. Tạo ra tính phi tuyến cho phương trình hồi quy nhằm giúp nó có đường biên phân chia giữa hai nhóm tốt hơn.

Đầu ra dự đoán của logistic regression thường được viết chung dưới dạng:

|  |
| --- |
| 2.3 |

Trong đó :

* là logistic function
* là tập các hệ số
* là tập các biến độc lập khác nhau.

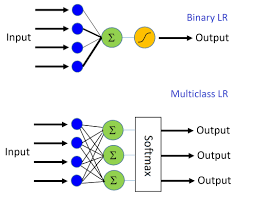
Đối với bài toán phân loại từ 3 loại trở lên, ta sử dụng θ là hàm Softmax σ

|  |
| --- |
| 2.4 |

Trong đó:

* là tập vector các biến độc lập
* là độ dài của

Đầu ra của Softmax là vector xác suất của mỗi lớp phân loại, ta lấy lớp có xác suất cao nhất để phân lớp cho đối tượng.



Hình . Minh họa cho quá trình Logistic Regression đưa ra phân loại

Trong một số trường hợp, logistic regression có thể không hoạt động tốt vì các đặc trưng đầu vào không đủ mạnh để phân loại sự kiện xảy ra hay không. Logistic regression cũng có một số hạn chế, bao gồm:

* Phụ thuộc vào các đặc trưng đầu vào: Mô hình logistic regression không hoạt động tốt khi các đặc trưng đầu vào không đủ mạnh để phân loại sự kiện xảy ra hay không.
* Không phù hợp cho các sự kiện có nhiều hơn hai nhãn: Logistic regression chỉ phù hợp cho các bài toán phân loại hai nhãn (binary classification). Nếu có nhiều hơn hai nhãn, người dùng có thể sử dụng các phương pháp khác như decision tree.
* Không phù hợp cho các bài toán hồi quy (regression): Logistic regression chỉ phù hợp cho các bài toán phân loại, không phù hợp cho các bài toán hồi quy. Nếu cần dự đoán một giá trị liên tục, người dùng có thể sử dụng các phương pháp hồi quy như linear regression.

#### Support vector machine

SVM là một thuật toán được giám sát mạnh mẽ, hoạt động tốt nhất trên các bộ dữ liệu nhỏ nhưng trên các bộ dữ liệu phức tạp. Support Vector Machine, viết tắt là SVM có thể được sử dụng cho cả nhiệm vụ hồi quy và phân loại, nhưng nhìn chung, chúng hoạt động tốt nhất trong các bài toán phân loại. Chúng rất nổi tiếng vào khoảng thời gian chúng được tạo ra vào những năm 1990, và tiếp tục là phương pháp tiếp cận cho một thuật toán hiệu suất cao với một chút điều chỉnh.

Về vấn đề học máy có giám sát, trong đó chúng ta cố gắng tìm một siêu phẳng phân tách tốt nhất hai lớp. Đừng nhầm lẫn giữa SVM và hồi quy logistic. Cả hai thuật toán đều cố gắng tìm siêu phẳng tốt nhất, nhưng sự khác biệt chính là hồi quy logistic là một cách tiếp cận xác suất trong khi SVM dựa trên các phương pháp thống kê.

SVM có các khái niệm sau:

* **Các Support Vectors**: Đây là những điểm gần siêu phẳng nhất. Một đường phân cách sẽ được xác định với sự trợ giúp của các điểm dữ liệu này.
* **Margin**: là khoảng cách giữa siêu phẳng và các quan sát gần siêu phẳng nhất (vector support). Trong SVM có khoảng margin lớn là được coi là tốt.



Hình . Support vector machine

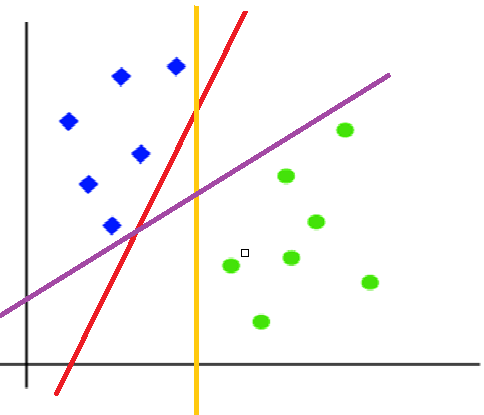
SVM được xác định sao cho nó chỉ được xác định theo support vector, chúng ta không phải lo lắng về các quan sát khác vì lề được tạo bằng cách sử dụng các điểm gần siêu phẳng nhất (vectơ hỗ trợ), trong khi trong hồi quy logistic, phân loại được xác định trên tất cả các điểm.

Giả sử chúng ta có một tập dữ liệu có hai lớp (xanh lục và xanh lam). Chúng ta muốn phân loại điểm dữ liệu mới là xanh lam hoặc xanh lục.



Hình . Ví dụ dữ liệu gồm hai lớp (xanh lục và lam)

Để phân loại các điểm này, chúng ta có thể có nhiều ranh giới quyết định, nhưng câu hỏi đặt ra là ranh giới nào là tốt nhất và chúng ta tìm ra nó như thế nào. Vì ta đang vẽ các điểm dữ liệu trong biểu đồ 2 chiều, ta gọi ranh giới quyết định này là một đường thẳng nhưng nếu ta có nhiều chiều hơn, ta gọi ranh giới quyết định này là “siêu phẳng”



Hình . Các ranh giới có thể phân loại hai lớp dữ liệu

Siêu phẳng tốt nhất là mặt phẳng có khoảng cách lớn nhất từ cả hai lớp và đây là mục tiêu chính của SVM. Điều này được thực hiện bằng cách tìm các siêu phẳng khác nhau phân loại nhãn theo cách tốt nhất, sau đó nó sẽ chọn siêu phẳng cách xa điểm dữ liệu nhất hoặc siêu phẳng có lề tối đa.



Hình . Siêu phẳng tối ưu nhất

Trong thuật toán SVM, chúng ta đang tìm cách tối đa hóa lề giữa các điểm dữ liệu và siêu phẳng. Hàm loss giúp tối ưu hóa là hinge loss như sau:

|  |  |
| --- | --- |
|  | 2.5 |

Trong đó:

* W là trọng số mô hình
* M là số mẫu dữ liệu
* y là giá trị thực
* là giá trị dự đoán
* X là vector giá trị đầu vào

Giá trị của mẫu dữ liệu x được tính như sau:

|  |  |
| --- | --- |
| + b | 2.6 |

Trong đó:

* w là trọng số mô hình
* là giá trị dự đoán
* x là vector một mẫu dữ liệu

### Các phương pháp đánh giá mô hình

Việc đánh giá mô hình Học máy là vô cùng quan trọng. Nó không chỉ là thước đo để so sánh sự phù hợp của các mô hình cho bài toán đang đặt ra mà còn ảnh hưởng đến quá trình hội tụ của các mô hình trong quá trình huấn luyện.

Trong học có Giám Sát (supervised learning), có hai loại công việc của mô hình học máy: Phân lớp (classification) và Hồi quy (regression). Trong công thức (2.1), giá trị đầu ra là liên tục với nhiệm vụ Hồi quy (regression), trong khi giá trị đầu ra là rời rạc với nhiệm vụ Phân lớp (classification). Mỗi một bài toán sẽ có phương pháp đánh giá mô hình khác nhau.

#### Các độ đo đánh giá trong mô hình hồi quy

Các chỉ số MSE, MAE, RMSE và R-Squared chủ yếu được sử dụng để đánh giá tỷ lệ lỗi dự đoán và hiệu suất mô hình trong bài toán hồi quy.

***MSE (Mean Squared Error)***

Trong thống kê, sai số bình phương trung bình (MSE) là một công cụ đo lường trung bình bình phương của chênh lệch giữa các giá trị dự đoán và giá trị thực trên dữ liệu. Nó được dùng để đo lường rủi ro (uncertainty) của các thủ tục ước tính số lượng không quan sát được. MSE tương đương với giá trị dự kiến của mất lỗi bình phương.

MSE có nghĩa là giá trị sai số bình phương trung bình hoặc là lỗi bình phương trung bình. Trong thông kê, khó để xác định rõ nguồn gốc của sai số - liệu nó là do mô hình hay là do ngẫu nhiên. Lỗi bình phương trung bình (MSE) được sử dụng để đo lường trung bình của chênh lệch bình phương giữa tham số dự đoán và tham số thực sự đã được quan sát được.

|  |  |
| --- | --- |
|  | 2.7 |

***RMSE (Root Mean Squared Error)***

Lỗi trung bình bình phương (RMSE) là một chỉ số thống kê được sử dụng để đo lường độ lệch chuẩn giữa giá trị dự đoán của mô hình và giá trị thực tế. Nó là kết quả của việc bình phương và cảnh báo các phần dư (lỗi dự đoán). RMSE được tính bằng cách tính giá trị bình phương của khoảng cách giữa các giá trị dự đoán và thực tế, rồi cảnh báo giá trị gốc của kết quả này. Nó được sử dụng rộng rãi trong các lĩnh vực như khí hậu học, dự báo và phân tích hồi quy để xác minh kết quả thí nghiệm. Giá trị RMSE càng nhỏ tức là sai số càng bé thì mức độ ước lượng cho thấy độ tin cậy của mô hình có thể đạt cao nhất. Nó có nghĩa là dự đoán của mô hình càng gần với giá trị thực tế thì chênh lệch giữa hai giá trị càng nhỏ, và do đó chính xác của mô hình càng cao. RMSE là một thước đo quan trọng trong xác định hiệu quả của một mô hình học máy và để so sánh với các mô hình khác trong cùng một bài toán.

Lỗi trung bình bình phương (RMSE) là một trong những chỉ số thường được sử dụng để đánh giá hiệu năng của mô hình học máy. Nó đo lường sai số bình phương trung bình giữa giá trị dự đoán và giá trị thực tế trên tập dữ liệu kiểm tra. Khi RMSE càng nhỏ, có nghĩa là mô hình càng chính xác trong việc dự đoán giá trị thực tế.

|  |  |
| --- | --- |
|  | 2.8 |

***R-squared***

R-squared là một đại lượng thống kê cho phép so sánh mức độ phù hợp của một mô hình hồi quy. Giá trị tối ưu của R-squared là 1, có nghĩa là mô hình đạt được tổng bình phương các phần dư nhỏ nhất và phù hợp với dữ liệu tốt nhất. R-squared được tính dựa trên tổng bình phương các phần dư (SSres) và tổng bình phương các độ lệch trung bình (SStotal).

Tổng bình phương các độ lệch trung bình là đo lường khoảng cách giữa các điểm dữ liệu và đường trung bình (hoặc giá trị trung bình). Việc giảm giá trị của tổng bình phương các độ lệch trung bình là một trong những cách để tăng giá trị R-squared, tức là tăng mức độ phù hợp của mô hình hồi quy.

Chart, scatter chart

Description automatically generated

Hình . Biểu diễn Sstotal

Tổng bình phương các phần dư là đo lường khoảng cách giữa các điểm dữ liệu và đường hồi quy tốt nhất (best fitted line) của mô hình. Nó cho thấy mức độ tương quan giữa các giá trị dự đoán và giá trị thực tế. Tổng bình phương các phần dư càng nhỏ thì mô hình hồi quy càng tốt.

Chart, scatter chart

Description automatically generated

Hình . Biểu diễn Ssres

Công thức tính R-squared

|  |
| --- |
| 2.9 |

#### Các độ đo đánh giá trong mô hình phân lớp

Để đánh giá nhiệm vụ phân lớp (classification), người ta thường dùng ma trận đánh giá (***Confusion matrix***).

Xét ví dụ thực tế: Tại bệnh viện A có 100 bệnh nhân đến khám một loại bệnh, giả sử biết trước trong 100 bệnh nhân có 60 người mắc bệnh, 40 người không có bệnh. Sau khi thăm khám, bệnh viện đưa ra kết quả:

* Trong 60 người bệnh thật thì có 45 người chuẩn đoán có bệnh, 15 người chuẩn đoán không mắc bệnh.
* Trong 40 người không mắc bệnh thì có 30 người chuẩn đoán không mắc bệnh, 10 người chuẩn đoán là mắc bệnh.

Bảng 2.1 Ví dụ về confusion matrix

|  |  |  |
| --- | --- | --- |
| **Dự đoán Thực tế** | **Dương tính** | **Âm tính** |
| **Dương tính** | **TP (45)** | FP (10) |
| **Âm tính** | FN (15) | **TN (30)** |

Từ ma trận cơ bản này, ta sẽ có một số thuật ngữ sau:

* ***True positive (TP):*** Số các ca dự đoán dương tính đúng hay dương tính thật.
* ***True negative (TN):*** Số các ca dự đoán âm tính đúng hay âm tính thật.
* ***False positive (FP):*** Số các ca dự đoán dương tính sai hay dương tính giả.
* ***False negative (FN):*** Số các ca dự đoán âm tính sai hay âm tính giả.

Nói khác đi, nếu chúng ta thấy chữ “True”, điều đó nghĩa là tất cả những gì chúng ta dự đoán đều đúng hết cả. Nếu chúng ta dự đoán 45 bệnh nhân có bệnh (TP) thì đúng là trong thực tế 45 bệnh nhân đó đang mang bệnh. Còn nếu chúng ta dự đoán có 30 bệnh nhân không mang bệnh (TN) thì trong thực tế đúng là họ đang rất khỏe mạnh.

Ngược lại, nếu ta thấy chữ “False” có nghĩa là những gì chúng ta dự đoán trật hết. Những bệnh nhân ta đoán là có bệnh thì lại không trong thực tế và ngược lại.

***Precision / Recall***

Với confusion matrix, chúng ta có thể tính được hai đại lượng quan trọng là Precision và Recall.

***Precision:*** là tỷ lệ số trường hợp dự đoán đúng trong số trường hợp dự đoán là có sự việc xảy ra. Nó cho thấy tính chính xác của mô hình trong việc dự đoán sự việc xảy ra.

2.10

***Recall*** (đôi khi còn được gọi là Sensitivity): là tỷ lệ số trường hợp dự đoán đúng trong số trường hợp thực sự xảy ra sự việc. Nó cho thấy tính tập trung của mô hình trong việc phát hiện sự việc xảy ra.

|  |
| --- |
| 2.11 |

***F1-score***

F1-score là một chỉ số kết hợp của Precision và Recall, nó tính toán tỷ lệ sự tốt hoặc xấu của một mô hình trong việc phân loại. Nó được tính bằng cách lấy trung bình cộng của Precision và Recall. F1-score là một chỉ số ưu tiên cho cả Precision và Recall, vì vậy nó có thể được sử dụng để tìm mô hình tốt nhất với một cân bằng giữa số dự đoán đúng và số dự đoán sai.

2.12

Khi định nghĩa của "positive" thay đổi, giá trị của F1-score cũng sẽ tương ứng thay đổi. Điều này do thay đổi của giá trị TP trong công thức Precision và Recall. Vì vậy, khi chọn định nghĩa cho "positive", nên cẩn thận để tránh sai lầm, vì F1-score sẽ phụ thuộc vào định nghĩa này.

F1-score là một chỉ số hữu dụng khi ta muốn có một sự cân bằng giữa Precision và Recall. Đặc biệt hữu ích khi dữ liệu khảo sát của chúng ta không đồng đầu giữa nhãn "có" và "không" (ví dụ như số người khảo sát có bệnh chiếm tỷ trọng lớn hơn so với số người khảo sát không có bệnh). Nếu cải thiện một trong số Precision hoặc Recall, nhưng giá trị còn lại có thể bị ảnh hưởng lớn thì F1-score có thể rất nhỏ.

F1-score là một chỉ số cân bằng giữa Precision và Recall, cho phép ta đánh giá mức độ phân lớp của một mô hình. Giá trị của F1-score nằm trong khoảng (0,1], với giá trị càng cao tương ứng với mức độ phân lớp và mô hình tốt hơn.

* Trong trường hợp tốt nhất, khi Precision và Recall đều là 100%, tất cả các dự đoán của mô hình là chính xác.
* Trong trường hợp một trong hai giá trị là rất thấp, F1-score cũng sẽ thấp và cho thấy mô hình có vấn đề.

**Accuracy và Specificity**

Để đánh giá xem mô hình của chúng ta có hiệu quả hay không, có rất nhiều chỉ số khác ngoài F1-score ta có thể sử dụng. Ví dụ như accuracy và specificity, có thể được tính dựa trên các thành phần của confusion matrix.

**Accuracy**: là tỷ lệ dự đoán đúng trong tất cả các dự đoán của chúng ta. Nó đo lường tỷ lệ số dự đoán đúng so với tổng số dự đoán. Nếu dữ liệu khảo sát của chúng ta có số lượng người có và không có bệnh gần ngang nhau, chỉ số accuracy là một chỉ số hữu ích để đánh giá mô hình.

|  |  |
| --- | --- |
|  | 2.13 |

***Specificity:*** là tỷ lệ của số lượng trường hợp "không" đúng mà mô hình đã dự đoán đúng. Chúng ta sử dụng chỉ số Specificity khi chúng ta quan tâm đến việc xác định số lượng trường hợp "không" chính xác và muốn giảm số lượng False Positive. Ví dụ, trong trường hợp kiểm tra xem ai có bán ma túy hay không, chúng ta muốn Specificity cao vì chúng ta không muốn bắt giam những người không bán ma túy.

|  |  |
| --- | --- |
|  | 2.14 |

## Tiền xử lý dữ liệu

Tiền xử lý dữ liệu là quá trình chuẩn hoá, làm sạch và chuyển đổi dữ liệu ban đầu để chuẩn bị cho việc huấn luyện mô hình Học Máy. Điều này gồm các bước như loại bỏ dữ liệu trùng lặp, chuẩn hoá kích thước của dữ liệu và xử lý dữ liệu thiếu. Đây là một bước quan trọng trong quá trình giải quyết vấn đề Học Máy, tuy nhiên khi áp dụng vào dữ liệu thực tế, thì có thể gặp những vấn đề khác nhau vì các bộ dữ liệu tương ứng với các bài toán khác nhau và mỗi bài toán có những thách thức riêng về mặt dữ liệu.

Tiền xử lý dữ liệu là một quá trình quan trọng trong học máy, dữ liệu được sử dụng trong các vấn đề liên quan đến học máy cần phải được xử lý, làm sạch và biến đổi trước khi mô hình học máy được huấn luyện. Dữ liệu ban đầu có thể là dữ liệu có cấu trúc, bán cấu trúc hoặc không có cấu trúc và chất lượng dữ liệu được đánh giá theo những yếu tố như chính xác, hiện hành, toàn vẹn và nhất quán. Trong quá trình tiền xử lý dữ liệu, chất lượng dữ liệu (data quality) được xem là rất quan trọng. Có 4 thuộc tính chính của chất lượng dữ liệu: tính chính xác, tính hiện hành, tính toàn vẹn và tính nhất quán.

* Tính chính xác nghĩa là giá trị được ghi nhận là chính xác so với giá trị thực.
* Tính hiện hành nghĩa là giá trị được ghi nhận không bị lỗi thời.
* Tính toàn vẹn nghĩa là tất cả các giá trị dành cho một biến/thuộc tính đều được ghi nhận.
* Tính nhất quán nghĩa là tất cả giá trị dữ liệu đều được biểu diễn như nhau trong tất cả các trường hợp.

A picture containing timeline

Description automatically generated

Hình . Tiền xử lý dữ liệu cho học máy

Trong quá trình tiền xử lý dữ liệu, có nhiều kỹ thuật được sử dụng để cải thiện chất lượng dữ liệu và kết quả khai phá. Các kỹ thuật này bao gồm:

* Làm sạch dữ liệu: Xử lý các lỗi và nhiễu trong dữ liệu, để cải thiện tính chính xác và tính nhất quán của dữ liệu.
* Tích hợp dữ liệu: Kết hợp dữ liệu từ nhiều nguồn khác nhau thành một kho dữ liệu duy nhất để dễ dàng quản lý và sử dụng.
* Biến đổi dữ liệu: Chuyển đổi dữ liệu thông qua chuẩn hoá hoặc tiền xử lý để tạo ra dữ liệu mới và chuẩn hóa dữ liệu để dễ dàng huấn luyện và dự đoán.
* Thu giảm dữ liệu: Nó bao gồm các thao tác thu giảm kích thước dữ liệu bằng cách kết hợp dữ liệu (data aggregation), loại bỏ các đặc điểm dư thừa (redundant features) hoặc gom cụm dữ liệu. Các thao tác này có thể giúp giảm thiểu số lượng dữ liệu được sử dụng trong quá trình huấn luyện mô hình, giảm thời gian xử lý và tăng tốc độ huấn luyện mô hình.

Như vậy có thể thấy có rất nhiều các kỹ thuật tiền xử lý dữ liệu, trong phần này, em xin trình bày về hai kỹ thuật em áp dụng trong bài toán của mình là kỹ thuật xử lý dữ liệu khuyết và chuẩn hóa dữ liệu

### Xử lý dữ liệu khuyết

#### Đặt vấn đề

Trong thực tế, việc thu thập dữ liệu có thể bị giới hạn bởi những vấn đề như chi phí, cơ sở hạ tầng hoặc sự tham gia của con người. Điều này có thể dẫn đến việc thiếu dữ liệu do lỗi của máy móc hoặc thậm chí là dữ liệu không tồn tại tại một thời điểm nhất định. Nghiên cứu đã xác định rằng dữ liệu thiếu thường xảy ra vì các nguyên nhân như:

***Khuyết ngẫu nhiên*** (Missing at Random – MAR): Dữ liệu có thể bị thiếu ngẫu nhiên, trong đó sự thiếu dữ liệu của một đặc trưng không liên quan đến tính chất của đặc trưng đó nhưng có thể liên quan đến một số đặc trưng khác. Nói cách khác, sự thiếu dữ liệu của đặc trưng này có thể được giải thích bởi mối quan hệ giữa nó và một số đặc trưng khác.

***Khuyết hoàn toàn ngẫu nhiên*** (Missing Completely at Random – MCAR): là khi xu hướng bị khuyết của một đặc trưng là hoàn toàn ngẫu nhiên, không có bất kỳ mối quan hệ nào giữa giá trị đó với các giá trị giả định hoặc các ràng buộc trên các đặc trưng khác. Điều này có nghĩa là, tập dữ liệu bị khuyết chỉ là một tập con ngẫu nhiên của tập dữ liệu gốc.

***Khuyết không ngẫu nhiên*** (Missing not at Random – MNAR): là tình trạng khi một điểm dữ liệu bị thiếu, được xác định dựa trên cả các giá trị giả định (ví dụ như một số người không muốn chia sẻ thông tin về mức thu nhập của họ) và các giá trị của các đặc trưng khác. Nói cách khác, trong trường hợp này, sự thiếu dữ liệu không là một quy trình ngẫu nhiên mà có mối quan hệ với các giá trị giả định hoặc các ràng buộc trên các đặc trưng khác.

#### Các phương pháp xử lý khuyết thiếu dữ liệu

Trong thực tế, dữ liệu có thể không được thu thập hoặc ghi lại một cách chính xác do các nguyên nhân như chi phí, cơ sở hạ tầng hay lỗi của người. Điều này có thể gây ra các sự thiếu dữ liệu. Vì vậy, trong quá trình phân tích dữ liệu, chúng ta phải chú ý đến cách xử lý sự thiếu dữ liệu vì nếu không xử lý đúng cách, có thể dẫn đến kết luận sai lầm.

1. **Xóa đi dữ liệu bị khuyết**

Khi gặp trường hợp dữ liệu bị thiếu trong bộ dữ liệu được thu thập, một cách cơ bản để giải quyết là xóa đi các dữ liệu bị thiếu. Điều này giúp cho bộ dữ liệu trở nên rõ ràng hơn bằng cách loại bỏ những dữ liệu mập mờ và nhiễu.

Trong trường hợp dữ liệu bị thiếu ngẫu nhiên hoặc hoàn toàn ngẫu nhiên, xóa bỏ các dữ liệu bị thiếu dựa vào số lần xuất hiện của chúng là được chấp nhận. Tuy nhiên, trong trường hợp dữ liệu bị thiếu không ngẫu nhiên, việc xóa đi các dữ liệu có thể gây ảnh hưởng đến mô hình, vì vậy, chúng ta cần rất cẩn trọng trước khi xóa bỏ những dữ liệu này.

1. **Xóa bỏ các đặc trưng**

Thay vì xóa bỏ những đặc trưng không quan trọng từ tập dữ liệu thu thập được, chúng ta nên giữ lại những đặc trưng này. Tuy nhiên, trong một số trường hợp, việc xóa bỏ một số đặc trưng không quan trọng có thể là cần thiết để giảm thiểu tốn kém và tăng tốc độ tiền xử lý dữ liệu, đặc biệt khi chúng không quá quan trọng và bị khuyết hơn 60% tổng số quan sát. Tuy nhiên, tốt hơn hết là tìm cách xử lý dữ liệu bị khuyết thay vì xóa bỏ các đặc trưng đó.

1. **Điền vào dữ liệu bị khuyết**

Một cách khác để giải quyết vấn đề dữ liệu bị thiếu là sử dụng kĩ thuật "điền" (impute) giá trị thay vì xóa bỏ dữ liệu. Điều này có thể được thực hiện bằng cách thay thế các giá trị bị thiếu bằng giá trị trung bình hoặc trung vị của các giá trị còn lại. Đây là các phương pháp an toàn và đơn giản, vì giá trị trung bình và trung vị có xác suất xảy ra cao. Tuy nhiên, lựa chọn này cần được suy nghĩ cẩn thận và dựa trên dữ liệu đã xử lý trước đó.

### Chuẩn hóa dữ liệu

Chuẩn hóa dữ liệu là một bước quan trọng trong việc giải quyết các vấn đề trong học máy. Nó giúp đảm bảo rằng các thông tin được lấy từ dữ liệu là đồng nhất và phù hợp với các phương pháp phân tích dữ liệu khác nhau, giúp chúng ta tập trung vào các khía cạnh khác nhau của dữ liệu.

Chúng ta có một số phương pháp chuẩn hóa dữ liệu như sau:

#### Chuẩn hóa min-max (rescaling)

Chuẩn hóa min-max là một kĩ thuật chuẩn hóa dữ liệu, nó thực hiện việc co giãn các giá trị của đặc trưng từ giá trị thật về một phạm vi nhất định, trong trường hợp này là [0,1] hoặc [-1,1] . Công thức chung cho phương pháp này là:

|  |
| --- |
| 2.15 |

với x là giá trị ban đầu, x’ là giá trị sau khi chuẩn hóa, min(x) là giá trị nhỏ nhất của đặc trưng và max(x) là giá trị lớn nhất của đặc trưng.

#### Co giãn trung bình (mean normalization)

Tương tự như phương pháp rescaling, phương pháp co giãn xoay quanh trung bình có giá trị nằm trong khoảng [-0.5, 0.5] và được cho bởi công thức:

|  |
| --- |
| 2.16 |

với x là giá trị ban đầu, x’ là giá trị sau khi chuẩn hóa, min(x) là giá trị nhỏ nhất của đặc trưng và max(x) là giá trị lớn nhất của đặc trưng, average(x) là giá trị trung bình của đặc trưng.

#### Chính quy hóa (standardisation)

Trong lĩnh vực học máy, chúng ta có thể sẽ phải xử lý một lượng lớn các kiểu dữ liệu khác nhau, ví dụ như dữ liệu dạng tín hiệu âm thanh, các điểm ảnh trong một bức ảnh,… và những dữ liệu này có thể là các dữ liệu nhiều chiều. Việc chính quy hóa dữ liệu giúp cho giá trị của mỗi đặc trưng có trung bình bằng 0 và phương sai bằng 1. Phương pháp này được sử dụng rộng rãi trong việc chuẩn hóa dữ liệu của nhiều thuật toán học máy (SVM, logistic regression và ANNs).

|  |
| --- |
| 2.17 |

Trong đó x là véc-tơ đặc trưng ban đầu, average(x) là trung bình của véc-tơ đặc trưng đó và std(x) là độ lệch chuẩn của nó.

#### Mã hóa One-hot

Khi có các đặc trưng không có thứ tự, sử dụng mã hóa số có thể thay đổi bản chất dữ liệu. Điều này làm cho mô hình suy ra rằng các giá trị của đặc trưng dạng nhóm có thứ tự, dẫn đến kết quả dự đoán không chính xác. Vì vậy, sử dụng mã hóa one-hot là một giải pháp hiệu quả hơn, nó xóa bỏ đặc trưng dạng nhóm và chuyển đổi từng giá trị thành một biến nhị phân.

Phương pháp mã hóa one-hot sử dụng để chuyển đổi đặc trưng dạng nhóm với nhiều giá trị rời rạc thành nhiều đặc trưng nhị phân. Trong ví dụ về đặc trưng màu sắc, có 3 giá trị rời rạc, nên sẽ tạo ra 3 đặc trưng nhị phân và đặc trưng màu sắc gốc sẽ bị xóa bỏ. Tổng quát hóa, cho một đặc trưng nhóm có n giá trị rời rạc, thì sẽ cần n đặc trưng mới để lưu trữ giá trị.

# ỨNG DỤNG PHƯƠNG PHÁP VÀ XÂY DỰNG MÔ HÌNH

## Công cụ cài đặt mô hình

Từ cơ sở lý thuyết như trên, em sẽ tiến hành xây dựng mô hình Học máy bằng ngôn ngữ Python và các thư viện hỗ trợ của Python.

### Các thư viện hỗ trợ

Các thư viện của Python hỗ trợ cho việc phân tích dữ liệu và xây dựng mô hình bao gồm các thư viện sau:

#### Thư viện Numpy

Thư viện Numpy là một công cụ hữu ích và không thể thiếu đối với những người sử dụng Python trong ngành toán học và khoa học dữ liệu, là thư viện toán học mã nguồn mở trong Python dùng để xử lý các mảng và ma trận số học. Nó cung cấp nhiều công cụ hỗ trợ cho việc tính toán đa chiều, tính toán thống kê và xử lý ảnh.

#### Thư viện Pandas

Thư viện Pandas là một thư viện mã nguồn mở trong Python dùng để xử lý và phân tích dữ liệu. Nó cung cấp các công cụ hỗ trợ cho việc đọc và ghi dữ liệu từ các nguồn khác nhau, như tập tin CSV, Excel và SQL.

#### Thư viện Matplotlib

Thư viện Matplotlib là một thư viện mã nguồn mở trong Python dùng để vẽ các biểu đồ và hình học. Nó cung cấp rất nhiều kiểu biểu đồ khác nhau, bao gồm các biểu đồ đơn giản như đồ thị cột và đồ thị tròn, và các biểu đồ phức tạp hơn như đồ thị hàm số và đồ thị trục phức tạp.

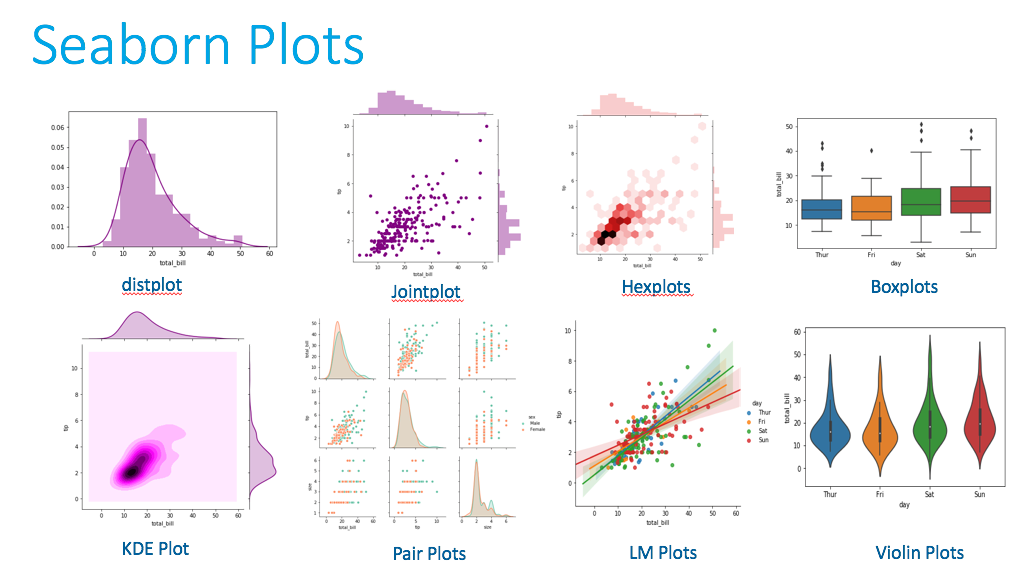
#### Thư viện Seaborn

Thư viện Seaborn là một thư viện trực quan hóa dữ liệu trong Python cung cấp rất nhiều công cụ hỗ trợ để vẽ các biểu đồ và hình học. Nó được xây dựng trên cơ sở của thư viện Matplotlib và cung cấp rất nhiều tính năng bổ sung để giúp người dùng vẽ các biểu đồ phức tạp hơn và trực quan hóa dữ liệu một cách dễ dàng hơn.

Thư viện Seaborn cung cấp rất nhiều kiểu biểu đồ khác nhau, bao gồm các biểu đồ thông thường như đồ thị cột, đồ thị tròn và đồ thị hàm số, và cũng có thể vẽ các biểu đồ phức tạp hơn như đồ thị phân bố liên tục và đồ thị phân tích hồi quy. Nó cũng có thể vẽ các hình học khác nhau như hình học phân tích hồi quy và hình học trục phức tạp.

Thư viện Python Seaborn được sử dụng để giảm bớt nhiệm vụ trực quan hóa dữ liệu và nó dựa trên Matplotlib. Seaborn cho phép tạo đồ họa thống kê thông qua các chức năng sau:

* Một API dựa trên tập dữ liệu cho phép so sánh giữa nhiều biến
* Hỗ trợ lưới nhiều ô để dễ dàng xây dựng các hình ảnh trực quan phức tạp
* Hình ảnh hóa đơn biến và lưỡng biến có sẵn để so sánh giữa các tập hợp con dữ liệu
* Có sẵn các bảng màu khác nhau để hiển thị các loại mẫu khác nhau
* Dự toán và âm mưu tuyến tính hồi quy tự động



Hình .: Một vài biểu đồ được tạo bởi seaborn

#### Thư viện Sklearn

Thư viện Scikit-learn là một trong những thư viện hỗ trợ machine learning phổ biến nhất trong ngôn ngữ lập trình Python. Nó cung cấp một loạt các thuật toán và hàm hỗ trợ cho việc huấn luyện và đánh giá mô hình, giúp cho việc phát triển và áp dụng các giải pháp machine learning trở nên dễ dàng hơn.

Scikit-learn (Sklearn) là thư viện mạnh mẽ nhất dành cho các thuật toán học máy được viết trên ngôn ngữ Python. Thư viện cung cấp một tập các công cụ xử lý các bài toán machine learning. Thư viện này tích hợp rất nhiều thuật toán hiện đại và cổ điển giúp ta vừa học vừa tiến hành đưa ra các giải pháp hữu ích cho các bài toán một cách đơn giản.

## Triển khai mô hình lên web bằng Flask Framework

Python ngày càng chứng minh ưu thế của mình trong việc xây dựng và triển khai nhiều loại ứng dụng khác nhau như web application, desktop application, phân tích thống kê, Machine Learning, Deep Learning…. Đó là nhanh chóng, mạnh mẽ, thân thiện, có thể thực thi ở mọi nơi và hoàn toàn miễn phí.

Flask là một web framework nhỏ gọn và linh hoạt trong ngôn ngữ lập trình Python, được sử dụng để phát triển các ứng dụng web và API (Application Programming Interface). Nó có cú pháp ngắn gọn và dễ hiểu, có thể dễ dàng kết hợp với các thư viện khác và có một cộng đồng lập trình rộng rãi cung cấp nhiều tài liệu hướng dẫn và hỗ trợ.

Flask cung cấp các tính năng cơ bản như routing (điều hướng), template rendering (kết xuất mẫu), và form validation (xác nhận biểu mẫu) để giúp ta phát triển ứng dụng web một cách dễ dàng. Nó cũng có thể dễ dàng kết hợp với các database (cơ sở dữ liệu) và các thư viện machine learning để phát triển các ứng dụng web có chức năng phong phú.

Timeline

Description automatically generated

Hình .: Các điểm mạnh của Python Flask

Flask là một lựa chọn ưu việt cho việc phát triển các ứng dụng web nhỏ và trung bình, hoặc làm một phần của một hệ thống lớn hơn. Nó cũng là một lựa chọn tốt cho việc học lập trình web và khám phá các khái niệm cơ bản của web development.

Một điểm ưu việt của Flask là nó có thể dễ dàng mở rộng bằng cách sử dụng các extension (phần mở rộng), mà không cần phải viết thêm nhiều mã nguồn. Có rất nhiều extension được cung cấp bởi cộng đồng lập trình viên và có thể dễ dàng cài đặt, giúp mở rộng chức năng của ứng dụng web của mình một cách dễ dàng.

Flask cũng hỗ trợ việc sử dụng các database như MySQL, PostgreSQL và SQLite để lưu trữ dữ liệu, và có thể dễ dàng kết hợp với các thư viện khác như SQLAlchemy để tiện lợi hơn trong việc làm việc với cơ sở dữ liệu.

### WSGI

WSGI được biết đến như là Web Server Gateway Interface (Giao diện cổng vào máy chủ Web) được sử dụng như một tiêu chuẩn để phát triển ứng dụng web Python. WSGI mang đặc điểm kỹ thuật giao diện chung giữa máy chủ web và ứng dụng web.

WSGI là một tiêu chuẩn mở, có thể được sử dụng với nhiều loại máy chủ web khác nhau và có thể được sử dụng với nhiều loại ứng dụng web khác nhau. Nó cung cấp một kết nối đồng nhất giữa các ứng dụng và các máy chủ.

WSGI là một phần không thể thiếu của các framework web trong Python như Flask hoặc Django, và cũng được sử dụng rộng rãi trong các ứng dụng web khác sử dụng Python. Nó có thể giúp bạn xây dựng các ứng dụng web động và hiệu quả hơn và dễ dàng triển khai chúng trên các máy chủ web.

WSGI có một tập hợp các hàm và biến mô hình (model variables) có sẵn để giúp các ứng dụng web và các máy chủ web giao tiếp với nhau. Nó cung cấp các hàm để xử lý các yêu cầu từ người dùng và trả lại các phản hồi tương ứng, ví dụ như trả lại một trang web hoặc một thông báo lỗi. Nó cũng có các biến mô hình để lưu trữ các thông tin về yêu cầu và phản hồi, giúp cho việc xử lý và hiển thị dữ liệu trên trang web dễ dàng hơn.

### Werkzeug

Werkzeug là một bộ công cụ WSGI thực hiện các yêu cầu, phản hồi đối tượng, và các chức năng tiện ích. Điều này cho phép xây dựng một web framework trên đó. Flask sử dụng Werkzeug làm một trong những cơ sở xây dựng web của nó.

Werkzeug là một thư viện Python dùng để xây dựng các ứng dụng web và cung cấp các công cụ hỗ trợ cho việc triển khai các ứng dụng. Nó được thiết kế với mục đích là độc lập với các framework web khác, nhưng cũng có thể sử dụng với các framework như Flask hoặc Django.

Werkzeug là một thư viện Python dùng để xây dựng các ứng dụng web và cung cấp các công cụ hỗ trợ cho việc triển khai các ứng dụng. Nó cung cấp các tính năng như xử lý yêu cầu và phản hồi HTTP, hỗ trợ cho việc chuyển hướng URL, xử lý các tệp đính kèm và các header HTTP, và hỗ trợ cho việc xử lý lỗi. Nó cũng cung cấp các công cụ hỗ trợ cho việc kiểm tra và debug các ứng dụng, và có thể hỗ trợ cho việc đăng nhập và xác thực người dùng, xử lý dữ liệu đầu vào và đầu ra, và lưu trữ dữ liệu trong các session và các cookie.

### Jinja2

Jinja2 là một template engine (máy kết xuất mẫu) trong ngôn ngữ lập trình Python, được sử dụng để tạo ra các mẫu HTML, XML hoặc các định dạng văn bản khác từ các biến và logic trong code. Nó cung cấp một cú pháp dễ hiểu và linh hoạt, giúp bạn dễ dàng tạo ra các mẫu động và tương tác với người dùng.

Jinja2 hỗ trợ việc tạo ra các macro (hàm tổng hợp) và các filter (bộ lọc) để xử lý dữ liệu trước khi hiển thị trên trang web. Macro giúp bạn tạo ra các đoạn mã HTML hay sử dụng thường xuyên, trong khi filter giúp bạn điều chỉnh hoặc chuyển đổi dữ liệu trước khi hiển thị.

Jinja2 cũng hỗ trợ việc sử dụng các biến mô hình (model variables) để tạo ra các mẫu động và tương tác với người dùng. Bạn có thể sử dụng các biến mô hình để hiển thị thông tin từ database hoặc các dữ liệu khác trên trang web.

Jinja2 có thể sử dụng để tạo ra các mẫu cho các framework web như Flask hoặc Django, hoặc có thể sử dụng độc lập với các dự án khác. Nó cung cấp các tính năng như các biến, các vòng lặp, các điều kiện và các hàm để tạo ra các mẫu động và xử lý dữ liệu trước khi hiển thị trên màn hình. Jinja2 cũng hỗ trợ việc tạo ra các macro (hàm tổng hợp) và các filter (bộ lọc) để xử lý dữ liệu trước khi hiển thị.

Flask luôn hỗ trợ các thành phần tiện ích mở rộng cho ứng dụng như tích hợp cơ sở dữ liệu, xác thực biểu mẫu, xử lý upload, các công nghệ xác thực, template, email, RESTful... Người dùng có thể tập trung xây dựng web application ngay từ đầu trong một khoảng thời gian rất ngắn và có thể phát triển quy mô của ứng dụng tùy theo yêu cầu.

Flask thật sự phù hợp cho việc xây dựng các web application có quy mô vừa và nhỏ, các API và web services:

* Xây dựng web application rất giống với việc viết các module Python chuẩn, cấu trúc gọn gàng và rõ ràng.
* Thay vì cung cấp hết tất cả mọi thứ, Flask cung cấp cho người dùng các thành phần cốt lõi thường được sử dụng nhất của khung ứng dụng web như URL routing, request & response object, template...
* Với Flask, việc chọn component nào cho ứng dụng là việc của chúng ta. Điều này thật tuyệt, vì mỗi web application có những đặc điểm và tính năng riêng, nó không phần phải chứa các component mà nó không dùng.

Flask có kiến trúc nhỏ, gọn nên hoàn toàn không bị bó buộc bởi bộ khung cồng kềnh, không gặp bất cứ khó khăn nào khi cấu hình hay tổ chức ứng dụng. Không những thế, Flask còn có các ưu điểm nổi bật như: cực kỳ linh hoạt, tối giản, dễ tìm hiểu và sử dụng, định tuyến dễ dàng, rất dễ mở rộng.

## Bài toán phân loại dự đoán khách hàng hủy đặt phòng

Bài toán dự đoán khách hàng hủy đặt phòng là khá quan trọng trong ngành khách sạn và du lịch. Để giải quyết bài toán này, chúng ta cần sử dụng một bộ dữ liệu chứa thông tin về khách hàng và các yếu tố có thể ảnh hưởng đến việc hủy đặt phòng. Bộ dữ liệu có thể bao gồm các thuộc tính như loại phòng, giá phòng, thời gian đặt phòng, hình thức thanh toán và các yếu tố khác. Biến mục tiêu trong trường hợp này là dự đoán liệu khách hàng sẽ hủy đặt phòng hay không.

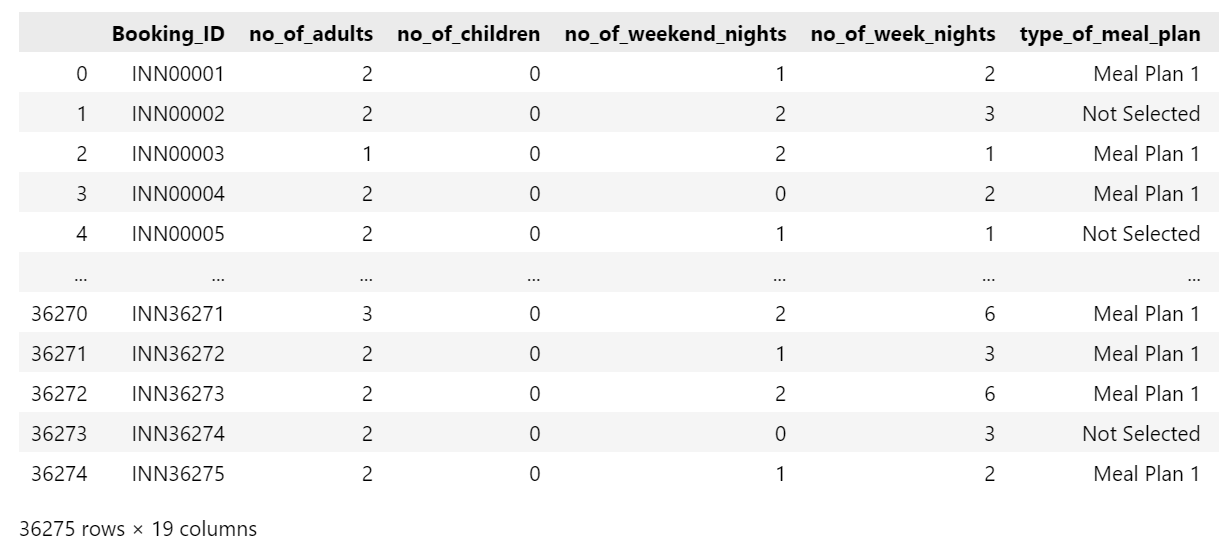
Bằng cách áp dụng các mô hình học máy, chúng ta có thể xây dựng một mô hình dự đoán dựa trên dữ liệu đầu vào để phân loại khách hàng thành hai nhóm: khách hàng có khả năng hủy đặt phòng và khách hàng không có khả năng hủy đặt phòng. Mô hình này có thể được huấn luyện trên một tập dữ liệu đã được gán nhãn, trong đó nhãn cho biết liệu khách hàng đã hủy đặt phòng hay không. Việc áp dụng các mô hình học máy vào bài toán dự đoán khách hàng hủy đặt phòng giúp nhà quản lý khách sạn hiểu rõ được hành vi của khách hàng và đưa ra các biện pháp phòng ngừa hủy đặt phòng.

### Mô tả bộ dữ liệu

Bộ dữ liệu đã được em lưu vào file “Hotel Reservations.csv”, để cài đặt bài toán, ta đọc bộ dữ liệu

|  |
| --- |
| df = pandas.read\_csv('Hotel Reservations.csv') |

Kết quả chạy như sau (Hình 3.4)



Hình .: Cấu trúc của dữ liệu

Bộ dữ liệu gồm: 36275 mẫu dữ liệu và 19 thông tin các trường như sau

Bảng .: Các trường thông tin trong tập dữ liệu

|  |  |  |
| --- | --- | --- |
| **Tên trường** | **Kiểu giá trị** | **Ý nghĩa** |
| Booking\_ID | object | Mã định danh đặt phòng |
| no\_of\_adults | int64 | Số lượng người lớn |
| no\_of\_children | int64 | Số lượng trẻ em |
| no\_of\_weekend\_nights | int64 | Số đêm cuối tuần (thứ bảy hoặc chủ nhật) của khách lưu trú hoặc đặt phòng lưu trú tại khách sạn |
| no\_of\_week\_nights | int64 | Số đêm trong tuần (thứ 2 đến thứ 6) khách lưu trú hoặc đặt phòng tại khách sạn |
| type\_of\_meal\_plan | object | Loại kế hoạch bữa ăn được đặt bởi khách hàng |
| required\_car\_parking\_space | int64 | Khách hàng có yêu cầu chỗ để ô tô hay không (0 - No, 1- Yes) |
| room\_type\_reserved | object | Loại phòng mà khách hàng đặt trước |
| lead\_time | int64 | Số ngày giữa ngày đặt phòng và ngày đến |
| arrival\_year | int64 | Năm đến |
| arrival\_month | int64 | Tháng đến |
| arrival\_date | int64 | Ngày đến |
| market\_segment\_type | object | Phương thức đặt phòng |
| repeated\_guest | int64 | Khách có phải là khách cũ không (0 - No, 1- Yes) |
| no\_of\_previous\_cancellations | int64 | Số lượt đặt trước đã bị khách hàng hủy trước lượt đặt hiện tại |
| no\_of\_previous bookings\_not\_canceled | int64 | Số lượt đặt trước không bị hủy bởi khách hàng trước lượt đặt hiện tại |
| avg\_price\_per\_room | float64 | Giá trung bình mỗi ngày của phòng |
| no\_of\_special\_requests | int64 | Tổng số yêu cầu đặc biệt của khách hàng (ví dụ: tầng cao, tầm nhìn từ phòng, v.v.) |
| booking\_status | object | Trạng thái cho biết khách có hủy đặt phòng hay không |

### Trực quan hóa dữ liệu

Sau đây là một vài phép trực quan hóa dữ liệu để chúng ta có thể dễ dàng quan sát được các đặc tính của dữ liệu

Sử dụng code sau để quan sát phân bố nhãn trạng thái đặt phòng của tập dữ liệu

|  |
| --- |
| df['booking\_status'].value\_counts().plot(kind='bar')  plt.xticks(rotation=0) |

Ta được kết quả như sau (Hình 3.5):

A blue and white rectangles

Description automatically generated with medium confidence

Hình .: Phân bố chất trạng thái hủy đặt phòng

Qua biểu đồ trên (hình 3.5), thấy được phân bố các nhãn ‘Not\_Canceled’ của booking\_status có tần suất xuất hiện nhiều hơn rất nhiều so với nhãn ‘Canceld’, cho thấy dữ liệu đang mất cân bằng.

Ta sử dụng code sau để quan sát sự tương quan giữa các thuộc tính:

|  |
| --- |
| plt.figure(figsize=(20,20))  sns.heatmap(df.corr(), annot=True, cmap='Blues')  plt.show() |

A screenshot of a computer screen

Description automatically generated with low confidence

Hình . Heatmap biểu hiện mức độ tương quan giữa các thuộc tính

Để quan sát phân mối mức giá của các căn phòng, ta chạy mã sau:

|  |
| --- |
| df['avg\_price\_per\_room'].hist(grid=False)  plt.xlabel('avg\_price\_per\_room')  plt.ylabel('frequency')  plt.title('Histogram of avg\_price\_per\_room')  plt.show() |

A picture containing rectangle, screenshot, graphics, symbol

Description automatically generated

Hình . Phân phối mức giá trung bình của phòng khách sạn

Ta thấy được mức giá phòng được đặt nhiều nhất là từ 50 – 100 trong tập dữ liệu khảo sát. Thực hiện tiếp câu lệnh sau để quan sát được tỉ lệ đặt phòng đối với giá phòng.

Để quan sát phân mối mức giá của các căn phòng, ta chạy mã sau:

|  |
| --- |
| plt.figure(figsize=(10,5))  sns.kdeplot(data = df, x = 'avg\_price\_per\_room', hue = 'booking\_status', shade=True)  plt.title('avg\_price\_per\_room') |

A picture containing screenshot, design

Description automatically generated

Hình . Phân phối giá phòng đối với booking\_status

Qua biểu đồ hình 3.8, ta có thể thấy được rằng trong khoảng giá phòng từ 50-100, tỉ lệ khách không hủy đơn cao, từ mức giá trung bình 100 trở đi, ta thấy được tỉ lệ đơn bị hủy cao hơn rất nhiều so với tổng đơn đặt phòng. Quan sát kỹ ta còn thấy được rằng mức giá trên 200 có khoảng mà đơn hủy đặt phòng nhiều hơn đơn không hủy. Điều đó cho ta thấy xu hướng rằng với mức giá phòng càng cao, khả năng đơn đặt phòng bị hủy cũng cao.

Để quan sát xem khách hàng đặt phòng theo phương thức nào thông qua trường thông tin *market\_segment\_type,* ta làm như sau:

|  |
| --- |
| # pie plot  df['market\_segment\_type'].value\_counts().plot(kind='pie', autopct='%1.1f%%', figsize=(10,5)) |

Ta được kết quả như hình dưới

A picture containing circle, screenshot, diagram, graphics

Description automatically generated

Hình . Tỉ lệ phân bố các phương thức đặt phòng

Qua biểu đồ trên, ta thấy khách hàng chủ yếu đặt phòng qua phương thức Online, chiếm đến 64% tổng số đơn đặt phòng, tiếp đó là hình thức offline, chiếm 29%. Ngoài ra các phương thức thanh toán như *Corporate, Complementary, Aviation* chiếm tỉ lệ nhỏ còn lại.

Để quan sát thêm sự tương quan giữa phương thức đặt phòng và trạng thái hủy đặt phòng, ta làm như sau:

|  |
| --- |
| plt.figure(figsize=(10,5))  sns.countplot(data = df, x = 'market\_segment\_type', hue = 'booking\_status')  plt.title('market\_segment\_type vs booking\_status') |

Ta được kết quả như hình dưới

A picture containing screenshot, text, rectangle, design

Description automatically generated

Hình . Mối quan hệ giữa phương thức đặt phòng và trạng thái hủy đặt phòng

Qua biểu đồ trên, ta thấy được rằng tỉ lệ hủy đặt phòng của 2 phương thức thanh toán Online và Offline đều bằng khoảng một nửa so với số đơn không bị hủy. Tuy nhiên ta thấy được ở 2 phương thức thanh toán *Corporate và Complementary* có tỉ lệ đơn đặt hàng bị hủy rất thấp.

Để khảo sát thêm trường thông tin về khách quen đối với tỉ lệ hủy đơn, ta làm như sau:

|  |
| --- |
| plt.figure(figsize=(10,5))  sns.countplot(data = df, x = 'market\_segment\_type', hue = 'booking\_status')  plt.title('market\_segment\_type vs booking\_status') |

Ta được kết quả như hình dưới (3.11)

A picture containing text, screenshot, rectangle, design

Description automatically generated

Hình . Mối quan hệ giữa khách quen và trạng thái hủy đặt phòng hay không

Qua biểu đồ trên, ta thấy được rằng khách quen (có chỉ số là 1) tuy chiếm số lượng ít nhưng lại không có tình trạng hủy đơn đặt phòng, trong khi tỉ lệ hủy đơn đặt phòng đối với khách không quen là gần 1/3 tổng số đơn đặt phòng.

## Các bước giải quyết bài toán

A diagram of a flowchart

Description automatically generated with low confidence

Hình . Các bước giải quyết bài toán

Để giải quyết bài toán này, em đề xuất các bước giải quyết bài toán như hình trên (Hình 3.12). Từ tập dữ liệu ban đầu, ta sẽ tiến hành tiền xử lý dữ liệu làm sạch dữ liệu, tiếp đến chuẩn hóa dữ liệu sao cho dữ liệu phù hợp với đầu vào của mô hình huấn luyện. Chia tập dữ liệu đã chuẩn hóa thành hai tập train và test. Bộ dữ liệu tập train để đưa vào xây dựng các mô hình học máy gồm Logistic Regression, Decision Tree, Random Forest Support Vector Machine. Sau khi mô hình được hoàn thành, ta đưa bộ dữ liệu test vào mô hình để đánh giá kiểm tra độ chính xác và hiệu quả của mô hình. Từ đó có thể chọn được mô hình thích hợp để triển khai lên web.

### Tiền xử lý dữ liệu

Với tiền xử lý dữ liệu, ta sẽ có các bước tìm kiếm xử lý missing data và chuẩn hóa dữ liệu như sau

#### Xử lý dữ liệu bị thiếu

Để tìm dữ liệu các bị thiếu, ta đếm số lượng các số liệu bị thiếu trong từng trường dữ liệu bằng đoạn code sau:

|  |
| --- |
| df.isnull().sum() |

Kết quả trả về như sau:

|  |
| --- |
| Booking\_ID 0  no\_of\_adults 0  no\_of\_children 0  no\_of\_weekend\_nights 0  no\_of\_week\_nights 0  type\_of\_meal\_plan 0  required\_car\_parking\_space 0  room\_type\_reserved 0  lead\_time 0  arrival\_year 0  arrival\_month 0  arrival\_date 0  market\_segment\_type 0  repeated\_guest 0  no\_of\_previous\_cancellations 0  no\_of\_previous\_bookings\_not\_canceled 0  avg\_price\_per\_room 0  no\_of\_special\_requests 0  booking\_status 0  dtype: int64 |

#### Chuẩn hóa dữ liệu

Để dữ liệu đầu vào phù hợp với mô hình học máy, ta cần phải chuẩn hóa dữ liệu. Các thuật toán học máy chỉ nhận các giá trị đầu vào dưới dạng các con số. Vì vậy nếu dữ liệu ở dạng khác ta cần phải chuẩn hóa về dạng số liệu. Đầu tiên, ta kiểm tra xem dữ liệu đã phù hợp hay chưa, ta dùng câu lệnh sau:

|  |
| --- |
| df.dtypes |

Ta được kết quả như hình bên dưới (Hình 3.13)

A screenshot of a computer program

Description automatically generated with medium confidence

Hình . Kiểm tra kiểu data của các trường thông tin

Ta có thể thấy một số các trường thông tin như *Booking\_ID*, *type\_of\_meal\_plan*, *room\_type\_reserved*, *market\_segment\_type*, *booking\_status.* Ngoại trừ *Booking\_ID* là trường thông tin không có ý nghĩa trong việc dự đoán khách hàng hủy đơn, ta cần phải chuẩn hóa các thông tin còn lại thành dạng số để có thể đưa vào mô hình học máy. Ta làm như sau:

|  |
| --- |
| from sklearn.preprocessing import LabelEncoder  le = LabelEncoder()  df['booking\_status'] = le.fit\_transform(df['booking\_status'])  df['type\_of\_meal\_plan'] = le.fit\_transform(df['type\_of\_meal\_plan'])  df['room\_type\_reserved'] = le.fit\_transform(df['room\_type\_reserved'])  df['market\_segment\_type'] = le.fit\_transform(df['market\_segment\_type']) |

Ta sử dụng thư viện LabelEncoder của sklearn để gán cho các giá trị kiểu object thành số hay nói cách khác, ta gán các số đại diện cho giá trị của object.

#### Xử lý mất cân bằng dữ liệu

Như đã trình bày ở biểu đồ hình 3.5, ta thấy được dữ liệu bị mất cân bằng, các mẫu dữ liệu có các đơn không bị hủy cao hơn rất nhiều so với đơn bị hủy đặt phòng, việc này sẽ ảnh hưởng rất nhiều đến quá trình huấn luyện mô hình. Vì vậy ta sẽ lấy ngẫu nhiên lượng mẫu đơn có nhãn không bị hủy ( Not\_Canceled) bằng với số lượng mẫu có nhãn bị hủy (Canceled) để giải quyết cho vấn đề mất cân bằng dữ liệu. Ta làm như sau:

|  |
| --- |
| df\_0 = df[df['booking\_status'] == 0]  df\_1 = df[df['booking\_status'] == 1]  n\_samples = min(df\_0.shape[0], df\_1.shape[0])  df\_0 = df\_0.sample(n=n\_samples, random\_state=42)  df\_1 = df\_1.sample(n=n\_samples, random\_state=42)  # concat 2 dataframe  df\_ = pandas.concat([df\_0, df\_1], axis=0) |

Thông tin của dữ liệu sau khi cân bằng:

A screenshot of a computer program

Description automatically generated with medium confidence

### Chia tập dữ liệu

Dữ liệu sẽ được chia 70% để huấn luyện, 30% còn lại để test mô hình. Bằng việc sử dụng train\_test\_split của thư viện sklearn, với test\_size là kích thước tập test chiếm 30% tập dữ liệu, random\_state là đối tượng điều khiển sự ngẫu nhiên trong quá trình tách.

|  |
| --- |
| from sklearn.model\_selection import train\_test\_split  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=.30, random\_state=42) |

### Xây dựng và huấn luyện mô hình học máy

Ta tiến hành xây dựng mô hình bằng các thuật toán học máy Logistic Regression, Decision Tree, Random Forest, Support Vector Machine.

#### Xây dựng mô hình Logistic Regression

Ta xây dựng logistic regression bằng thư viện sklearn của python. Thư viện sklearn hỗ trợ cung cấp đầy đủ công cụ cần thiết để xây dựng mô hình học máy logistic regression, giúp việc xây dựng mô hình cho bài toán trở nên đơn giản và đỡ phức tạp hơn.

Đầu tiên, ta khai báo thư viện cho logistic regression

|  |
| --- |
| from sklearn.linear\_model import LogisticRegression |

Tiếp đến, ta khởi tạo đối tượng cho logistic regression

|  |
| --- |
| logreg = LogisticRegression() |

Mô hình logistic regression sẽ được huấn luyện theo theo tập train của chúng ta, X\_train là tập các thuộc tính đặc trưng của tập mẫu, y\_train là tập phân loại theo tập mẫu tương ứng với X\_train.

|  |
| --- |
| logreg.fit(X\_train, y\_train) |

Cuối cùng, ta tính toán dự đoán cho mô hình ta vừa huấn luyện trên tập test, rồi lấy kết quả này so sánh với tập y\_test của tập dữ liệu để đánh giá độ chính xác của mô hình

|  |
| --- |
| pred = logreg.predict(X\_test) |

#### Xây dựng mô hình Decision Tree

Tương tự như logistic regression ta xây dựng mô hình Decision Tree bằng thư viện sklearn của python.

Đầu tiên, ta khai báo thư viện cho Decision Tree

|  |
| --- |
| from sklearn.tree import DecisionTreeClassifier |

Tiếp đến, ta khởi tạo đối tượng cho Decision Tree

|  |
| --- |
| dt = DecisionTreeClassifier() |

Mô hình Decision Tree sẽ được huấn luyện theo theo tập train của chúng ta, X\_train là tập các thuộc tính đặc trưng của tập mẫu, y\_train là tập phân loại theo tập mẫu tương ứng với X\_train.

|  |
| --- |
| dt.fit(X\_train, y\_train) |

Cuối cùng, ta tính toán dự đoán cho mô hình ta vừa huấn luyện trên tập test, rồi lấy kết quả này so sánh với tập y\_test của tập dữ liệu để đánh giá độ chính xác của mô hình

|  |
| --- |
| pred = dt.predict(X\_test) |

Để tối ưu cho thuật toán DT, ta có thể điều chỉnh tham số max\_depth của DT, ta làm như sau:

|  |
| --- |
| # cho max\_depth range(2, 51), mỗi giá trị tính accuracy rồi plot các giá trị  max\_depths = range(2, 51)  accuracies = []  for max\_depth in max\_depths:  dt = DecisionTreeClassifier(max\_depth=max\_depth)  dt.fit(X\_train, y\_train)  y\_pred = dt.predict(X\_test)  accuracies.append(accuracy\_score(y\_test, y\_pred))  plt.plot(max\_depths, accuracies)  plt.xlabel('max\_depths')  plt.ylabel('accuracy')  plt.title('max\_depths vs accuracy')  plt.show()  # in ra kết quả max\_depth tốt nhất  print('Max\_depths = ', max\_depths[accuracies.index(max(accuracies))]) |

Ta được kết quả như sau:

A picture containing text, screenshot, diagram, line

Description automatically generated

Hình . Tìm tham số max\_depth tối ưu cho DT

Ta thấy được tham số max\_depth tối ưu cho DT đối với tập dữ liệu này là 16, từ đó ta thiết lập mô hình DT với tham số tối ưu như sau:

|  |
| --- |
| # thiết lập max\_depth tốt nhất  max\_depth = max\_depths[accuracies.index(max(accuracies))]  dt = DecisionTreeClassifier(max\_depth=max\_depth)  dt.fit(X\_train, y\_train)  y\_pred = dt.predict(X\_test)  # accuracy  accuracy\_score(y\_test, y\_pred) |

#### Xây dựng mô hình Random Forest

Tương tự, ta xây dựng mô hình Random Forest bằng thư viện sklearn của python.

Đầu tiên, ta khai báo thư viện cho RandomForest.

|  |
| --- |
| from sklearn.ensemble import RandomForestClassifier |

Tiếp đến, ta khởi tạo đối tượng cho random forest, thiết lập n\_estimators( số lượng cây Decision Tree ) là tham số n\_estimators tối ưu vừa tìm được, trong quá trình thực nghiệm, đối với tập dữ liệu của chúng ta số cây n\_estimators vừa đảm bảo được độ chính xác của mô hình, vừa tối ưu được thời gian dự đoán của mô hình.

|  |
| --- |
| rf = RandomForestClassifier(n\_estimators=30) |

Mô hình RF sẽ được huấn luyện theo theo tập train của chúng ta, X\_train là tập các thuộc tính đặc trưng của tập mẫu, y\_train là tập phân loại theo tập mẫu tương ứng với X\_train.

|  |
| --- |
| rf.fit(X\_train, y\_train) |

Cuối cùng, ta tính toán dự đoán cho mô hình ta vừa huấn luyện trên tập test, rồi lấy kết quả này so sánh với tập y\_test của tập dữ liệu để đánh giá độ chính xác của mô hình

|  |
| --- |
| pred = rf.predict(X\_test) |

Để tối ưu cho thuật toán RF, ta có thể điều chỉnh tham số n\_estimators của RF, ta làm như sau:

|  |
| --- |
| # thiết lập tìm n\_estimators tốt nhất  n\_estimators = range(2, 300, 10)  accuracies = []  for n\_estimator in n\_estimators:  rf = RandomForestClassifier(n\_estimators=n\_estimator)  rf.fit(X\_train, y\_train)  y\_pred = rf.predict(X\_test)  accuracies.append(accuracy\_score(y\_test, y\_pred))  plt.plot(n\_estimators, accuracies)  plt.xlabel('n\_estimators')  plt.ylabel('accuracy')  plt.title('n\_estimators vs accuracy')  plt.show()  # in ra kết quả n\_estimators tốt nhất  print('n\_estimators = ', n\_estimators[accuracies.index(max(accuracies))]) |

Ta được kết quả như sau:

A picture containing text, screenshot, diagram, plot

Description automatically generated

Hình . Tìm tham số n\_estimators tối ưu cho RF

Ta thấy được tham số n\_estimators tối ưu cho RF đối với tập dữ liệu này là 192, accuracy của RF hầu như dao động xung quanh giá trị từ 0.87 đến 0.89 từ n\_estimators >= 50. Từ đó ta thiết lập mô hình DT với tham số n\_estimators = 50 hoặc theo tham số tối ưu tìm được như sau:

|  |
| --- |
| # thiết lập max\_depth tốt nhất  max\_depth = max\_depths[accuracies.index(max(accuracies))]  dt = DecisionTreeClassifier(max\_depth=max\_depth)  dt.fit(X\_train, y\_train)  y\_pred = dt.predict(X\_test)  # accuracy  accuracy\_score(y\_test, y\_pred) |

#### Xây dựng mô hình SVM

Ta xây dựng SVM bằng thư viện sklearn của python. Thư viện sklearn hỗ trợ cung cấp đầy đủ công cụ cần thiết để xây dựng mô hình học máy SVM, giúp việc xây dựng mô hình cho bài toán trở nên đơn giản và đỡ phức tạp hơn.

Đầu tiên, ta khai báo thư viện cho SVM

|  |
| --- |
| from sklearn.svm import SVC |

Tiếp đến, ta khởi tạo đối tượng cho logistic regression

|  |
| --- |
| svc = SVC() |

Mô hình logistic regression sẽ được huấn luyện theo theo tập train của chúng ta, X\_train là tập các thuộc tính đặc trưng của tập mẫu, y\_train là tập phân loại theo tập mẫu tương ứng với X\_train.

|  |
| --- |
| svc.fit(X\_train, y\_train) |

Cuối cùng, ta tính toán dự đoán cho mô hình ta vừa huấn luyện trên tập test, rồi lấy kết quả này so sánh với tập y\_test của tập dữ liệu để đánh giá độ chính xác của mô hình

|  |
| --- |
| y\_pred = svc.predict(X\_test) |

# Kết quả và đánh giá mô hình và triển khai mô hình lên web

## Kết quả và đánh giá mô hình

Để đánh giá được kết quả thực nghiệm của hệ thống ta thực hiện đánh giá độ chính xác qua 10 lần thực nghiệm, trên các mô hình học máy khác nhau, với mỗi lần thử tập train test được lấy ra từ tập dữ liệu khác nhau. Sau khi xây dựng xong mô hình và đánh giá mô hình trên tập dữ liệu test, ta thu được kết quả độ chính xác của các mô hình học máy như sau.

Bảng .: Độ chính xác của LR với tiêu chí accuracy

|  |  |
| --- | --- |
| **Lần chạy thứ** | **Accuracy** |
| 1 | 75.43% |
| 2 | 75.78% |
| 3 | 76.92% |
| 4 | 77.59% |
| 5 | 76.14% |
| 6 | 76.35% |
| 7 | 76.92% |
| 8 | 75.38% |
| 9 | 75.34% |
| 10 | 75.49% |
| Độ chính xác trung bình của LR | 76.13% |

Bảng .: Độ chính xác của DT với tiêu chí accuracy

|  |  |
| --- | --- |
| **Lần chạy thứ** | **Accuracy** |
| 1 | 84.89% |
| 2 | 85.14% |
| 3 | 84.85% |
| 4 | 84.41% |
| 5 | 85.46% |
| 6 | 84.62% |
| 7 | 84.43% |
| 8 | 85.25% |
| 9 | 84.70% |
| 10 | 83.76% |
| Độ chính xác trung bình của DT | 84.75% |

Bảng .: Độ chính xác của RF với tiêu chí accuracy

|  |  |
| --- | --- |
| **Lần chạy thứ** | **Accuracy** |
| 1 | 88.70% |
| 2 | 89.14% |
| 3 | 88.87% |
| 4 | 88.55% |
| 5 | 88.62% |
| 6 | 88.26% |
| 7 | 88.64% |
| 8 | 88.34% |
| 9 | 88.95% |
| 10 | 87.88% |
| Độ chính xác trung bình của RF | 88.60% |

Bảng . Độ chính xác của SVM với tiêu chí accuracy

|  |  |
| --- | --- |
| **Lần chạy thứ** | **Accuracy** |
| 1 | 69.24 % |
| 2 | 68.93 % |
| 3 | 69.96 % |
| 4 | 69.52 % |
| 5 | 69.33 % |
| 6 | 69.92 % |
| 7 | 68.84 % |
| 8 | 69.62 % |
| 9 | 69.56 % |
| 10 | 69.56 % |
| Độ chính xác trung bình của SVM | 69.45% |

Từ những kết quả thu được từ 10 lần thử nghiệm bởi các mô hình **DT, RF, LR, SVM** như trên, ta quan sát được kết quả trung bình phân lớp của bốn thuật toán như sau:

Bảng .: Kết quả trung bình phân lớp của bốn thuật toán

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **LR** | **Decision Tree** | **RF** | **SVM** |
| **Accuracy** | 76.13% | 87.75% | 88.60% | 69.45% |

Hình .: Accuracy của các mô hình học máy

Qua kết quả trên, ta thấy độ chính xác của Decision Tree và Random Forest đạt hiệu quả tốt nhất, trong đó Random Forest có độ chính xác cao nhất.

A picture containing text, screenshot, font, number

Description automatically generated

Hình . Report classification của LR

A picture containing text, screenshot, font, receipt

Description automatically generated

Hình . Report classification của DT

A picture containing text, screenshot, receipt, font

Description automatically generated

Hình . Report classification của RF

A picture containing text, screenshot, font, receipt

Description automatically generated

Hình . Report classification của SVM

Kết quả khi sử dụng Random Forest mang lại hiệu quả cao đối với bài toán này, random forest tìm ra được những quy tắc, từ đó đưa ra được những dự đoán chính xác dựa trên những tiêu chí đó.

Sau khi so sánh mô hình, em thấy RF là mô hình cho kết quả tốt nhất. Vì vậy, em sử dụng mô hình này để triển khai lên web.

## Triển khai mô hình lên web

Mô hình của Random Forest được chọn làm mô hình dự đoán cho web.Giao diện của web gồm một trang duy nhất, dùng để nhập các trường thông tin để dự đoán khách hàng hủy đặt phòng.

Các giá trị đầu vào được nhập vào theo thiết lập đối với 17 trường thông tin (thông tin BOOKING\_ID không có ý nghĩa dự đoán). Trang web dự đoán mô hình đã được triển khai thành công và được minh họa bởi các tấm hình bên dưới.

A screenshot of a computer

Description automatically generated with low confidence

Hình . Giao diện chính ban đầu của web

Trang web bao gồm các ô input đầu vào để nhập các giá trị thông tin tương ứng. Sau khi nhập các trường thông tin và ấn nút **submit**, trang web sẽ trả về kết quả dự đoán khách hàng hủy đặt phòng.

A screenshot of a phone

Description automatically generated with low confidence

Hình . Web trả về kết quả dự đoán khách hàng

Kết quả trả về đối với các giá trị đầu vào tương ứng với ví dụ trên, cho ra kết quả dự đoán Not\_Canceled (không hủy đặt phòng).

A screenshot of a computer

Description automatically generated with medium confidence

Hình . Kiểm tra input đầu vào nằm ngoài khoảng phân phối

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Đồ án đã nghiên cứu, tìm hiểu bài toán

Xây dựng và ứng dụng học máy trong dự đoán khách hàng hủy đặt phòng, thực hiện phát triển, cài đặt phương án giải quyết cho bài toán dựa trên sự thống kê. Các kết quả đạt được như mục tiêu đề ra ban đầu như sau:

***Kết quả***

*Về mặt lý thuyết:*

* Tìm hiểu các phương pháp học máy Logistic Regression, Decision Tree, Random Forest, Support Vector Machine
* Tìm hiểu được tổng quan các bước xây dựng mô hình Học máy.

*Về mặt ứng dụng:*

* Thực nghiệm trên môi trường Python và đánh giá kết quả.
* Triển khai được web phân loại dự đoán khách hàng hủy đặt phòng cơ bản từ mô hình đã xây dựng bằng Flask.

***Hạn chế***

* Thực nghiệm mới chỉ để minh họa, xác nhận đúng cho đề tài, chỉ dừng ở bước nghiên cứu là chính, chưa có tính ứng dụng cao trong thực tế.
* Bộ dữ liệu còn nhỏ, chưa có tính khái quát cao.

***Hướng phát triển***

* Nghiên cứu thêm về các phương pháp của các thuật toán học máy, mang lại hiệu quả cao cho bài toán dự đoán khách hàng hủy đặt phòng khách sạn.
* Thực nghiệm trên các bộ dữ liệu có kích thước lớn hơn, có tính khái quát hơn.
* Hỗ trợ đưa ra các dự đoán cho các bộ dữ liệu dự đoán khách hàng hủy đặt phòng cho ngành quản lý khách sạn.

TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | AHSAN RAZA, Hotel Reservations. [Online]. Available: https://www.kaggle.com/code/christophertimmons/random-forest-97-accuracy-score |
| [2] | ”Tiền xử lý dữ liệu trong lĩnh vực học máy”. Available: https://vimentor.com/en/lesson/tien-xu-ly-du-lieu-trong-linh-vuc-hoc-may-phan-1 |
| [3] | ”How to Handle Missing Data”. Available:  https://towardsdatascience.com/how-to-handle-missing-data-8646b18db0d4 |
| [4] | ”Decision Tree”. Available:  https://en.wikipedia.org/wiki/Decision\_tree |
| [5] | ”Random Forest”. Available: https://machinelearningcoban.com/tabml\_book/ch\_model/random\_forest.html |
| [6] | ”Logistic Regression”. Available: https://machinelearningcoban.com/2017/01/27/logisticregression/ |
| [7] | Pranav Kumar, Sarthak Sharma , " Hotel Booking Prediction using Machine Learning". [Online]. Available: <https://doi.org/10.22214/ijraset.2022.43036> |
| [8] | Enok Gartvall & Oscar Sk ̊anhagen, "Predicting hotel cancellations using machine learning" [Online]. Available: [https://gupea.ub.gu.se/bitstream/handle/2077/70742/gupea\_2077\_70742\_1.pdf?sequence=1](https://gupea.ub.gu.se/bitstream/handle/2077/70742/gupea_2077_70742_1.pdf?sequence=1&fbclid=IwAR1hHhOkaXJSj6ekfUeZwvDSgSEYbjH6wRUfjnGVuOLalDL2FfP6ezA0Etg) |