Hotel Booking Cancellation Prediction*

*Dự đoán khả năng hủy đặt phòng khách sạn

1st Nguyễn Thị Hoàng Anh
Khoa Khoa học và Kỹ thuật thông tin
Đại học Công nghệ thông tin
Đại học Quốc gia TPHCM
20520134@gm.uit.edu.vn

2nd Nguyễn Hà Dung
Khoa Khoa học và Kỹ thuật thông tin
Đại học Công nghệ thông tin
Đại học Quốc gia TPHCM
20520165@gm.uit.edu.vn

3rd Nguyễn Minh Trí
Khoa Khoa học và Kỹ thuật thông tin
Đại học Công nghệ thông tin
Đại học Quốc gia TPHCM
2052052@gm.uit.edu.vn

***Tóm tắt nội dung*—Ở báo cáo này, chúng em tiến hành phân tích trên bộ dữ liệu Hotel Booking Demand về thông tin các lượt đặt phòng đã được thu thập và áp dụng một vài mô hình máy học để tìm ra mô hình tối ưu cho bài toán dự đoán.**

***Keywords*—Hotel Booking Demand, modeling, hyperparameter**

I. Giới thiệu

Trong điều kiện du lịch phát triển ngày nay, việc đặt phòng khách sạn trở nên rất phổ biến và song song đó tỉ lệ hủy đặt phòng cũng có tác động đáng kể tới việc quản lý nhân sự trong ngành dịch vụ khách sạn. Nhằm hạn chế những rắc rối do việc hủy đặt phòng gây ra, ta nhận thấy được một khách sạn cần thiết có một mô hình cho phép dự đoán lượt đặt phòng có bị hủy không một cách chính xác nhất. Từ đó, nhóm chúng em quyết định phân tích độ chính xác của một số mô hình máy học trên bộ dữ liệu Hotel Booking Demand nhằm tìm ra mô hình tối ưu nhất cho bài toán dự đoán.

Bài toán dự báo (Prediction) là một trong những bài toán thông dụng trong Machine Learning, nằm trong nhóm học có giám sát (Supervised learning). Bài toán dự đoán đặt ra là xác định một lượt đặt phòng có bị hủy hay không. Bài toán thuộc bài toán phân lớp (Classification) với đầu vào (Input) là thông tin đặt phòng và đầu ra (Output) là 0 (không) hoặc 1 (có).

II. Dữ liệu

A. Giới thiệu bộ dữ liệu

Bộ dữ liệu ban đầu từ bài báo Hotel Booking Demand Datasets[1] được viết bởi Nuno Antonio, Ana Almeida và Luis Nunes cho Data in Brief. Sau đó được tải về , làm sạch bởi Thomas Mock và Antoine Bichat và đăng tải tại Kaggle[2].

Bộ dữ liệu gồm 32 biến mô tả 119.390 mẫu quan sát được ở hai loại hình khách sạn (City hotel và Resort hotel). Thời gian quan sát từ 1/7/2015 đến 31/8/2017. Mỗi mẫu đại diện cho một lượt đặt phòng ở khách sạn bao gồm cả các lượt đặt thành công và lượt đã bị hủy. Một mẫu quan sát gồm các thuộc tính liên quan đến dịch vụ như ngày nhận phòng, số ngày ở, số người, ... với nhãn (Label) là thuộc tính is_canceled (0/1)

Identify applicable funding agency here. If none, delete this.

Kiểu dữ liệu	Thuộc tính
Label	is_canceled
Numeric	lead_time, arrival_date_year, arrival_date_week_number, arrival_date_day_of_month, stays_in_week_nights, adults, children, babies, is_repeated_guest, previous_cancellations, previous_bookings_not_canceled, booking_changes, agent, required_car_parking_spaces, total_of_special_requestscompany, days_in_waiting_list, adr
Category	hotel, arrival_date_month, meal, country, market_segment, distribution_channel, reserved_room_type, assigned_room_type, deposite_type, agent, company, customer_type, reservation_status, reservation_status_date

Bảng I: Mô tả thuộc tính

B. Tiền xử lý dữ liệu

Trong tác vụ học máy, dữ liệu là điều không thể thiếu, cần phải có quá trình xử lý làm cho chúng trở nên hữu ích và hoạt động tốt bằng các tác vụ tiền xử lý dữ liệu (Data preprocessing - DP).

DP là một quá trình chuẩn bị dữ liệu thô và làm cho nó phù hợp với mô hình máy học. Đây là bước đầu tiên và quan trọng trong quá trình đào tạo mô hình học máy. Khi xây dựng mô hình máy học và tìm kiếm dữ liệu, không phải lúc nào ta cũng bắt gặp dữ liệu đã được xử lý theo định dạng nhất định và sạch sẽ, không thể sử dụng trực tiếp cho các mô hình máy học. Vì thế, tiền xử lý dữ liệu là bước quan trọng cần thiết phải thực hiện với dữ liệu, điều này làm tăng độ chính xác và hiệu quả của các mô hình học máy.

1) Xử lý dữ liệu bị thiếu: Trong bộ dữ liệu, có nhiều cột tồn tại giá trị Null, đối với thuộc tính có tỉ lệ giá trị Null thấp ta sẽ tiến hành điền thiếu và đối với các thuộc tính tỷ lệ cao ta sẽ tiến hành xóa. Nhưng đối với thuộc tính company và agent, dựa trên ý tưởng thu thập dữ liệu của tác giả, giá trị missing tồn tại có thể do khách không đăng ký thông qua agent/company nên ta sẽ tiến hành điền thiếu bằng giá trị '0'.

Thuộc tính	Số lượng Null	Phần trăm	Phương pháp
children	4	0.003%	Loại bỏ
country	488	0.04%	Điền giá trị 'unknown'
agent	16340	13.68%	Điền giá trị '0'
company	112593	94.30	Điền giá trị '0'

Bảng II: Dữ liệu bị thiếu và phương pháp xử lý

2) Xử lý kiểu dữ liệu: Khi tiếp cận với một bài toán Machine Learning, khả năng cao là chúng ta sẽ phải đối mặt với dữ liệu dạng phân loại (categorical data). Khác với các dữ liệu dạng số, máy tính sẽ không thể hiểu và làm việc trực tiếp với categorical variable. Do vậy, nhiệm vụ của chúng ta là phải tìm cách "encode" dữ liệu dạng categorical, đưa nó về dạng khác để có thể đưa vào mô hình của mình.

Với mỗi giá trị khác nhau của categorical feature, chúng ta sẽ tạo ra một feature mới. Mỗi feature mới này sẽ được gán cho giá trị 0 hoặc 1. Nếu sản phẩm thuộc category nào thì giá trị ở đó sẽ là 1. Những features mới tạo ra được gọi là Dummy variables. Dummy encoding cần k-1 dummy variables.

III. Phương pháp máy học

A. Logistic Regression

Logistic Regression là thuật toán đơn giản nhưng lại rất hiệu quả trong bài toán phân loại (Classification). Mô hình hồi quy Logistic là sự tiếp nối ý tưởng của hồi quy tuyến tính vào các bài toán phân loại.

Các biến dự báo trong Logistic Regression có thể là categorical hoặc numeric, và biến mục tiêu của Logistic Regression là nhị phân hoặc lưỡng phân. Do đó, hồi quy logistic không thể dự đoán các biến mục tiêu của nhiều hơn hai lớp. Mặc dù Logistic Regression có thể có một số điểm yếu, nó thường có thể cạnh tranh với các kỹ thuật máy học khác, chẳng hạn như mạng nơ-ron, SVM, Random Forest và Gradient Boosting. Việc chính thức hóa hồi quy logistic được phát biểu[3] như sau:

$$\pi = Probability\ (Y|X) = \frac{e^{\alpha+\beta_1 x_1+\beta_2 x_2+...+\beta_n x_n}}{1+e^{\alpha+\beta_1 x_1+\beta_2 x_2+...+\beta_n x_n}} \quad (1)$$

Chú thích:

Y: là kết quả quan tâm
X: x, một giá trị cụ thể của X
π: là xác suất của kết quả mong muốn
e: 2.71828 (Cơ số của hệ Logarit tự nhiên)
α: là chặn của Y
β_n: là hệ số hồi quy
x_n: là tập hợp các biến dự báo

B. k-Nearest Neighbors

k-Nearest Neighbors (kNN) là một trong những thuật toán non-parametric supervised-learning (học có giám sát không tham số) đơn giản nhất trong Machine Learning. Đây cũng là lý do thuật toán này được xếp vào loại lazy learning, nó phân loại quan sát mới bằng cách tìm điểm tương đồng giữa quan sát mới này với dữ liệu sẵn có. k-Nearest Neighbors có thể áp dụng được vào cả hai loại của bài toán Supervised learning là Phân lớp (Classification) và Hồi quy (Regression).

Trong bài toán phân lớp kNN, đầu ra là một thành viên của lớp. Một đối tượng được phân loại bằng một phiếu bầu giữa các điểm láng giềng gần nhất, với đối tượng được gán vào lớp phổ biến nhất trong số k điểm láng giềng gần nhất của nó (k là một số nguyên dương, thường nhỏ). KNN tính toán khoảng cách giữa các mẫu và xác định lớp cho mỗi giá trị. Nó có ba phần thiết yếu: đầu tiên, tập hợp các đối tượng được dán nhãn; thứ hai, khoảng cách giữa các đối tượng; thứ ba, số lượng hàng xóm gần nhất. Việc chính thức hóa phân loại KNN (khoảng cách Euclide) được nêu như sau [4]:

$$Dist(X,Y) = \sqrt{\sum_{i=1}^{D}(X_i - Y_i)^2} \quad (2)$$

Chú thích:

X: là lớp không đặt trước
Y: là lớp đặt trước
X_i (1, ..., N): là thuộc tính của đối tượng mẫu X
Y_i (1, ..., N): là thuộc tính của đối tượng mẫu Y
D: là khoảng cách những láng giềng gần nhất

C. Decision Tree/Classification and Regression tree (CART)

Mô hình cây quyết định là một mô hình được sử dụng khá phổ biến và hiệu quả trong cả bài toán phân lớp và hồi quy. Khác với những thuật toán khác trong học có giám sát, mô hình cây quyết định không tồn tại phương trình dự báo. Nó bắt chước mức độ suy nghĩ của con người nên nó đơn giản để hiểu và chuẩn bị dữ liệu, giúp bạn thấy được logic từ dữ liệu. Cây quyết định là cây mà mỗi nút biểu diễn một đặc trưng (Feature), mỗi nhánh (Branch) biểu diễn một quy luật (Rule) và mỗi lá biểu diễn một kết quả (giá trị cụ thể hay một nhánh tiếp tục).

CART có thể giải quyết một vấn đề phân loại. Giống như tên gọi của nó, thuật toán CART trông giống như một cấu trúc cây. Nó có một nút gốc, các nút lá và các nhánh; và một số ưu điểm, chẳng hạn như tính phi tham số, thích ứng với bất kỳ tập dữ liệu nào và có thể xử lý mối quan hệ phi tuyến tính [5]. CART là một thuật toán được sử dụng trong cây quyết định [6] và nó sử dụng chỉ số Gini để đánh giá sự phân tách. Điểm tốt nhất là 0 và điểm kém nhất là một giá trị bằng nhau cho mỗi lớp. Việc chính thức hóa chỉ mục Gini được nêu như sau[7]:

$$i(t) = \sum_{i=1}^{k}\sum_{j=1,j\neq i}^{k} p(c_i|t)p(c_j|t) = 1 - \sum_{j=i}^{k}(p(c_j|t))^2 \quad (3)$$

Chú thích:

i(t): là xác suất ước tính của việc phân loại sai theo Chỉ số Gini
c_j: là lớp của j
c_i :: là lớp của i
P: là xác suất
t: là nút
k: là lớp

D. Neural Network

Neural là tính từ của neuron (nơ-ron), network chỉ cấu trúc đồ thị nên Neural Network (NN) là một hệ thống tính toán lấy cảm hứng từ sự hoạt động của các nơ-ron trong hệ thần kinh mà một chuỗi thuật toán được đưa ra để tìm kiếm mối quan hệ cơ bản trong tập hợp dữ liệu. Thông qua việc bắt chước cách thức hoạt động từ não bộ con người, có khả năng thích

ứng mọi thay đổi từ đầu vào và có thể đưa ra mọi kết quả một cách tốt nhất mà bạn không cần thiết kế lại tiêu chí đầu ra.

Trong số một số kiến trúc của mô hình mạng nơ-ron, mô hình được sử dụng rộng rãi nhất trong dự báo là mô hình perceptron đa lớp lan truyền ngược (MLP). Như Hình 1 minh họa, một mạng nơ-ron MLP bao gồm ba hoặc nhiều lớp: một lớp đầu vào, một lớp đầu ra và ít nhất một lớp ẩn. Đầu ra của một nút (nơ-ron) được tính[8] như sau:

$$O_i = g\left(\sum w_{ij} I_j\right) \tag{4}$$

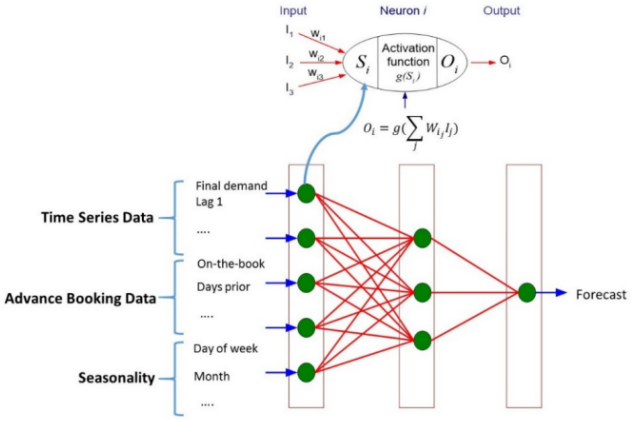
Chú thích:
O_i: là đầu ra từ nút i
g: là hàm kích hoạt
w_{ij}: là trọng số từ nút i tới nút j
I_j: là đầu vào từ nút j

Các hàm kích hoạt thường được sử dụng bao gồm hàm sigmoid (logistic), tuyến tính từng phần (Rectified Linear Unit) và hàm bước, như được trình bày bên dưới:

$$g(x) = \frac{1}{1+e^{-x}} \qquad sigmoid(logistic) \tag{5}$$

$$g(x) = max(0, x) \qquad piecewise\ linear \tag{6}$$

$$g(x) = \begin{cases} 1 & if\ x \geqslant 0 \\ 0 & if\ x < 0 \end{cases} \qquad step\ function \tag{7}$$



Hình 1: Mô hình Neural Network

IV. Các thí nghiệm tinh chỉnh mô hình

A. Độ đo đánh giá

Trong thực tế có ba độ đo chủ yếu để đánh giá một mô hình phân loại là Accuracy, Precision and Recall. Accuracy được định nghĩa là tỷ lệ phần trăm dự đoán đúng cho dữ liệu thử nghiệm. Nó có thể được tính toán dễ dàng bằng cách chia số lần dự đoán đúng cho tổng số lần dự đoán.

$$accuracy = \frac{correct\ predictions}{all\ predictions} \tag{8}$$

Precision được định nghĩa là phần nhỏ của các ví dụ có liên quan (tích cực thực sự) trong số tất cả các ví dụ được dự đoán là thuộc một lớp nhất định.

$$precision = \frac{true\ positives}{true\ positives + false\ positives} \tag{9}$$

Recall được định nghĩa là phần nhỏ của các ví dụ được dự đoán thuộc về một lớp so với tất cả các ví dụ thực sự thuộc về lớp đó.

$$recall = \frac{true\ positives}{true\ positives + false\ negatives} \tag{10}$$

Trong trường hợp các nhãn không được phân bố đồng đều thì việc sử dụng Accuracy làm thước đo đánh giá mô hình thường không hiệu quả bởi hầu hết chúng đều đạt độ chính xác rất cao. Khi đó ta có thể cân nhắc tới một số độ đo thay thế như Precision, Recall, F1-score, AUC,.... Các chỉ số này sẽ không quá lớn để dẫn tới ngộ nhận độ chính xác, đồng thời chúng tập trung hơn vào việc đánh giá độ chính xác trên nhóm thiểu số, nhóm mà chúng ta muốn dự báo chính xác hơn so với nhóm đa số. Do đó, đồ án sẽ sử dụng độ đo F1 và AUC để làm độ đo đánh giá. Độ đo F1 được kết hợp giữa Precision và Recall:

$$F1 - score\ = 2 * \frac{precision * recall}{pecision + recall} \tag{11}$$

AUC là phần diện tích bên dưới ROC curve, dùng để đánh giá hiệu năng phân loại của các mô hình với nhau. Mô hình có AUC càng lớn (ROC curve càng gần góc cao bên trái) thì cho kết quả càng chính xác.

B. Trích xuất đặc trưng

Một trong những phương pháp xác định độ quan trọng của các đặc trưng trong một tập dữ liệu là thuật toán Random Forest. Random Forest đơn giản là tập hợp của các cây quyết định Decision Tree. Đối với bất kỳ cây quyết định nào, các đặc trưng được chọn bằng cách chúng phân vùng tập huấn thành các tập con thuần túy (cross entropy nhỏ nhất). Sau đó, một cây quyết định sẽ tạo một danh sách các bước để phân vùng dữ liệu hiệu quả nhất và theo số bước nhanh nhất. Vì vậy, các đặc trưng cao nhất trong cây (gần node gốc nhất) có thể được coi là quan trọng nhất.

Trong Random Forest, mỗi cây thực hiện phân loại trên một tập con dữ liệu và đặc trưng khác nhau. Điều này giúp tránh outliers, đồng thời ngăn overfitting và giảm sự phụ thuộc lẫn nhau giữa các đặc trưng. Cuối cùng, việc trích xuất đặc trưng có thể được thực hiện bằng cách lấy trung bình mức độ quan trọng trên tất cả các cây. Sau khi áp dụng phương pháp này trên bộ dữ liệu 'Hotel Booking Demand', ta trích xuất được 30 đặc trưng (Bảng III).

Ta tách tập dữ liệu ban đầu thành 2 tập mới: tập Important gồm 30 thuộc tính quan trọng nhất và tập Rest gồm các thuộc tính "không quan trọng" còn lại. Cả 4 thuật toán khác nhau đều cho độ chính xác trên tập Important xấp xỉ (thậm chí cao hơn vì được loại bỏ những đặc trưng phụ thuộc) tập Full, trong khi độ chính xác sử dụng những thuộc tính không quan trọng thấp hơn hẳn so với 2 tập trên. Điều này chứng tỏ 30 thuộc

tính đã chọn ảnh hưởng quan trọng đến quyết định hủy đặt phòng khách sạn của khách hàng, và việc chạy mô hình chỉ trên các đặc trưng được trích xuất không ảnh hưởng đến độ chính xác của mô hình.

Đặc trưng	Độ quan trọng
status_minus_arrival_date	0.108952
lead_time	0.086364
deposit_type_No Deposit	0.067821
country_PRT	0.058496
deposit_type_Non Refund	0.050479
stays_in_week_nights	0.049168
total_of_special_requests	0.046074
adr	0.045246
price_per_guest	0.043134
arrival_date_day_of_month	0.034310

Bảng III: 10 đặc trưng quan trọng nhất

Mô hình	Độ đo F1		
	Full	Important	Rest
Logistic Regression	0.797	0.777	0.651
K-Nearest Neighbors	0.819	0.872	0.652
Decision Tree	0.927	0.929	0.681
Neural Network	0.961	0.969	0.687

Bảng IV: So sánh độ đo F1 của các mô hình

Mô hình	Time		
	Full	Important	Rest
Logistic Regression	5.331	3.014	12.35
K-Nearest Neighbors	0.031	0.013	0.027
Decision Tree	3.409	0.860	3.586
Neural Network	323.23	149.49	420.97

Bảng V: So sánh thời gian huấn luyện của 4 mô hình

Kết luận, trích xuất đặc trưng không chỉ giúp thu gọn kích thước khối dữ liệu, giúp người dùng suy diễn nhiều hơn từ tập dữ liệu đó mà còn tiết kiệm thời gian huấn luyện thuật toán.

C. Thử nghiệm trên các mô hình

	0	1
Logistic Regression	0.85	0.70
K-Nearest Neighbors	0.91	0.84
Decision Tree	0.94	0.90
Neural Network	0.98	0.96

Bảng VI: Độ đo F1 theo nhãn

Thuật toán Logistic Regression và K-Nearest Neighbors hoạt động tốt hơn nhiều trên nhãn 0 so với nhãn 1. Độ đo F1 trên nhãn 0 của 2 thuật toán lần lượt là 85% và 91%, trong khi đối với nhãn 1, độ đo F1 chỉ đạt được 70% và 84%. Thuật toán Decision Tree và Neural Network cũng có xu hướng phân loại nhãn 0 tốt hơn, nhưng sự chênh lệch này không rõ ràng như 2 thuật toán trên.

Trong 4 mô hình thử nghiệm, Neural Network có khả năng phân loại tốt nhất với f1_score là 96.9% và AUC là 0.966, đều cao hơn so với các mô hình còn lại.

Mô hình	Độ đo F1	AUC
Logistic Regression	0.777177	0.766493
K-Nearest Neighbors	0.873381	0.864802
Decision Tree	0.919828	0.921089
Neural Network	0.968865	0.965753

Bảng VII: So sánh độ đo F1 và AUC của 4 mô hình

D. Tối ưu hóa mô hình sử dụng Grid Search

Grid Search là một kỹ thuật giúp tìm kiếm tham số phù hợp cho mô hình đối với một bộ dữ liệu cụ thể. Trong sklearn chúng ta có thể sử dụng GridSearchCV để tạo không gian tham số. Vì Neural Network có khả năng phân loại tốt nhất đối với f1_score nên ta chọn mô hình Neural Network để tinh chỉnh bằng GridSearchCV.

Tham số		Độ đo F1	
alpha	hidden_layer_sizes	Trung bình	Độ lệch chuẩn
1	(50, 50, 50)	0.966995	0.001266
1	(100, 100)	0.963394	0.002440
0.1	(50, 50, 50)	0.970786	0.000973
0.1	(100, 100)	0.970246	0,001525

Bảng VIII: Tỉnh chỉnh mô hình Neraul Network

Sau khi tinh chỉnh mô hình Neural Network, bộ tham số tối ưu thu được là {'alpha': 0.1, 'hidden_layer_sizes': (50, 50, 50)}, cải thiện độ đo F1 lên 97.3%. Độ đo F1 trên tập huấn luyện xấp xỉ gần bằng độ đo F1 trên tập kiểm thử, do đó, mô hình đã huấn luyện không bị quá khớp (overfitting) hay chưa khớp (underfitting). Bên cạnh đó, cross validation score gần bằng độ đo trên tập kiểm thử chứng tỏ dữ liệu có tính đại diện cho tất cả dữ liệu muốn dự đoán. Vì thế, mô hình Neural Network có thể tổng quát hóa tốt và có khả năng dự đoán tốt trên tập dữ liệu mới.

Độ đo F1 tốt nhất trên tập train	0.9745205578391842
Độ đo F1 tốt nhất trên tập test	0.9687931796260842
Cross validation score	0.9727570028980846

Bảng IX: Độ đo F1 sau khi sử dụng bộ tham số tối ưu

V. Phân tích lỗi, hướng phát triển

A. Phân tích lỗi

	Predicted Not Canceled	Predicted Canceled
Actually Not Canceled	14878	125
Actually Canceled	562	8277

Bảng X: Ma trận nhầm lẫn của mô hình Neural Network

Dựa trên ma trận nhầm lẫn, 99.17% lượt đặt phòng không bị hủy và 93.64% lượt đặt phòng bị hủy được phân loại đúng. Có 125 lượt đặt phòng bị phân loại nhầm thành bị hủy. Sự sai lầm này khiến cho nhân viên khách sạn chưa thực sự sẵn sàng để đón khách hoặc trong trường hợp xấu hơn, phòng sẽ bị chuyển sang cho khách hàng khác. Có 562 lượt đặt phòng được dự đoán là không bị hủy trong khi thực chất nó bị hủy.

Dự đoán sai sẽ gây lãng phí nguồn lực khách sạn và bị mất đi một số khách hàng.

Để hiểu rõ hơn về một số lý do dẫn đến sự nhầm lẫn của mô hình Neural Network, ta khảo sát một vài lượt đặt phòng bị phân loại sai:

Index	Thực tế	Dự đoán	Previous_ cancellations	booking_ chances	required_car_ parking_spaces	total_of_ special_requests
3667	1	0	0	0	0	1
20879	1	0	0	0	0	1
5290	0	1	0	0	0	0
18913	0	1	0	0	0	0

Bảng XI: Một số mẫu quan sát dự đoán sai

Đối với 2 trường hợp 3667 và 20879, mô hình đã phân loại lượt đặt phòng không bị hủy trong khi thực tế nó đã bị hủy. Những lượt đặt phòng trên có lịch sử hủy phòng bằng 0. Thông thường nếu số lần đã từng hủy phòng càng ít thì khả năng hủy phòng lần này không cao, đây có thể là một trong những nguyên nhân gây ra sự nhầm lẫn. Các lượt đặt phòng mã số 5290 và 18913 đều bị phân loại nhầm thành bị hủy. Dựa trên số yêu cầu đặc biệt, yêu cầu về chỗ đậu xe và số lần thay đổi thông tin đặt phòng bằng 0, mô hình sẽ có xu hướng dự đoán lượt này bị hủy trong khi thực chất nó không bị hủy. Khi quan sát thêm các mẫu dữ liệu bị phân loại sai, ta sẽ khám phá được thêm nhiều nguyên nhân khác gây ra sự nhầm lẫn.

B. Hướng phát triển

Thao tác tiền xử lý dữ liệu trong đồ án này còn nhiều hạn chế và chưa thực sự hợp lý. Bộ dữ liệu hiện đang bị mất cân bằng nhẹ với tỷ lệ 63:37, có thể cân bằng lại nhãn để cải thiện quá trình học.

Hơn nữa, phương pháp Dummy Encoding chưa phù hợp để xử lý biến category trong bộ dữ liệu này. Khi biến categorical có một số lượng lớn các giá trị khác nhau, hoặc trong dataset có nhiều biến categorical, chúng ta sẽ cần đến rất nhiều biến dummy để mã hóa dữ liệu. Ví dụ biến 'country' có 178 giá trị khác nhau sẽ tương đương với 178 biến mới. Quá nhiều biến dummny được tạo ra dễ gây tràn bộ nhớ và làm giảm hiệu quả máy học. Do có cần áp dụng một cách mã hóa khác hiệu quả hơn đối với biến categorical.

Trong thực tế, các khách sạn cần biết cụ thể hơn những thuộc tính quan trọng ảnh hưởng như thế nào, có mối tương quan dương hay âm với khả năng hủy đặt phòng của khách hàng. Thuật toán Random Forest chưa đáp ứng được điều này mà nó chỉ đưa ra danh sách những thuộc tính quan trọng nhất, có thể sử dụng mô hình hồi quy Logistic hay hồi quy Lasso để khám phá sâu hơn về các mối tương quan đó. Một hệ thống kết hợp kỹ thuật chọn lựa thuộc tính và một mô hình có khả năng phân loại tốt như Neural Network có thể được đưa vào sử dụng trong các khách sạn, khu nghỉ dưỡng để phục vụ cho việc chuẩn bị tài nguyên và nhân lực.

VI. KẾT LUẬN

Dựa trên kết quả thử nghiệm huấn luyện 4 thuật toán Logistic Regression, K-Nearest Neighbors, Decision Tree và Neural Network trên bộ dữ liệu Hotel Demand Booking; và bằng cách sử dụng thuật toán Random Forest để chọn ra 30 thuộc tính quan trọng, ta thu được mô hình Neural Network có hiệu năng cao nhất với độ đo F1 lên đến 97.3% sau khi tinh chỉnh bằng Grid Search. Tuy còn nhiều hạn chế nhưng nhìn chung với khả năng phân loại tốt, chúng em cho rằng mô hình nêu trên có thể đưa vào áp dụng trong thực tế.

TÀI LIỆU

[1] N. Antonio, A. de Almeida, and L. Nunes, "Hotelbooking demand datasets," Data in Brief, vol.22, pp.41–49, 2019. doi: https://doi.org/10.1016/j.dib.2018.11.126

[2] https://www.kaggle.com/datasets/jessemostipak/hotel-booking-demand

[3] C. Y. J. Peng, K. L. Lee, and G. M. Ingersoll, "An Introduction to Logistic Regression Analysis and Reporting," J. Educ. Res., vol.96, no.1, pp.3–14, 2002, doi: 10.1080/00220670209598786

[4] S. Zhang, "Cost-Sensitive KNN Classification," Neurocomputing, vol.391, no.xxxx, pp.234–242, 2020,doi: 10.1016/j.neucom.2018.11.101.

[5] X. E. Pantazi, D. Moshou, and D. Bochtis, Artificial intelligence in agriculture. 2020.

[6] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, Classification and regression trees. 1999.

[7] L. E. Raileanu and K. Stoffel, "Theoretical Comparison Between the Gini Index and Information Gain Criteria∗", no. 2100, pp. 77–93, 2004.

[8] Misuk Lee, Xinpan Mu, Yiqiao Zhang, "A machine learning approach to improving forcasting accuracy of Hotel Demand: A comparative analysis of neural networks and trditional models", Issues in Information Systems, Volume 21, Issue 1, pp. 12-21, 2020, doi: https://doi.org/10.48009/1_iis_2020_12-21

BẢNG PHÂN CÔNG

20520134 - Nguyễn Thị Hoàng Anh	**Mức độ hoàn thành (%)**
Tìm hiểu Logistic Regression Coding, làm báo cáo các phần liên quan Tổng hợp bài báo cáo Thuyết trình	100
20520165 - Nguyễn Hà Dung	**Mức độ hoàn thành (%)**
Làm sạch và phân tích dữ liệu Tìm hiểu Random Forest Importance, Decision Tree Coding, làm báo cáo các phần liên quan Phân tích lỗi, hướng phát triển Làm Powerpoint	100
20522052 - Nguyễn Minh Trí	**Mức độ hoàn thành (%)**
Tìm hiểu kNN, Neural Network Coding, làm báo cáo các phần liên quan Tinh chỉnh mô hình Kết luận	100