TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA TOÁN - ỨNG DỤNG

TIỂU LUẬN CUỐI KỲ

Môn Máy Học

Đề tài:

NGHIÊN CỨU MÔ HÌNH HỌC MÁY ĐỂ DỰ ĐOÁN GIÁ PHÒNG TRUNG BÌNH HÀNG NGÀY (ADR) TRONG NGÀNH KHÁCH SẠN

Họ và tên sinh viên: Phạm Hoàng Tiến

MSSV: 3123580051

Lớp: DDU1231

Tên Giảng Viên: TS. Vũ Ngọc Thanh Sang

TP. Hồ Chí Minh, tháng 5 năm 2025

LỜI CẢM ƠN

Trước tiên, em xin được gửi một lời cảm ơn sâu sắc và lòng biết ơn chân thành nhất dành đến cho thầy Vũ Ngọc Thanh Sang vì đã chuẩn bị và đầu tư những bài giảng tâm huyết, nhằm truyền đạt những kiến thức phân tích và lập trình bổ ích mà không kém phần thú vị trong suốt thời gian thực học môn Máy Học học kì vừa qua.

Em xin chân thành cảm ơn!

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC TỪ VIẾT TẮT

Chữ viết tắt	**Nguyên Nghĩa**
ADR	Average Daily Rate

PHẦN MỞ ĐẦU

1. Lý do chọn đề tài

Trong bối cảnh cạnh tranh gay gắt của ngành khách sạn, việc dự đoán chính xác Giá Phòng Trung bình Hàng ngày (ADR) đóng vai trò cực kỳ quan trọng đối với sự thành công và lợi nhuận của doanh nghiệp. ADR không chỉ là một chỉ số tài chính cốt lõi mà còn là nền tảng để xây dựng các chiến lược định giá tối ưu và quản lý doanh thu hiệu quả. Tuy nhiên, bài toán dự đoán ADR đặt ra nhiều thách thức đáng kể do tính biến động cao và sự phụ thuộc vào nhiều yếu tố phức tạp như mùa vụ, sự kiện đặc biệt và hành vi khách hàng. Sự phức tạp này đòi hỏi các phương pháp phân tích và dự báo tiên tiến.

Xuất phát từ tầm quan trọng thực tế và tính thời sự của vấn đề này trong ngành khách sạn, cùng với sự quan tâm sâu sắc đến lĩnh vực học máy và khả năng ứng dụng của nó trong giải quyết các bài toán kinh doanh phức tạp, em đã lựa chọn đề tài 'Nghiên cứu Mô hình Học máy để Dự đoán Giá Phòng Trung bình Hàng ngày (ADR) trong Ngành Khách sạn'. Đề tài này cho phép em vận dụng và mở rộng kiến thức đã học về các thuật toán học máy như XGBoost, Random Forest, và các mô hình Ensemble khác, đồng thời đóng góp vào việc tìm kiếm các giải pháp dự báo hiệu quả hơn cho ngành khách sạn

2. Mục tiêu nghiên cứu

Mục tiêu của nghiên cứu này là xây dựng một mô hình dự đoán ADR với độ chính xác cao, sử dụng các thuật toán học máy tiên tiến như XGBoost, Random Forest và các phương pháp khác, nhằm đưa ra những dự báo chính xác về giá phòng trong ngành khách sạn.

3. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu:

Đối tượng nghiên cứu của đề tài là các yếu tố ảnh hưởng đến chỉ số giá phòng trung bình (ADR - Average Daily Rate) trong lĩnh vực khách sạn, cũng như các mô hình học máy được sử dụng để dự đoán giá trị ADR. Nghiên cứu tập trung vào việc phân tích, đánh giá tác động của các biến đầu vào đến ADR và so sánh hiệu quả của các thuật toán dự đoán khác nhau.

Phạm vi nghiên cứu:

Đề tài tập trung vào việc dự đoán Average Daily Rate (ADR) trong ngành khách sạn, dựa trên bộ dữ liệu đặt phòng của khách sạn thành phố và khu nghỉ dưỡng. Về không

gian, nghiên cứu giới hạn trong lĩnh vực khách sạn. Về thời gian, phạm vi phụ thuộc vào giai đoạn thu thập dữ liệu có sẵn trong tập dữ liệu

4. Phương pháp nghiên cứu

Nghiên cứu áp dụng quy trình khoa học gồm các bước: thu thập, xử lý dữ liệu, xây dựng và đánh giá mô hình học máy để dự đoán ADR. Dữ liệu được lấy từ bộ dữ liệu đặt phòng có sẵn (dữ liệu thứ cấp). Quá trình tiền xử lý bao gồm làm sạch, xử lý giá trị thiếu và loại bỏ các bản ghi không hợp lệ. Các mô hình được xây dựng gồm Linear Regression, Random Forest, XGBoost, …. Nhằm so sánh hiệu quả giữa các thuật toán. Mức độ chính xác của mô hình được đánh giá qua các chỉ số như RMSE, MAE và R^2. Toàn bộ quá trình được thực hiện bằng Python với các thư viện học máy chuyên dụng như Scikit-learn [2], XGBoost, LightGBM và CatBoost.

5. Cấu trúc bài tiểu luận

Để làm rõ mục tiêu nghiên cứu và giải quyết bài toán đặt ra, bài tiểu luận được cấu trúc thành các chương như sau:

Chương 1: Cơ sở lý thuyết – Giới thiệu các khái niệm nền tảng như chỉ số ADR, tổng quan về học máy, bài toán hồi quy và các thuật toán sử dụng, trình bày các chỉ số đánh giá mô hình như RMSE, MAE và R^2..

Chương 2: Dữ liệu & Thực nghiệm – Trình bày chi tiết về bộ dữ liệu đặt phòng khách sạn, các bước tiền xử lý như xử lý giá trị thiếu, dữ liệu trùng lặp, ngoại lệ, và lỗi logic, feature engineering giúp cải thiện độ chính xác của mô hình.

Chương 3: Phân tích và trực quan hóa dữ liệu – Phân tích các yếu tố ảnh hưởng đến ADR thông qua biểu đồ trực quan, bao gồm ảnh hưởng của loại khách sạn, thời gian đặt phòng, mùa, loại bữa ăn, kênh phân phối, nhóm khách quay lại và xu hướng ADR theo thời gian.

Chương 4: Xây dựng và Phát triển mô hình dự đoán – Chương này trình bày quy trình xây dựng mô hình dự đoán ADR gồm các bước: chuẩn bị và xử lý dữ liệu, mã hóa biến, lựa chọn đặc trưng và chuẩn hóa. Mô hình học máy được triển khai. Được tiếp tục tối ưu siêu tham số bằng Optuna và phân tích sâu hơn qua SHAP và RFECV để nâng cao độ chính xác dự đoán..

Kết luận và hướng phát triển – Tổng kết nội dung nghiên cứu, đóng góp chính và đề xuất các hướng mở rộng trong tương lai.

PHẦN NỘI DUNG

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1. Ngành khách sạn & chỉ số ADR

Ngành khách sạn giữ vai trò quan trọng trong kinh tế dịch vụ, nơi quản lý doanh thu là yếu tố then chốt. Trong đó, ADR (Average Daily Rate) – Giá phòng Trung bình Hàng ngày – là chỉ số cốt lõi phản ánh doanh thu trung bình mỗi phòng mỗi ngày.

Dự đoán ADR chính xác giúp khách sạn xây dựng chiến lược định giá hợp lý, tối ưu hóa lợi nhuận và hỗ trợ ra quyết định kinh doanh hiệu quả. Tuy nhiên, bài toán này đầy thách thức do dữ liệu khách sạn biến động mạnh theo mùa, sự kiện và hành vi khách hàng phức tạp. Dù vậy, việc nghiên cứu và giải quyết bài toán dự đoán ADR vẫn mang lại giá trị thực tiễn cao.

1.2. Học máy & bài toán hồi quy

Học máy (Machine Learning) là một nhánh của Trí tuệ nhân tạo, cho phép máy tính học từ dữ liệu để thực hiện các tác vụ như phân loại, hồi quy hoặc phân cụm mà không cần lập trình tường minh. Có hai dạng học phổ biến: học có giám sát (dữ liệu có nhãn) và học không giám sát (dữ liệu không nhãn).

Bài toán dự đoán ADR trong nghiên cứu thuộc học có giám sát, cụ thể là bài toán hồi quy, vì ADR là một giá trị liên tục. Quá trình giải quyết bài toán học máy bao gồm các bước: thu thập và xử lý dữ liệu, chọn và huấn luyện mô hình, tinh chỉnh tham số và đánh giá hiệu quả bằng các chỉ số như RMSE, MAE, R^2.

1.3. Các thuật toán sử dụng (LR, RF, XGBoost,...)

1. Hồi quy tuyến tính (Linear Regression)

Mục tiêu: Dự đoán giá trị liên tục dựa trên mối quan hệ tuyến tính giữa các biến.

Công thức mô hình:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon$$

Trong đó:

- $\hat{y}$**:** giá trị dự đoán
- x_i**:** đặc trưng đầu vào
- β_i**:** hệ số hồi quy
- ε**:** sai số

Cơ chế học: Tối ưu hàm mất mát bình phương sai số (Mean Squared Error – MSE).

2. Random Forest

Mục tiêu: Tăng độ chính xác và giảm overfitting bằng cách kết hợp nhiều cây quyết định.

$$\hat{y} = \frac{1}{n}\sum_{i=1}^{n} h_i\,(x)$$

Trong đó:

- $\hat{\boldsymbol{y}}$ là giá trị dự đoán cuối cùng.
- $\boldsymbol{n}$ là số lượng cây trong rừng.
- $\boldsymbol{h_i(x)}$ là dự đoán của cây thứ i đối với mẫu x.

Nguyên lý:

- Dữ liệu được lấy mẫu ngẫu nhiên (bootstrapping).
- Mỗi cây học từ một tập con đặc trưng tại mỗi nút.
- Dự đoán đầu ra là trung bình kết quả của tất cả các cây (đối với hồi quy).

3. XGBoost (Extreme Gradient Boosting) [3]

Mục tiêu: Tối ưu hóa hiệu suất học Boosting bằng kỹ thuật gradient và chính quy hóa.

Hàm mục tiêu:

$$\mathcal{L} = \sum_{i=1}^{n} l\,(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega\,(f_k)$$

Trong đó:

- $\boldsymbol{l}$**:** hàm mất mát (thường là MSE)
- $\boldsymbol{\Omega(f_k)}$**:** thành phần chính quy hóa (L1 và L2)

Đặc điểm nổi bật: Chính quy hóa, pruning, xử lý missing values tốt, hỗ trợ tính toán song song.

4. CatBoost [5]

Mục tiêu: Tối ưu Gradient Boosting cho dữ liệu phân loại (categorical features).

CatBoost dựa trên ý tưởng **Boosting theo gradient**, trong đó mô hình được xây dựng dần theo từng bước lặp (iteration):

$$\hat{y}^{(t)}(x) = \hat{y}^{(t-1)}(x) + \eta \cdot h_t(x)$$

Trong đó:

- $\hat{\boldsymbol{y}}^{(t)}(\boldsymbol{x})$ là giá trị dự đoán tại vòng lặp thứ t
- $\hat{\boldsymbol{y}}^{(0)}(\boldsymbol{x})$ là giá trị khởi tạo (thường là trung bình)
- $\boldsymbol{h_t}(\boldsymbol{x})$ là cây quyết định được huấn luyện để khớp với phần dư (residual) tại vòng lặp thứ t
- $\boldsymbol{\eta}$ là hệ số học (learning rate)
- Số vòng lặp (iterations) được định nghĩa trong mô hình

Nguyên lý:

- Sử dụng ordered boosting và target encoding có kiểm soát để tránh rò rỉ dữ liệu.
- Tự động xử lý đặc trưng phân loại mà không cần mã hóa thủ công.
- Cây đối xứng (oblivious trees) giúp mô hình nhanh và ổn định.

5. LightGBM (Light Gradient Boosting Machine) [4]

Mục tiêu: Boosting hiệu quả trên tập dữ liệu lớn.

Tương tự như CatBoost hay XGBoost, LightGBM cũng dựa trên cơ chế Boosting theo gradient. Mỗi cây được xây dựng để học phần dư (residuals) của các cây trước đó.

$$\hat{y}^{(t)}(x) = \hat{y}^{(t-1)}(x) + \eta \cdot h_t(x)$$

Trong đó:

- $\hat{\boldsymbol{y}}^{(t)}(\boldsymbol{x})$ là dự đoán tại vòng lặp thứ t
- $\boldsymbol{\eta}$ là **learning rate**
- $\boldsymbol{h_t}(\boldsymbol{x})$ là cây quyết định thứ t, được huấn luyện để giảm hàm mất mát

Kỹ thuật chính:

- GOSS: Giữ lại các mẫu có gradient lớn, giúp học tốt hơn.
- EFB: Gom các đặc trưng phân tán thành một đặc trưng duy nhất.
- Xây dựng cây theo leaf-wise thay vì level-wise, tăng tốc hội tụ.

1.4. Các chỉ số đánh giá mô hình (RMSE, MAE, R^2)

Trong các bài toán hồi quy như dự đoán ADR (Average Daily Rate), việc đánh giá độ chính xác của mô hình là rất quan trọng. Các chỉ số phổ biến được sử dụng gồm:

RMSE (Root Mean Squared Error) – Căn bậc hai của sai số bình phương trung bình

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

RMSE phạt nặng hơn cho các sai số lớn do bình phương sai số. Đây là chỉ số nhạy với ngoại lệ, giúp phát hiện mô hình có dự đoán lệch nhiều ở một số điểm.

MAE (Mean Absolute Error):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

$\boldsymbol{R^2}$ (Hệ số xác định): Đo lường mức độ phù hợp của mô hình với dữ liệu.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Chỉ số R^2 phản ánh tỷ lệ phương sai của dữ liệu đầu ra được mô hình giải thích. Giá trị càng gần 1 thì mô hình càng tốt. Nếu $R^2 = 1$, mô hình dự đoán hoàn hảo; nếu $R^2 = 0$, mô hình không giải thích được gì so với trung bình.

CHƯƠNG 2: DỮ LIỆU & THỰC NGHIỆM

2.1. Mô tả bộ dữ liệu

2.1.1. Nguồn và giới thiệu dữ liệu.

Nguồn dữ liệu được lấy từ Kaggle [1].

Bộ dữ liệu được sử dụng trong dự án này là dữ liệu đặt phòng khách sạn, được thu thập từ hai loại hình khách sạn: Resort Hotel và City Hotel. Dữ liệu phản ánh các giao dịch đặt phòng thực tế trong khoảng thời gian từ năm 2015, bao gồm thông tin chi tiết về từng lượt đặt phòng, trạng thái đặt phòng, đặc điểm khách hàng, các yêu cầu đặc biệt, kênh phân phối, cũng như các yếu tố liên quan đến việc hủy đặt phòng.

2.1.2. Quy mô dữ liệu

Code:

```
print(df.shape)
```

Output:

```
(119390, 32)
```

Bộ dữ liệu có quy mô lớn với tổng cộng 119.390 bản ghi (dòng dữ liệu), mỗi bản ghi tương ứng với một giao dịch đặt phòng. Mỗi bản ghi chứa 32 thuộc tính (cột dữ liệu), bao gồm cả các biến định lượng (số lượng) và định tính (phân loại). Các thuộc tính này cung cấp thông tin đa chiều về khách hàng, thời gian đặt phòng, loại phòng, giá trị giao dịch, trạng thái đặt phòng, cũng như các đặc điểm liên quan đến hành vi và nhu cầu của khách hàng.

2.1.3. Ý nghĩa các đặc trưng chính

- **Hotel:** (H1 = Khu nghỉ dưỡng, H2 = Khách sạn thành phố)

- **is_canceled:** Giá trị cho biết liệu đặt phòng có bị hủy hay không (1: bị hủy, 0: không bị hủy)

- **lead_time:** Số ngày đã trôi qua giữa ngày nhập dữ liệu đặt phòng vào hệ thống quản lý khách sạn (PMS) và ngày đến của khách

- **arrival_date_year:** Năm của ngày đến

- **arrival_date_month:** Tháng của ngày đến

- **arrival_date_week_number:** Số tuần trong năm của ngày đến

- **arrival_date_day_of_month:** Ngày trong tháng của ngày đến

- **stays_in_weekend_nights:** Số đêm cuối tuần (Thứ Bảy hoặc Chủ Nhật) mà khách đã ở hoặc đã đặt phòng ở khách sạn

- **stays_in_week_nights:** Số đêm trong tuần (Từ Thứ Hai đến Thứ Sáu) mà khách đã ở hoặc đã đặt phòng ở khách sạn

- adult: Số người lớn

- **Children:** Số trẻ em

- **babies:** Số trẻ sơ sinh

- **meal:** Loại bữa ăn đã đặt. Các loại được trình bày theo các gói bữa ăn tiêu chuẩn trong ngành khách sạn: Undefined/SC – không có gói bữa ăn; BB – Bed & Breakfast (Chỉ bữa sáng); HB – Half board (bữa sáng và một bữa ăn khác – thường là bữa tối); FB – Full board (bữa sáng, bữa trưa và bữa tối)

- country: Quốc gia xuất xứ. Các danh mục được biểu diễn theo định dạng ISO 3155–3:2013

- market_segment: Phân khúc thị trường. Trong các danh mục, thuật ngữ "TA" có nghĩa là "Đại lý du lịch" và "TO" có nghĩa là "Nhà điều hành tour"

- distribution_channel: Kênh phân phối đặt phòng. Thuật ngữ "TA" có nghĩa là "Đại lý du lịch" và "TO" có nghĩa là "Nhà điều hành tour"

- is_repeated_guest: Giá trị cho biết liệu tên khách đặt phòng có phải là của khách quay lại hay không (1: khách quay lại, 0: khách lần đầu)

- previous_cancellations: Số lần đặt phòng trước đó đã bị hủy bởi khách hàng trước khi có đặt phòng hiện tại

- previous_bookings_not_canceled: Số lần đặt phòng trước đó không bị hủy bởi khách hàng trước khi có đặt phòng hiện tại

- reserved_room_type: Mã loại phòng đã đặt. Mã được sử dụng thay cho tên phòng để bảo vệ tính ẩn danh.

- assigned_room_type: Mã loại phòng đã được chỉ định cho đặt phòng. Đôi khi loại phòng được chỉ định khác với loại phòng đã đặt do lý do hoạt động của khách sạn (ví dụ: overbooking) hoặc yêu cầu của khách. Mã được sử dụng thay cho tên phòng để bảo vệ tính ẩn danh.

- booking_changes: Số lần thay đổi/sửa đổi đã được thực hiện đối với đặt phòng từ khi đặt phòng được nhập vào PMS cho đến khi khách check-in hoặc hủy bỏ

- deposit_type: Chỉ định liệu khách hàng có đặt cọc để đảm bảo đặt phòng hay không. Biến này có thể có ba loại: Không đặt cọc – không có đặt cọc; Không hoàn lại – đã đặt cọc bằng giá trị của tổng chi phí lưu trú; Hoàn lại – đã đặt cọc với giá trị thấp hơn tổng chi phí lưu trú.

- agent: ID của đại lý du lịch thực hiện đặt phòng

- company: ID của công ty/tổ chức thực hiện đặt phòng hoặc chịu trách nhiệm thanh toán cho đặt phòng. ID được sử dụng thay cho tên để bảo vệ tính ẩn danh.

- days_in_waiting_list: Số ngày mà đặt phòng đã ở trong danh sách chờ trước khi được xác nhận với khách hàng

- customer_type: Loại đặt phòng, gồm bốn danh mục: Contract – khi đặt phòng có hợp đồng hoặc một loại hợp đồng khác liên quan; Group – khi đặt phòng liên quan đến một nhóm; Transient – khi đặt phòng không phải là một phần của nhóm hoặc hợp đồng, và không liên quan đến các đặt phòng khác; Transient-party – khi đặt phòng là một phần của đặt phòng tạm thời, nhưng liên quan đến ít nhất một đặt phòng tạm thời khác.

- adr: Giá trung bình hàng ngày, được xác định bằng cách chia tổng số giao dịch lưu trú cho tổng số đêm lưu trú.

- required_car_parking_spaces: Số chỗ đậu xe yêu cầu bởi khách hàng.

- total_of_special_requests: Số lượng yêu cầu đặc biệt mà khách hàng đã yêu cầu (ví dụ: giường đôi hoặc phòng cao tầng).

- reservation_status: Trạng thái cuối cùng của đặt phòng, có ba loại: Canceled – đặt phòng đã bị hủy bởi khách hàng; Check-Out – khách đã check-in nhưng đã rời đi; No-Show – khách không check-in và không thông báo lý do với khách sạn.

- reservation_status_date: Ngày mà trạng thái cuối cùng được thiết lập. Biến này có thể được sử dụng cùng với ReservationStatus để hiểu khi nào đặt phòng bị hủy hoặc khi nào khách đã check-out khỏi khách sạn.

2.2. Tiền xử lý và chuẩn hóa dữ liệu

2.2.1 Chuẩn hoá tên cột

Code:

```
# chuẩn hoá tên cột
df.columns = [re.sub("[ -]","_", c).lower() for c in
df.columns]
```

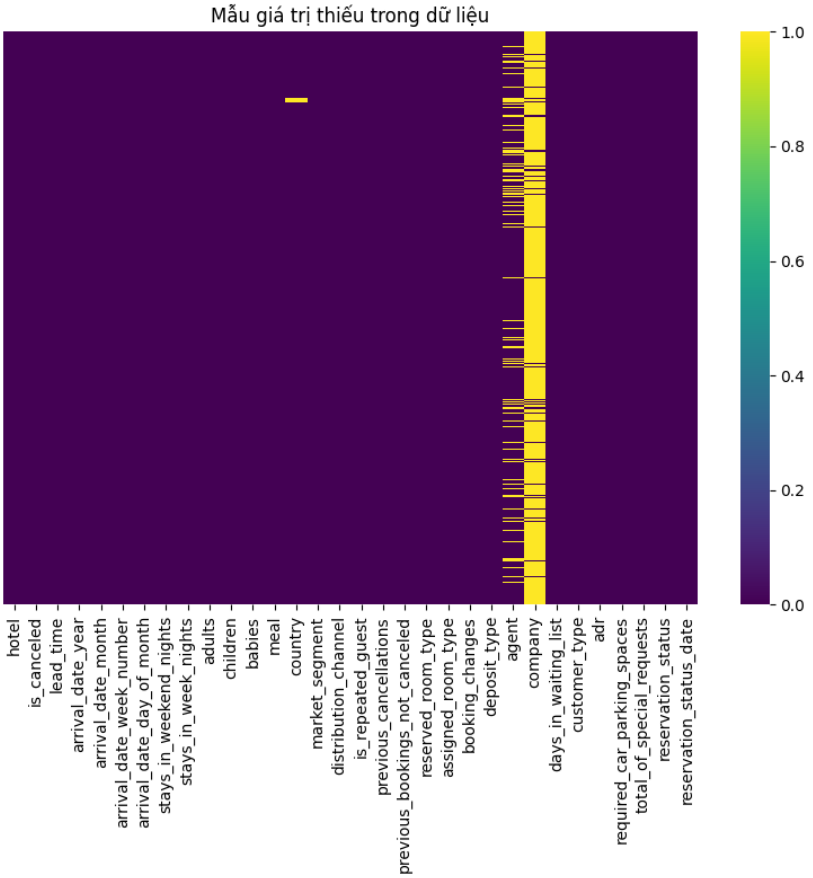
Thay thế toàn bộ các ký tự khoảng trắng và dấu gạch ngang trong tên cột bằng dấu gạch dưới. Đồng thời, tất cả các ký tự trong tên cột cũng được chuyển về dạng chữ thường.

Việc chuẩn hóa này giúp loại bỏ các ký tự đặc biệt hoặc không thống nhất, vốn có thể gây khó khăn khi truy xuất dữ liệu bằng mã lệnh. Ví dụ, các tên cột như "Arrival Date" hoặc "Lead-Time" sẽ được chuyển thành "arrival_date" và "lead_time". Điều này không chỉ giúp code trở nên dễ đọc, dễ bảo trì mà còn giảm thiểu nguy cơ phát sinh lỗi do sự khác biệt về định dạng tên cột trong quá trình phân tích và xây dựng mô hình.

2.2.2. Xử lý giá trị thiếu

Kiểm tra số lượng giá trị null trong từng cột để xác định cần xử lý thiếu dữ liệu ở đâu.

Heatmap



Hình 2. 1 Mẫu giá trị thiếu trong dữ liệu

In ra số liệu về những cột thiếu

Để đánh giá mức độ thiếu dữ liệu trong từng cột, nhóm tiến hành thống kê số lượng và tỷ lệ phần trăm các giá trị bị thiếu trong toàn bộ tập dữ liệu. Kết quả được sắp xếp giảm dần theo tỷ lệ phần trăm thiếu nhằm dễ dàng xác định các cột cần xử lý.

Code:

```
missing_values = df.isnull().sum()
missing_values_percent = (missing_values / len(df)) * 100
missing_summary = pd.DataFrame({
    'Missing Values': missing_values,
    'Percentage': missing_values_percent
})
missing_summary = missing_summary[missing_summary['Missing
Values'] > 0].sort_values(by='Percentage', ascending=False)
print("Thống kê giá trị thiếu:")
print(missing_summary)
```

Output:

```
Thống kê giá trị thiếu:
         Missing Values  Percentage
company          112593   94.306893
agent             16340   13.686238
country             488    0.408744
children              4    0.003350
```

Sau khi có thống kê về giá trị của những cột bị thiếu, bây giờ chúng ta sẽ thực hiện xử lý phù hợp với từng cột

Đối với cột **company:**

Vì Cột `**company**` thể hiện ID tổ chức/cty thực hiện đặt phòng hoặc thanh toán, những giá trị bị thiếu ở cột này có thể những người đặt phòng và thanh toán trực tiếp, việc bị thiếu hơn 94% giá trị, nên đã được loại bỏ khỏi dataset để tránh ảnh hưởng đến chất lượng phân tích và mô hình.

Code:

```
df = df.drop(['company'], axis = 1)
```

Đối với cột **agent:**

Biến agent là ID của đại lý du lịch đã thực hiện việc phòng nên những giá trị null này có thể là những phòng được đặt mà không thông qua đại lý nào hoặc là đại lý không nằm

trong danh sách. Vậy nên phương pháp dùng ở đây là set ID = 0 đối với những trường hợp này.

Đầu tiên sẽ xem có đại lý nào có ID = 0 không.

Code:

```
if 0 in df['agent'].unique():
    print("Có giá trị 0 trong cột 'agent'")
else:
    print("Không có giá trị 0 trong cột 'agent'")
```

Output:

```
Không có giá trị 0 trong cột 'agent'
```

Như vậy không có ID ở cột agent nào là 0, vậy chúng ta sẽ set giá trị agent cho những giá trị bị thiếu ở cột này

Code:

```
# set ID = 0 đối với những trường hợp thiếu.
df['agent'].fillna(0, inplace=True)
```

Đối với cột c**ountry:**

Những quốc gia trống này có thể là những quốc gia nằm ngoài danh sách hoặc những vùng lãnh thổ chưa được quốc tế công nhận nên sẽ thể hiện các quốc gia này bằng giá trị 'Unknown'

Code:

```
# Điền giá trị thiếu trong cột 'country' bằng 'Unknown'
df['country'].fillna('Unknown', inplace=True)
```

Đối với cột **Children:**

Các giá trị children bị thiếu có thể là trường hợp nhập thiếu, hoặc là không có đứa trẻ nào. Nên giải pháp cho trường hợp này sẽ dùng giá trị xuất hiện nhiều nhất

Code:

```
# Thay thế giá trị thiếu bằng giá trị xuất hiện nhiều nhất (mode) của chính cột `children`.
df['children'].fillna(df['children'].mode()[0], inplace=True)
```

Kiểm tra giá trị thiếu một lần nữa

```
Thống kê giá trị thiếu:
Empty DataFrame
Columns: [Missing Values, Percentage]
Index: []
```

Như vậy chúng ta đã thực hiện xong việc xử lý các giá trị thiếu ở các cột

2.2.3. Xử lý dữ liệu trùng lặp

Xử lý dữ liệu trùng lặp nhằm loại bỏ các bản ghi giống hệt nhau, tránh việc một thông tin bị tính lặp nhiều lần, giúp tăng độ chính xác, giảm độ thiên lệch và giảm tài nguyên

Code:

```
# Kiểm tra bản ghi trùng lặp
df_duplicates = df.duplicated().sum()
print(f"Số lượng bản ghi trùng lặp: {df_duplicates}")
```

Output:

```
Số lượng bản ghi trùng lặp: 32001
```

Chúng ta sẽ thực hiện xoá những giá trị trùng lặp này.

Code:

```
# Xóa tất cả các bản ghi trùng lặp
df = df.drop_duplicates()
print(f"Kích thước sau khi xóa trùng lặp: {df.shape}")
```

Output:

```
Kích thước sau khi xóa trùng lặp: (87389, 31)
```

Từ 119.390 dòng dữ liệu, sau khi thực hiện xoá trùng lặp. Bộ dữ liệu còn 87.389

2.2.4. Xử lý lỗi logic

Đối với giá trị ADR

ADR được tính dựa trên các giao dịch lưu trú thực tế, tức là chỉ áp dụng cho các đêm mà khách thực sự lưu trú (staying nights). Nếu một đặt phòng bị hủy (is_canceled = 1), khách không lưu trú, không có giao dịch lưu trú thực tế (vì khách không đến ở), nên giá trị ADR cho đặt phòng đó không tồn tại hoặc không có ý nghĩa trong bối cảnh tính toán thực tế.

Code:

```
# Kiểm tra số lượng đặt phòng bị hủy trước khi xóa
canceled_count = len(df[df['is_canceled'] == 1])
print(f"Số lượng đặt phòng bị hủy: {canceled_count}")
# Lọc bỏ các đặt phòng bị hủy (chỉ giữ lại is_canceled = 0)
df = df[df['is_canceled'] == 0]
# Kiểm tra số lượng mẫu sau khi xóa
print(f"Số lượng mẫu sau khi loại bỏ đặt phòng bị hủy:
{len(df)}")
```

Output:

```
Số lượng đặt phòng bị hủy: 24025
Số lượng mẫu sau khi loại bỏ đặt phòng bị hủy: 63364
```

Xem có tồn tại giá trị âm ở **adr** không, vì giá trị âm là một giá trị lỗi nhập liệu hoặc hệ thống. Sẽ cần loại bỏ để mang lại hiệu quả phân tích

Code:

```
# chúng ta sẽ xoá những phòng có giá trị âm này
df = df[df['adr'] >= 0]
```

Tiếp theo là kiểm tra các dòng có adr = 0

Code:

```
#  Kiểm tra số lượng giá trị ADR bằng 0
zero_adr_count = len(df[df['adr'] == 0])
```

```
# In kết quả
if zero_adr_count > 0:
    print(f"Có {zero_adr_count} giá trị ADR bằng 0.")
    print(df[df['adr'] == 0])  # Hiển thị các hàng có ADR = 0
else:
    print("Không có giá trị ADR nào bằng 0.")
```

Output:

```
Có 1593 giá trị ADR bằng 0.
```

Theo phân tích có thể thấy có 1593 dòng có giá trị adr bằng 0,các trường hợp này có thể là:

- Miễn phí hoặc khuyến mãi: Một số đặt phòng có thể được cung cấp miễn phí (ADR = 0) như một phần của chương trình khuyến mãi hoặc dịch vụ bổ sung.

- Lỗi dữ liệu: Giá trị 0 có thể là do nhập liệu sai hoặc thiếu thông tin về doanh thu.

- không tính phí: Có thể một số trường hợp không thu tiền (ví dụ, nhân viên nội bộ, khách VIP, hoặc lỗi hệ thống).

Những dòng có ADR = 0 được cung cấp miễn phí và không mang lại giá trị doanh thu, đồng thời có thể là dữ liệu sai hoặc thiếu thông tin cần thiết (ví dụ: thông tin về doanh thu, khách VIP, hoặc lỗi hệ thống). Việc loại bỏ giúp tập trung vào các trường hợp hợp lệ, có giá trị doanh thu thực tế để phân tích và xử lý hiệu quả hơn.

Code:

```
#  df = df[df['adr'] != 0]
```

Đối với giá trị thời gian.

Em đã tiến hành kiểm tra tính hợp lệ của các trường ngày tháng trong dữ liệu, bao gồm việc loại bỏ các bản ghi có ngày trong tháng lớn hơn 31, số tuần lớn hơn 53, và đặc biệt là kiểm tra tháng 2 không có ngày vượt quá 29. Việc này giúp đảm bảo dữ liệu đầu vào không chứa các giá trị bất hợp lý, từ đó nâng cao độ chính xác cho các phân tích tiếp theo.

```
Số dòng có thời gian lỗi:  (0, 31)
Kiểm tra số ngày của tháng 2 : (0, 31)
```

kiểm tra và loại bỏ các bản ghi có giá trị lead time âm, vì về mặt logic, khách hàng không thể đặt phòng sau ngày nhận phòng. Việc này giúp đảm bảo dữ liệu đầu vào hợp lý, nâng cao độ tin cậy cho các phân tích tiếp theo về hành vi đặt phòng của khách hàng.

```
Số dòng có lead time âm:  (0, 31)
```

Đối với các giá trị con người.

Để đảm bảo chất lượng dữ liệu, em đã kiểm tra các trường hợp đặt phòng bất hợp lý, bao gồm các phòng có tổng số khách bằng 0 và các phòng chỉ có trẻ em hoặc em bé mà không có người lớn đi kèm. Việc này giúp loại bỏ các bản ghi sai lệch, đảm bảo dữ liệu đầu vào cho các bước phân tích tiếp theo là hợp lệ và đáng tin cậy.

Code:

```
# Tổng số người trong phòng phải > 0
invalid_guests = df[
    (df['adults'] + df['children'] + df['babies']) == 0
]

# Trẻ em hoặc em bé không thể ở một mình
invalid_child_only = df[
    (df['adults'] == 0) &
    ((df['children'] > 0) | (df['babies'] > 0))
]
print('Số phòng đặt nhưng tổng số lượng người ở bằng 0:
',invalid_guests.shape)
print('Số phòng đặt nhưng chỉ có trẻ em:
',invalid_child_only.shape)
```

Output:

```
Số phòng đặt nhưng tổng số lượng người ở bằng 0:  (31, 31)
Số phòng đặt nhưng chỉ có trẻ em:  (134, 31)
```

Xem xét qua số lượng người ở bằng 0 thì ở trường này chúng ta sẽ xem xét nếu phòng này chưa huỷ chúng ta sẽ set giá trị adults = 1. Còn đối với những trường hợp không có người lớn nào thì chúng ta sẽ set adults = 1, vì phòng đặt phải có ít nhất một người lớn

Code:

```
for idx, row in invalid_guests.iterrows():
    if row['is_canceled'] != 1:
        # Nếu chưa huỷ, set adults = 1 (giả định có 1 người lớn)
        df.at[idx, 'adults'] = 1

for idx, row in invalid_child_only.iterrows():
    # Set adults = 1 cho tất cả các trường hợp này
    df.at[idx, 'adults'] = 1

# In ra số lượng dòng đã sửa để kiểm tra
print('Số phòng không người ở đã sửa:', invalid_guests[invalid_guests['is_canceled'] != 1].shape[0])
print('Số phòng chỉ có trẻ em đã sửa:', invalid_child_only.shape[0])
```

Output:

```
Số phòng không người ở đã sửa: 31
Số phòng chỉ có trẻ em đã sửa: 134
```

kiểm tra và phát hiện các trường hợp đặt phòng nhưng không có đêm lưu trú nào (tổng số đêm ở bằng 0)

```
Số lượng đặt phòng có 0 đêm lưu trú: 0
```

Trong bộ dữ liệu này không có giá trị nào bất hợp lý của tổng số đêm lưu trú

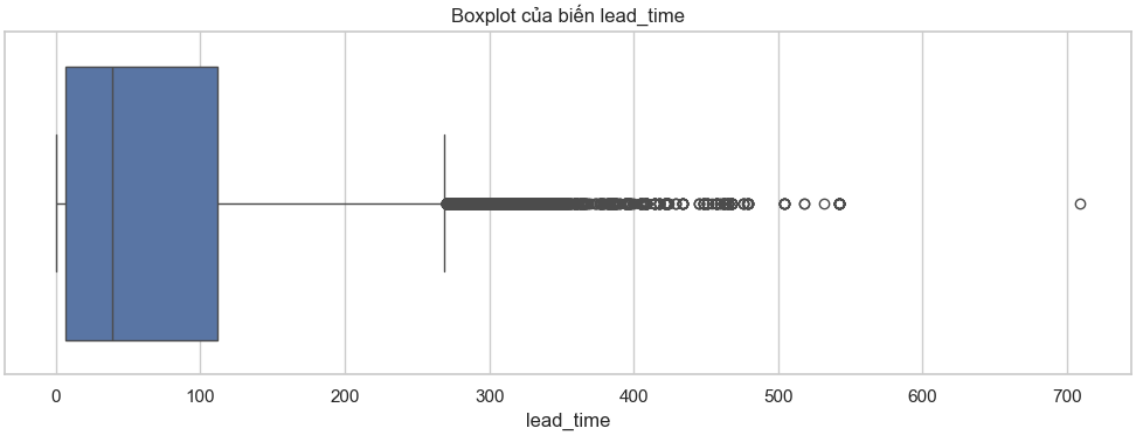
Sau khi thực hiện xử lý các lỗi logic có trong dữ liệu thì bộ dữ liệu của chúng ta còn:

```
(61770, 31)
```

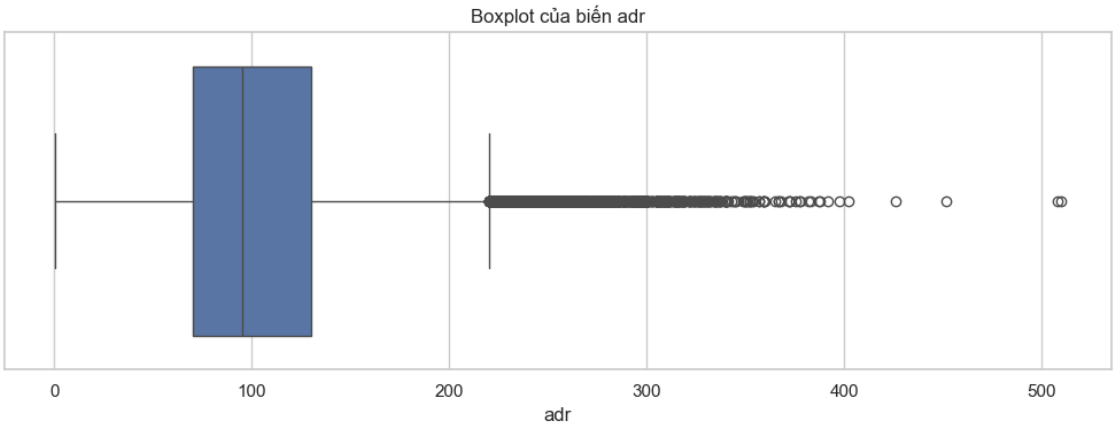
2.3. Xử lý ngoại lệ (Outlier)

2.3.1. Phân tích và phát hiện ngoại lệ

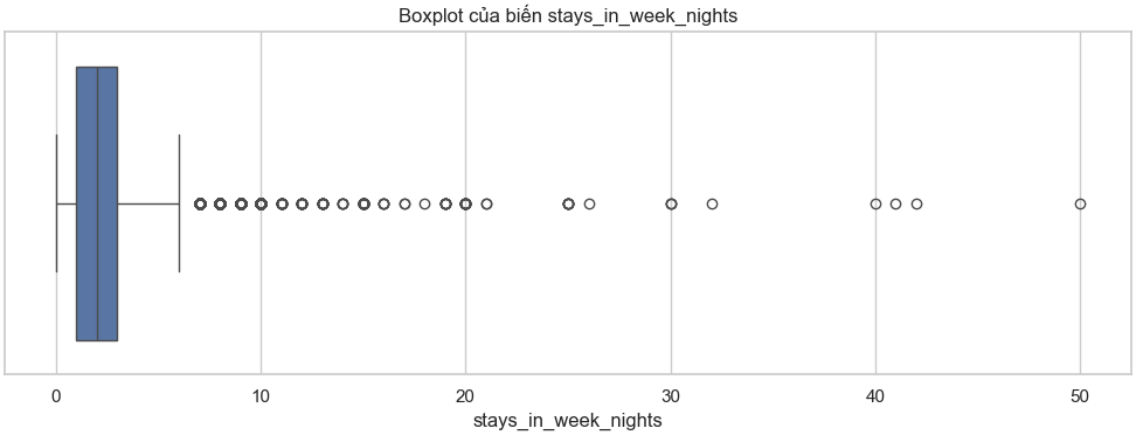
Em đã sử dụng các thống kê mô tả để nắm bắt đặc điểm tổng quan của dữ liệu, đồng thời vẽ biểu đồ boxplot cho các biến số quan trọng như lead_time, adr, stays_in_week_nights và stays_in_weekend_nights nhằm phát hiện các giá trị ngoại lệ. Việc này giúp em xác định các trường hợp bất thường, từ đó có phương án xử lý phù hợp để đảm bảo chất lượng dữ liệu cho các phân tích tiếp theo



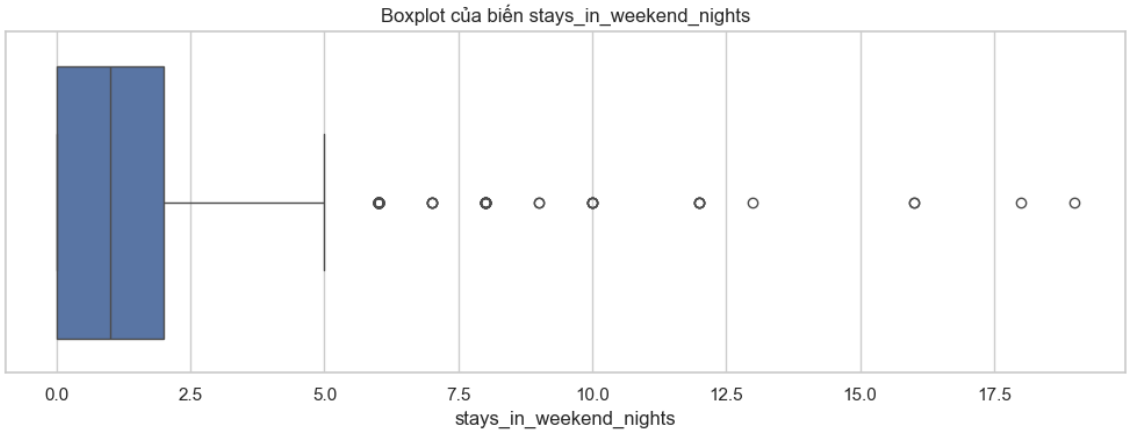
Hình 2. 2 Boxplot của biến lead_time



Hình 2. 3 Boxplot của biến adr



Hình 2. 4 Boxplot của biến stays_in_week_nights



Hình 2. 5 Boxplot của biến stays_in_weekend_nights

Tiếp theo là thực hiện thống kê chi tiết cho các biến số. Việc này giúp em nắm bắt được các đặc điểm cơ bản của dữ liệu, phát hiện các giá trị bất thường

```
Thông tin chi tiết:

adr:
Giá trị trung bình: 104.64
Giá trị tối đa: 510.0
Độ lệch chuẩn: 49.33

lead_time:
Giá trị trung bình: 70.97
Giá trị tối đa: 709
Giá trị trung vị: 39.0

stays_in_week_nights và stays_in_weekend_nights:
Giá trị tối đa stays_in_week_nights: 50
Giá trị trung vị stays_in_week_nights: 2.0
Giá trị tối đa stays_in_weekend_nights: 19
Giá trị trung vị stays_in_weekend_nights: 1.0
```

Đối với biến ADR, dữ liệu cho thấy sự chênh lệch đáng kể giữa giá trị trung bình và giá trị tối đa. Cụ thể, trong khi giá trị trung bình chỉ ở mức 104.64, giá trị tối đa lên đến 510.0, tạo ra một khoảng cách rất lớn. Độ lệch chuẩn cao (49.33) cũng cho thấy dữ liệu có độ phân tán rộng, điều này thường là dấu hiệu của sự hiện diện của các outlier.

Tương tự, biến lead_time cũng thể hiện những đặc điểm bất thường trong phân phối dữ liệu. Giá trị trung bình là 70.97 ngày, nhưng giá trị tối đa lại lên đến 709 ngày, có nghĩa là có những booking được đặt trước gần hai năm. Điều này không phản ánh hành vi đặt phòng thông thường của khách hàng và rất có thể là kết quả của lỗi dữ liệu hoặc những trường hợp booking đặc biệt không đại diện cho tổng thể.

Ngược lại, các biến về số đêm lưu trú như stays_in_week_nights và stays_in_weekend_nights có phân phối giá trị hợp lý, không xuất hiện các giá trị bất thường vượt quá giới hạn thực tế (số đêm tối đa vẫn nằm trong phạm vi chấp nhận được). Do đó, không cần thiết phải xử lý outlier đối với các biến này.

2.3.2. Phương pháp xử lý: Winsorization, Log-transform

Hiển thị thống kê mô tả chi tiết

```
          lead_time           adr
count  61770.000000  61770.000000
mean      70.966003    104.636355
std       81.551622     49.333880
min        0.000000      0.260000
1%         0.000000     27.900000
5%         0.000000     40.000000
25%        7.000000     70.000000
50%       39.000000     95.200000
75%      112.000000    130.000000
95%      237.000000    199.000000
99%      336.000000    259.000000
max      709.000000    510.000000

Phân tích outlier cho biến: lead_time
- Ngưỡng 1.5 × IQR: 1971 outlier (3.19%)
- Ngưỡng 2.0 × IQR: 824 outlier (1.33%)
- Ngưỡng 3.0 × IQR: 107 outlier (0.17%)

Phân tích outlier cho biến: adr
- Ngưỡng 1.5 × IQR: 1812 outlier (2.93%)
- Ngưỡng 2.0 × IQR: 766 outlier (1.24%)
- Ngưỡng 3.0 × IQR: 133 outlier (0.22%)
```

Dựa trên bảng thống kê mô tả và kết quả phân tích outlier, có thể thấy rằng cả hai biến lead_time và adr đều tồn tại các giá trị ngoại lệ đáng kể.

Cụ thể, biến lead_time có giá trị trung bình là 70.97 ngày, nhưng giá trị tối đa lên tới 709 ngày, cao gấp nhiều lần so với trung vị (39 ngày) và giá trị 99% (336 ngày). Phân tích outlier cho thấy với ngưỡng 1.5 × IQR, có tới 1971 giá trị ngoại lệ (chiếm 3.19% tổng số mẫu). Ngay cả khi sử dụng ngưỡng nghiêm ngặt hơn (3.0 × IQR), vẫn còn 107 giá trị ngoại lệ (0.17%). Điều này cho thấy phân phối của biến này bị lệch phải mạnh, một số trường hợp đặt phòng trước quá xa so với phần lớn khách hàng. Nếu không xử lý, các giá trị này có thể làm sai lệch các phân tích về hành vi đặt phòng và ảnh hưởng đến hiệu quả của các mô hình dự báo.

Tương tự, biến adr có giá trị trung bình là 104.64 nhưng giá trị tối đa lên tới 510.0, trong khi giá trị 99% chỉ là 259.0. Số lượng outlier ở ngưỡng 1.5 × IQR là 1812 (2.93%), và ở ngưỡng 3.0 × IQR vẫn còn 133 giá trị (0.22%). Điều này cho thấy tồn tại một số trường hợp có giá phòng bất thường, có thể do lỗi nhập liệu hoặc các tình huống đặc biệt không đại diện cho xu hướng chung. Việc giữ lại các giá trị này sẽ làm sai lệch các chỉ số tổng hợp và ảnh hưởng đến các phân tích tài chính.

Vì vậy, việc phát hiện và xử lý outlier ở hai biến này là cần thiết để đảm bảo kết quả phân tích chính xác, phản ánh đúng thực tế và nâng cao hiệu quả của các mô hình học máy.

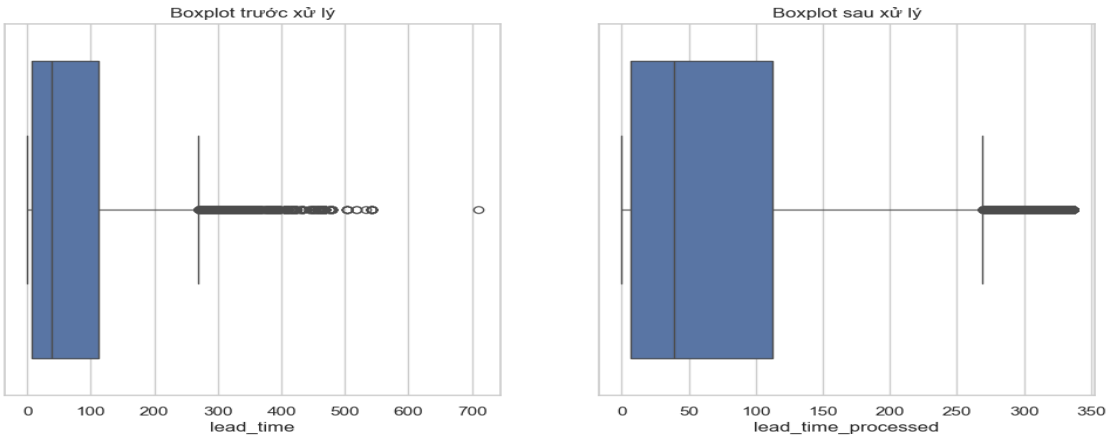
Biến lead_time (thời gian đặt trước):

Phương pháp phù hợp: Cắt ngưỡng (capping/winsorization)

Lý do: Trong quá trình tiền xử lý dữ liệu, biến lead_time có thể chứa các giá trị ngoại lệ lớn, gây ảnh hưởng đến quá trình huấn luyện mô hình. Để khắc phục, em áp dụng kỹ thuật Winsorization tại ngưỡng phân vị thứ 99% để giới hạn các giá trị quá lớn

	lead_time	lead_time_processed
count	61770.000000	61770.000000
mean	70.966003	70.451821
std	81.551622	79.539929
min	0.000000	0.000000
25%	7.000000	7.000000
50%	39.000000	39.000000
75%	112.000000	112.000000
max	709.000000	336.000000

Boxplot sau xử lý



Hình 2. 6 Boxplot của biến lead_time sau xử lý

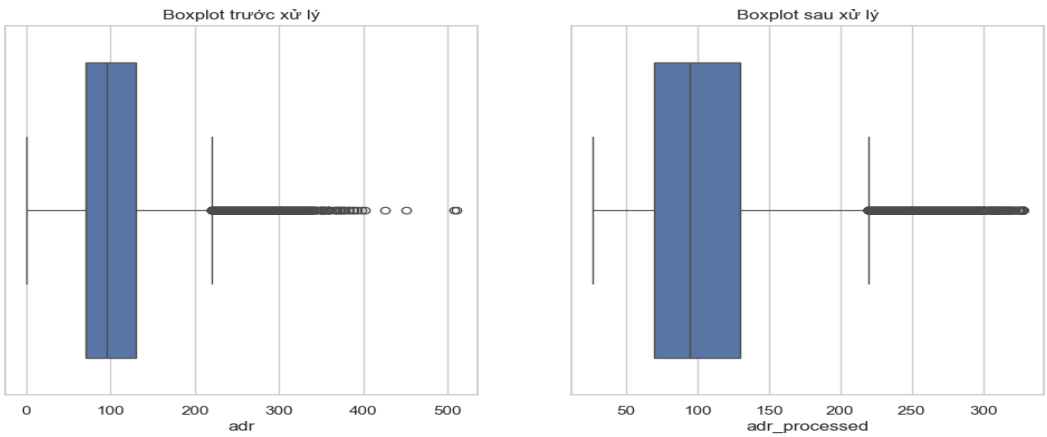
Biến adr:

Phương pháp phù hợp: Biến đổi logarit + cắt ngưỡng

Lý do: Áp dụng biến đổi log cho adr để giảm độ lệch, sau đó xử lý ngoại lệ bằng IQR và chuyển ngược về thang đo ban đầu.

	adr	adr_processed
count	61770.000000	61770.000000
mean	104.636355	104.680163
std	49.333880	49.003770
min	0.260000	27.329501
25%	70.000000	70.000000
50%	95.200000	95.200000
75%	130.000000	130.000000
max	510.000000	327.314993

Boxplot sau xử lý



Hình 2. 7 Boxplot của biến adr sau xử lý

2.4. Kỹ thuật xây dựng đặc trưng (Feature Engineering)

Feature Engineering là quá trình tạo ra, biến đổi hoặc lựa chọn các đặc trưng (biến đầu vào) phù hợp từ dữ liệu gốc nhằm nâng cao hiệu quả và độ chính xác của mô hình học máy.

2.4.1. Tổng số đêm lưu trú

Việc tạo thêm đặc trưng total_nights (tổng số đêm lưu trú) giúp tổng hợp thông tin từ hai biến riêng lẻ là số đêm ở ngày thường và cuối tuần, từ đó cung cấp một cái nhìn tổng quát hơn về thời gian lưu trú của khách. Đặc trưng này hỗ trợ các phân tích về hành vi khách hàng, dự báo doanh thu và giúp mô hình học máy dễ dàng khai thác mối quan hệ giữa tổng thời gian lưu trú với các biến mục tiêu.

Code:

```
# Tạo cột mới là tổng số đêm lưu trú
df['total_nights'] = df['stays_in_weekend_nights'] +
df['stays_in_week_nights']


# Kiểm tra 5 dòng đầu tiên để xác nhận
print(df[['stays_in_weekend_nights', 'stays_in_week_nights',
'total_nights']].head())
```

Output:

	stays_in_weekend_nights	stays_in_week_nights	total_nights
2	0	1	1
3	0	1	1
4	0	2	2
6	0	2	2
7	0	2	2

2.4.2. Mùa du lịch (Season)

Việc thêm đặc trưng season (mùa) từ tháng đến giúp mô hình nhận diện các xu hướng theo mùa vụ, hỗ trợ phân tích sự biến động về lượng khách, giá phòng hoặc hành vi đặt phòng theo từng mùa trong năm, từ đó nâng cao khả năng dự báo và tối ưu hóa hoạt động kinh doanh khách sạn.

Code:

```
# Tạo hàm xác định mùa
def get_season(month):
    if month in [12, 1, 2]:
        return 'Winter'
    elif month in [3, 4, 5]:
        return 'Spring'
    elif month in [6, 7, 8]:
        return 'Summer'
    elif month in [9, 10, 11]:
        return 'Autumn'

# Tạo cột mới 'season' dựa trên 'arrival_date_month'
df['season'] = df['arrival_date_month'].apply(get_season)

print(df[['arrival_date_month', 'season']])
```

Output:

```
        arrival_date_month  season
2                        7  Summer
3                        7  Summer
4                        7  Summer
6                        7  Summer
7                        7  Summer
...                    ...     ...
119385                   8  Summer
119386                   8  Summer
119387                   8  Summer
119388                   8  Summer
119389                   8  Summer

[61770 rows x 2 columns]
```

2.4.3. Tổng số khách (total_guests)

Việc tạo đặc trưng total_guests (tổng số khách) giúp tổng hợp toàn bộ số người trong mỗi đặt phòng, từ đó hỗ trợ phân tích nhu cầu lưu trú, tối ưu hóa phân bổ phòng và nâng cao hiệu quả dự báo các chỉ số kinh doanh liên quan đến số lượng khách

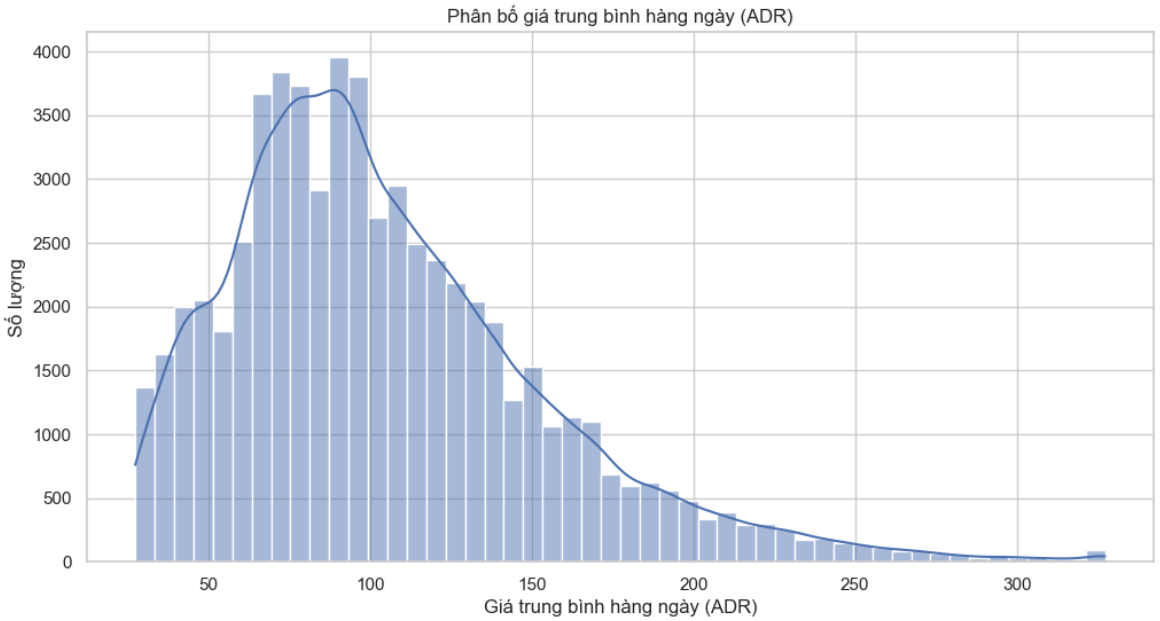
Code:

```
#Chúng ta sẽ tạo biến tổng số người bằng cách cộng các giá trị biểu thị là người
df['total_guests'] = df['adults'] + df['children'] + df['babies']
```

CHƯƠNG 3: PHÂN TÍCH VÀ TRỰC QUAN HOÁ DỮ LIỆU

Phân tích và trực quan hóa dữ liệu giúp khám phá, hiểu rõ cấu trúc và xu hướng của dữ liệu thông qua các thống kê và biểu đồ trực quan, từ đó hỗ trợ phát hiện vấn đề, đưa ra quyết định và định hướng các bước phân tích hoặc xây dựng mô hình tiếp theo

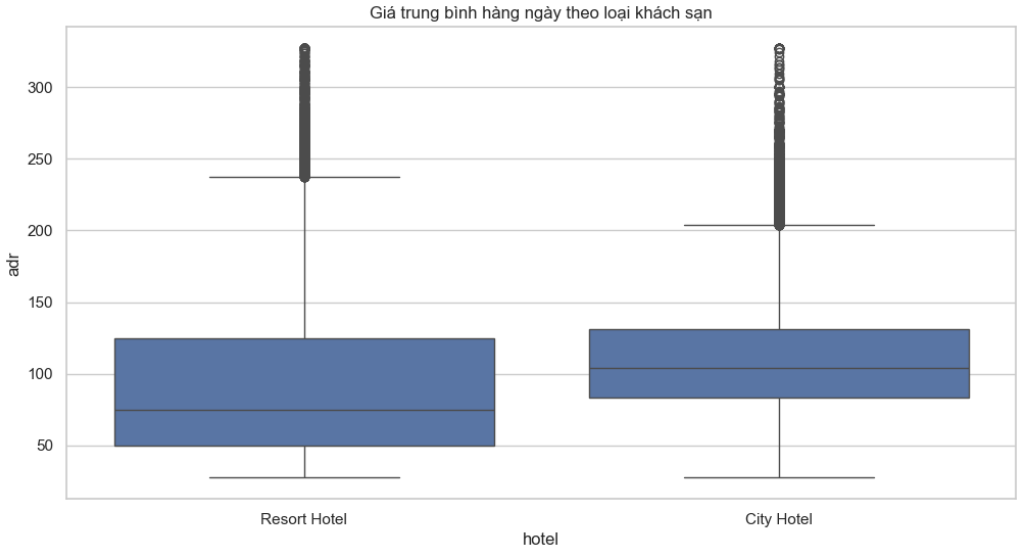
3.1. Phân tích phân bố Giá trung bình hàng ngày (ADR)



Hình 3. 1 Phân tích phân bố Giá trung bình hàng ngày

Biểu đồ thể hiện phân bố giá trung bình hàng ngày (ADR) có dạng chuông nghiêng phải, với phần lớn dữ liệu tập trung ở mức giá 80-100 đơn vị (tần suất cao nhất khoảng 3,800-4,000 quan sát). Giá trị dao động từ khoảng 20 đến trên 300 đơn vị, nhưng tần suất giảm dần khi giá tăng cao, cho thấy có rất ít quan sát ở mức giá trên 200 đơn vị. Phân bố này phản ánh cấu trúc thị trường điển hình của ngành dịch vụ lưu trú, nơi đa số các đơn vị có giá ở phân khúc trung bình và chỉ một số ít thuộc phân khúc cao cấp với mức giá rất cao.

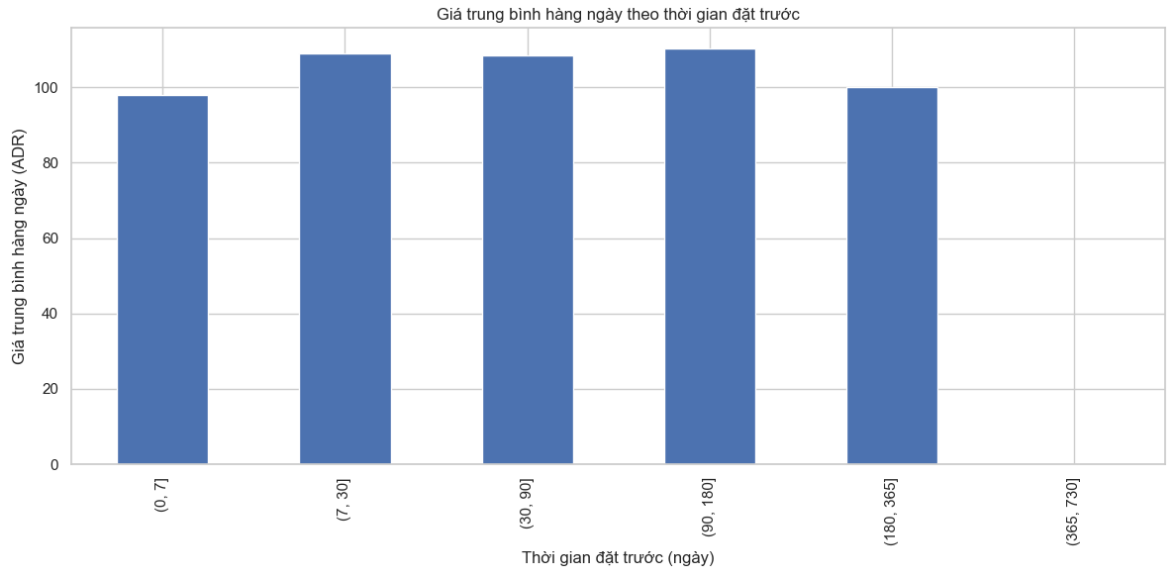
3.2. Giá trung bình hàng ngày theo loại khách sạn



Hình 3. 2 Giá trung bình hàng ngày theo loại khách sạn

Biểu đồ cho thấy giá trị ADR của City Hotel thường cao hơn Resort Hotel, với trung vị khoảng 100 so với khoảng 70. Tuy nhiên, mức giá tại Resort Hotel biến động lớn hơn, thể hiện qua hộp IQR rộng và nhiều giá trị ngoại lệ cao. Ngược lại, City Hotel có mức giá ổn định hơn, với phân bố hẹp hơn và ít dao động. Điều này cho thấy City Hotel thường có giá cao và ổn định hơn, trong khi Resort Hotel có xu hướng giá thấp hơn nhưng biến động mạnh hơn.

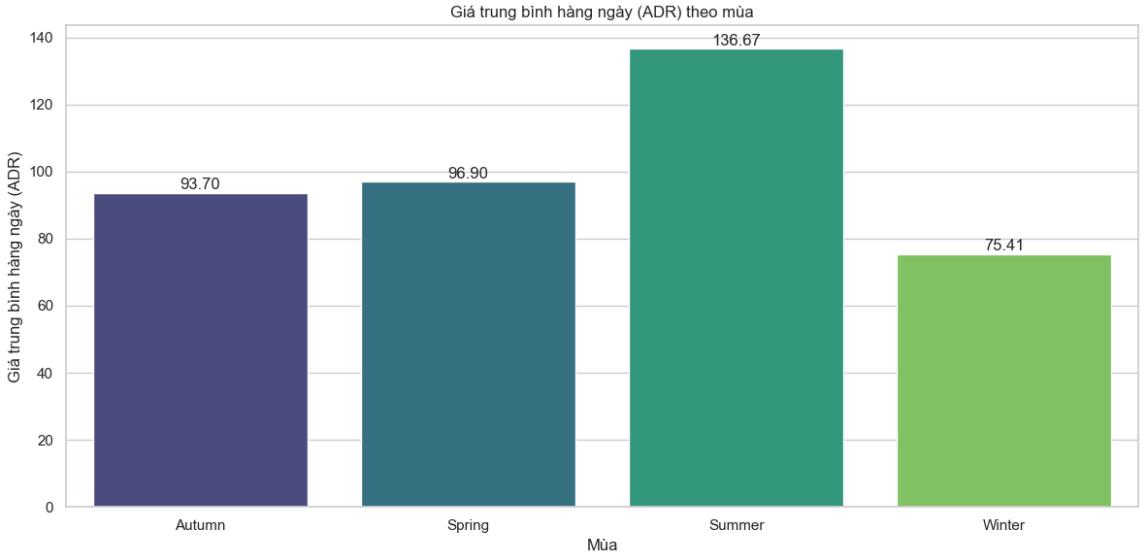
3.3. Giá trung bình hàng ngày theo thời gian đặt trước



Hình 3. 3 Giá trung bình hàng ngày theo thời gian đặt trước

Biểu đồ cho thấy giá trung bình hàng ngày (ADR) có xu hướng cao hơn khi khách đặt phòng trước từ 1 đến 6 tháng, đặc biệt cao nhất trong khoảng 90 - 180 ngày. Trong khi đó, đặt sát ngày (dưới 1 tuần) hoặc đặt quá sớm (trên 6 tháng) thường có mức giá thấp hơn. Điều này cho thấy có thể tồn tại chiến lược giá tối ưu trong khoảng thời gian đặt phòng từ 1 đến 6 tháng trước ngày lưu trú.

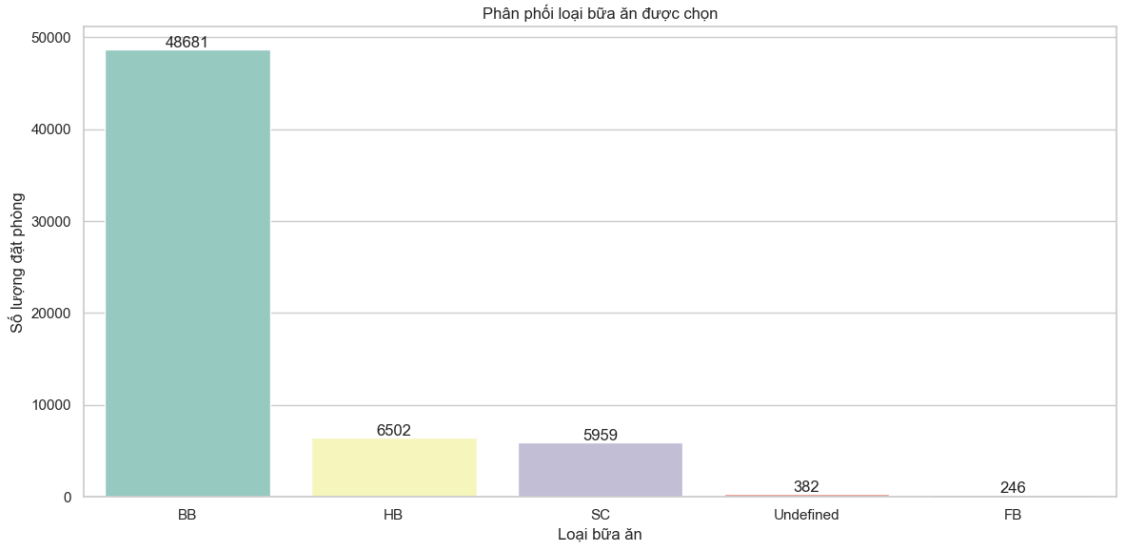
3.4. Phân tích Giá trung bình hàng ngày trung bình theo mùa



Hình 3. 4 Phân tích Giá trung bình hàng ngày trung bình theo mùa

Biểu đồ cho thấy giá trung bình hàng ngày (ADR) thay đổi theo mùa trong năm, với mức giá cao nhất vào mùa hè (136.67), tiếp theo là mùa xuân (96.90) và mùa thu (93.70), trong khi mùa đông có mức giá thấp nhất (75.41). Điều này phản ánh xu hướng giá phòng tăng mạnh vào mùa du lịch cao điểm (mùa hè), trong khi giảm vào mùa thấp điểm như mùa đông.

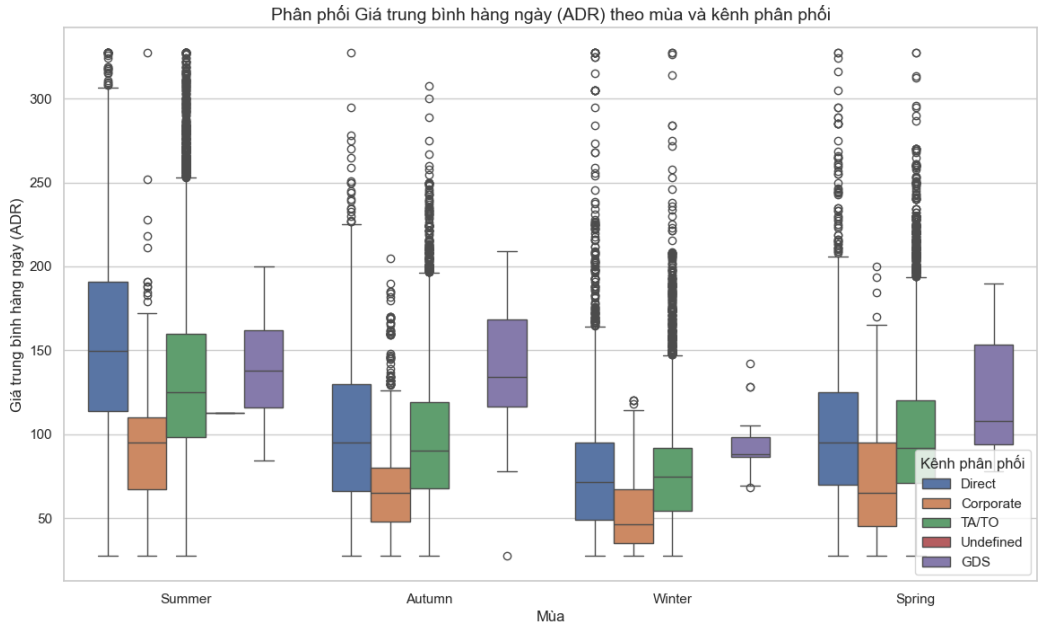
3.5. Phân tích loại bữa ăn được chọn



Hình 3. 5 Phân tích loại bữa ăn được chọn

Biểu đồ thể hiện phân phối số lượng đặt phòng theo loại bữa ăn cho thấy loại "BB" (chỉ bao gồm bữa sáng) là lựa chọn phổ biến nhất với 48,681 lượt đặt, vượt xa so với các loại còn lại. Các loại "HB" (bữa sáng và tối) và "SC" (không bao gồm bữa ăn) có số lượng đặt tương đối thấp hơn, lần lượt là 6,502 và 5,959 lượt. Trong khi đó, các loại "Undefined" và "FB" (đầy đủ ba bữa) chỉ chiếm tỷ lệ rất nhỏ, với lần lượt 382 và 246 lượt đặt, cho thấy khách hàng có xu hướng ưa chuộng loại hình đơn giản, linh hoạt hơn trong ăn uống.

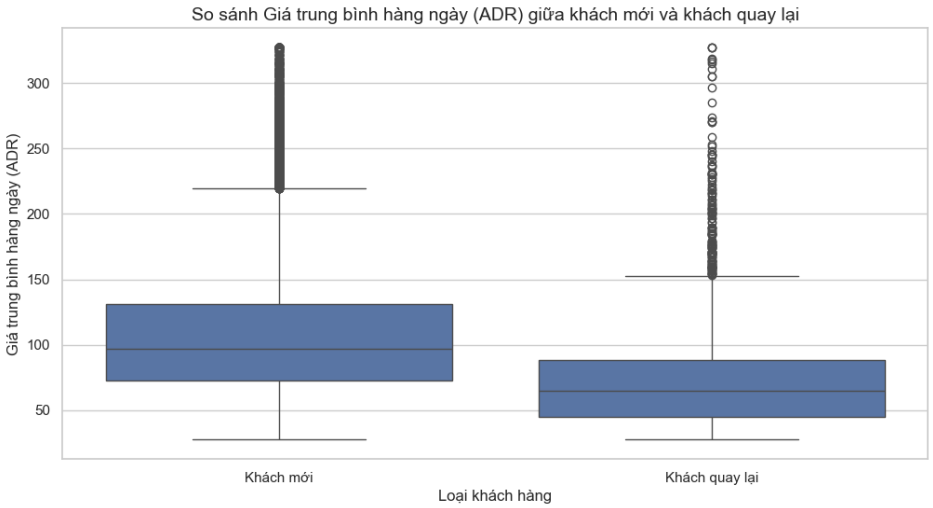
3.6. Phân tích ADR theo mùa và kênh phân phối



Hình 3. 6 Phân tích ADR theo mùa và kênh phân phối

Biểu đồ thể hiện sự phân phối Giá trung bình hàng ngày (ADR) theo mùa và kênh phân phối cho thấy ADR biến động mạnh theo thời gian và kênh bán. Mùa hè có ADR cao nhất, đặc biệt qua các kênh Direct và GDS, trong khi mùa đông ghi nhận mức ADR thấp nhất. Kênh GDS duy trì mức ADR cao và ổn định quanh năm, phản ánh phân khúc khách chi tiêu cao, trong khi kênh Corporate có ADR thấp và ít biến động, đặc trưng cho khách công vụ. Những khác biệt này là cơ sở quan trọng để mô hình máy học dự đoán ADR chính xác hơn, từ đó hỗ trợ tối ưu hóa doanh thu khách sạn.

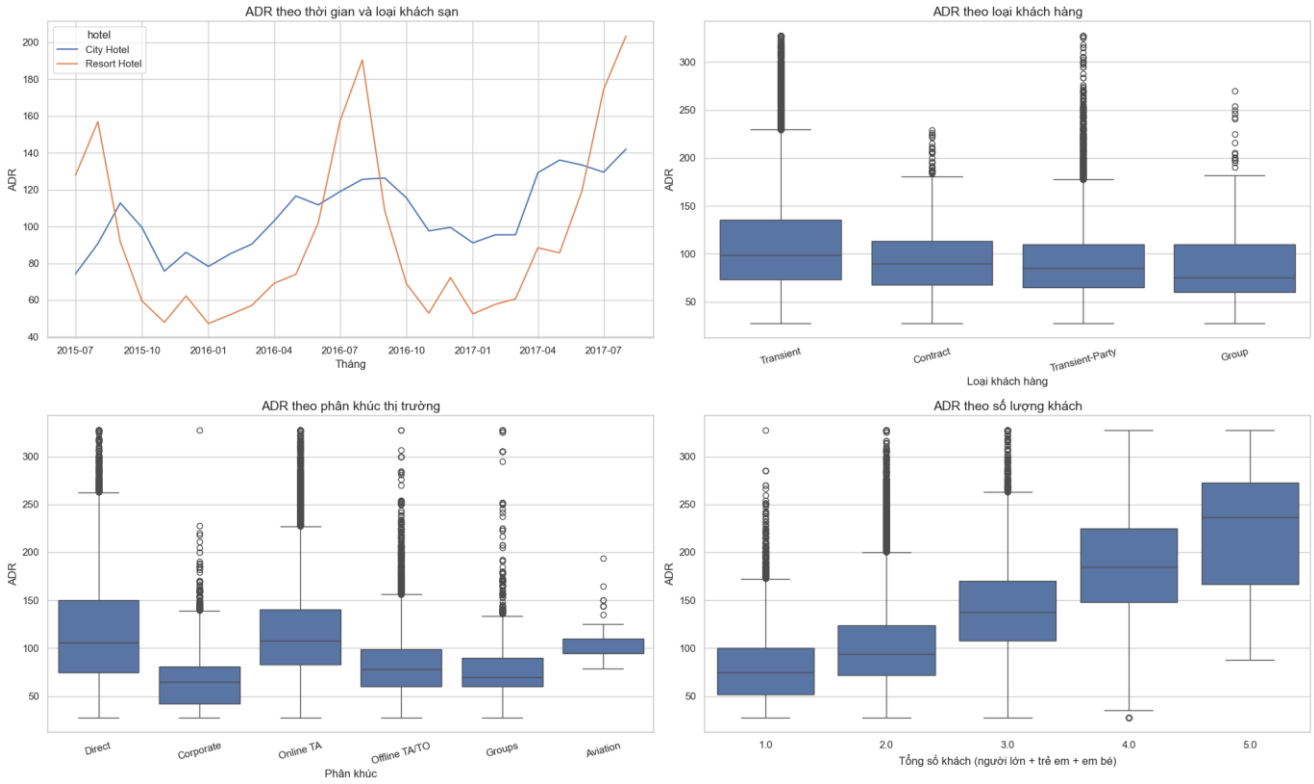
3.7. Phân tích khách hàng quay lại



Hình 3. 7 Phân tích khách hàng quay lại

Biểu đồ so sánh Giá trung bình hàng ngày (ADR) giữa khách mới và khách quay lại cho thấy khách mới thường chi trả mức giá cao hơn so với khách quay lại. Cụ thể, khách mới có giá trị trung vị ADR cao hơn và phân phối giá cũng rộng hơn, với nhiều ngoại lệ ở mức cao. Trong khi đó, khách quay lại có ADR thấp hơn và ít biến động hơn. Điều này gợi ý rằng khách sạn có thể đang áp dụng chính sách giá ưu đãi cho khách quay lại để khuyến khích lòng trung thành, và yếu tố này cần được xem xét khi xây dựng mô hình máy học dự đoán ADR.

3.8. Dashboard tổng quan về xu hướng theo thời gian



Hình 3. 8 Dashboard tổng quan về xu hướng theo thời gian

ADR theo thời gian và loại khách sạn (Biểu đồ trên cùng bên trái):

Biểu đồ đường cho thấy khách sạn Resort (Resort Hotel) có tính mùa vụ cao hơn khách sạn thành phố (City Hotel). ADR của Resort thường đạt đỉnh vào mùa hè (tháng 7-8) và giảm mạnh vào mùa đông. Trong khi đó, City Hotel có mức ADR ổn định hơn quanh năm. Điều này phản ánh hành vi tiêu dùng du lịch nghỉ dưỡng tăng mạnh theo mùa.

ADR theo loại khách hàng (Biểu đồ trên cùng bên phải):

Trong số các loại khách hàng, nhóm Transient (khách vãng lai) và Transient-Party có ADR trung bình cao hơn rõ rệt so với nhóm Contract (khách đặt theo hợp đồng) và Group (khách đoàn). Điều này cho thấy khách cá nhân thường chi trả nhiều hơn, còn khách đoàn hoặc theo hợp đồng có mức giá ưu đãi hơn.

ADR theo phân khúc thị trường (Biểu đồ dưới cùng bên trái):

Phân khúc Direct và Online TA (Travel Agents) có ADR cao hơn rõ rệt so với các kênh như Corporate hay Groups, thể hiện rằng khách đặt trực tiếp và qua đại lý trực tuyến thường chi trả cao hơn. Ngược lại, phân khúc Corporate có mức ADR thấp nhất, phản ánh các thỏa thuận giá thấp theo hợp đồng dài hạn. Aviation (hàng không) có mức ADR trung bình khá cao, có thể liên quan đến khách hàng có nhu cầu gấp hoặc đi công tác.

ADR theo số lượng khách (Biểu đồ dưới cùng bên phải):

Có mối tương quan thuận giữa số lượng khách và ADR: càng nhiều khách trong một đặt phòng thì ADR càng cao. Điều này có thể do các phòng lớn hơn hoặc dịch vụ bổ sung làm tăng chi phí.

CHƯƠNG 4: XÂY DỰNG VÀ PHÁT TRIỂN MÔ HÌNH DỰ ĐOÁN

4.1 Chuẩn bị dữ liệu cho mô hình học máy

4.1.1. Xử lý và Mã hóa Dữ liệu

chúng ta sẽ thực hiện encoding các biến phân loại để chuẩn bị tốt dữ liệu cho việc huấn luyện mô hình

Việc tách riêng các biến số (numeric) và biến phân loại (categorical) giúp lựa chọn phương pháp xử lý, phân tích và xây dựng mô hình phù hợp cho từng loại dữ liệu, đồng thời hỗ trợ quá trình tiền xử lý như chuẩn hóa, mã hóa
hoặc chọn đặc trưng hiệu quả hơn.

Dữ liệu số (Numeric Data)	**Dữ liệu phân loại (Categorical Data)**
is_canceled	hotel
lead_time	meal
arrival_date_year	country
arrival_date_month	market_segment
arrival_date_week_number	distribution_channel
arrival_date_day_of_month	reserved_room_type
stays_in_weekend_nights	assigned_room_type
stays_in_week_nights	deposit_type
adults	customer_type
children	reservation_status
babies	reservation_status_date
is_repeated_guest	season
previous_cancellations	

Dữ liệu số (Numeric Data)	Dữ liệu phân loại (Categorical Data)
previous_bookings_not_canceled	
booking_changes	
agent	
days_in_waiting_list	
adr	
required_car_parking_spaces	
total_of_special_requests	
total_nights	
total_guests	

Bây giờ sẽ liệt kê các giá trị duy nhất của từng biến phân loại quan trọng, giúp kiểm tra, hiểu rõ phạm vi dữ liệu và phát hiện các trường hợp bất thường hoặc giá trị hiếm, từ đó hỗ trợ quá trình phân tích và mã hóa dữ liệu hiệu quả hơn.

- **hotel:** Gồm hai loại khách sạn là Resort Hotel và City Hotel.
- **meal:** Loại suất ăn bao gồm BB (bed & breakfast), HB (half board), FB (full board), SC (self-catering), và Undefined.
- **country:** Quốc gia nơi khách hàng cư trú. Dữ liệu gồm hơn 100 quốc gia, trong đó các giá trị phổ biến là PRT (Bồ Đào Nha), GBR (Anh), ESP (Tây Ban Nha), USA (Hoa Kỳ), v.v.
- **market_segment:** Thể hiện phân khúc thị trường nơi đặt phòng như Online TA, Offline TA/TO, Groups, Corporate, Direct, Complementary, và Aviation.
- **distribution_channel:** Kênh phân phối đặt phòng như TA/TO (travel agent/tour operator), Corporate, Direct, GDS, hoặc Undefined.
- **reserved_room_type và assigned_room_type**: Loại phòng được khách đặt và loại phòng thực tế được giao, được mã hóa bằng các chữ cái (ví dụ: A, B, C…).

- **deposit_type**: Hình thức đặt cọc, gồm No Deposit, Refundable, và Non Refund.
- **customer_type:** Phân loại khách hàng như Transient, Contract, Transient-Party, và Group.
- **season:** Mùa đặt phòng, gồm Spring, Summer, Autumn, và Winter.

Phương pháp Label Encoding

Trong bộ dữ liệu ban đầu, biến hotel là một biến phân loại (categorical variable) với hai giá trị dạng chuỗi: "Resort Hotel" và "City Hotel". Tuy nhiên, các thuật toán máy học không thể xử lý trực tiếp dữ liệu dạng chuỗi, do đó cần chuyển đổi chúng sang dạng số học.

Để thực hiện việc này, em đã áp dụng Label Encoding bằng cách ánh xạ mỗi loại khách sạn sang một giá trị số nguyên như sau:

Code:

```
# Label Encoding
df_encoded['hotel'] = df_encoded['hotel'].map({'Resort
Hotel': 0, 'City Hotel': 1})
```

Tương tụ, tất cả các biến phân loại như **meal, market_segment, distribution_channel, deposit_type** và **customer_type** đều được xử lý bằng phương pháp Label Encoding, trong đó mỗi giá trị dạng chuỗi được ánh xạ sang một số nguyên cụ thể. Phương pháp này đơn giản và hiệu quả khi số lượng giá trị không quá lớn. Tuy nhiên, do Label Encoding có thể gây hiểu nhầm thứ tự giữa các nhãn, em đã cân nhắc kỹ lưỡng để đảm bảo rằng mô hình sử dụng không bị ảnh hưởng tiêu cực từ giả định này.

Hai biến reserved_room_type và assigned_room_type chứa các giá trị chữ cái (A, B, C,...) đại diện cho các loại phòng. Những giá trị này không có thứ tự tự nhiên (nominal categorical), và số lượng giá trị có thể lên đến 10 loại khác nhau. Do đó, em đã sử dụng LabelEncoder từ thư viện sklearn.preprocessing để mã hóa tự động các giá trị này thành số nguyên:

Code:

```
# from sklearn.preprocessing import LabelEncoder


le_reserved = LabelEncoder()
le_assigned = LabelEncoder()
```

```
df_encoded['reserved_room_type'] =
le_reserved.fit_transform(df_encoded['reserved_room_type'])
df_encoded['assigned_room_type'] =
le_assigned.fit_transform(df_encoded['assigned_room_type'])
pd.concat([df_encoded['reserved_room_type'],
df_encoded['assigned_room_type']]).unique()
```

Phương pháp này sẽ gán mỗi giá trị duy nhất trong cột một số nguyên từ 0 đến n-1, trong đó n là số lượng nhãn khác nhau

Mã hóa thứ bậc (Ordinal Encoding)

Đối với biến phân loại season (mùa), các giá trị "Spring", "Summer", "Autumn" và "Winter" mang tính chất có thứ tự tự nhiên theo thời gian trong năm. Do đó, em đã áp dụng Ordinal Encoding – phương pháp ánh xạ các giá trị này sang các số nguyên tương ứng với thứ tự của chúng:

Code:

```
# Ordinal Encoding
season_mapping = {'Spring': 0, 'Summer': 1, 'Autumn': 2,
'Winter': 3}
df_encoded['season'] =
df_encoded['season'].map(season_mapping)
```

Tiếp đến, em sẽ xoá cột `is_canceled`, `reservation_status`. Việc xóa cột giúp giảm nhiễu và tối ưu hóa dữ liệu cho bước huấn luyện tiếp theo vì chúng ta đã loại bỏ những giá trị in_canceled = 0

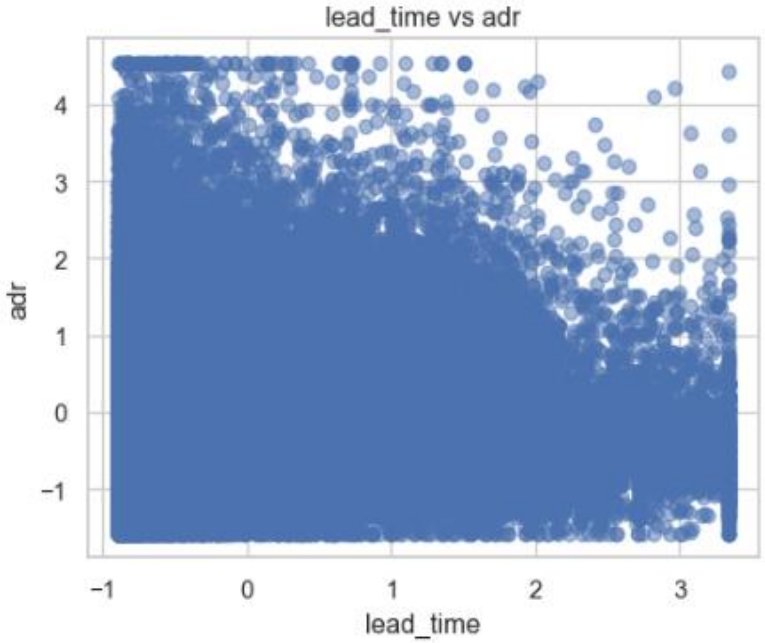
```
Đã xóa cột is_canceled và reservation_status.
```

4.1.2. Phân tích thêm các mối tương quan phi tuyến

Tạo lưới subplot và vẽ biểu đồ scatter cho từng cột so với adr

Bây giờ em chọn ra 3 biểu đồ sau có mối quan hệ trực quan, đáng chú ý và có thể gợi mở insight cho bài toán dự đoán adr.

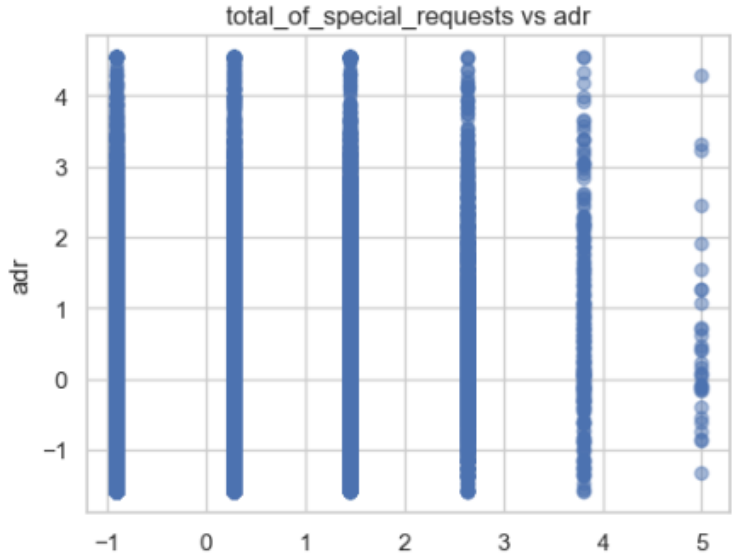
lead_time vs adr



Hình 4. 1 Tương quan lead_time vs adr

Có xu hướng giảm nhẹ của adr khi lead_time tăng, tức là các đơn đặt sớm hơn thường có mức giá thấp hơn. Điều này phản ánh chính sách giá ưu đãi cho khách hàng đặt trước từ sớm hoặc có thể do nhu cầu chưa cao ở thời điểm xa ngày lưu trú.

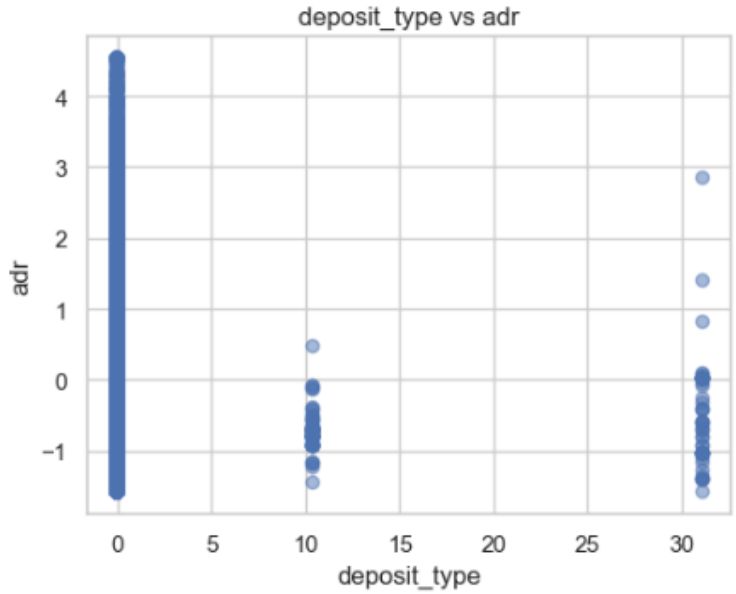
total_of_special_requests vs adr



Hình 4. 2 Tương quan total_of_special_requests vs adr

adr tăng nhẹ theo số lượng yêu cầu đặc biệt. Điều này hợp lý vì những khách hàng đưa ra nhiều yêu cầu đặc biệt thường là khách hàng có ngân sách cao hơn hoặc yêu cầu dịch vụ tốt hơn, từ đó làm tăng tổng chi phí lưu trú.

deposit_type vs adr

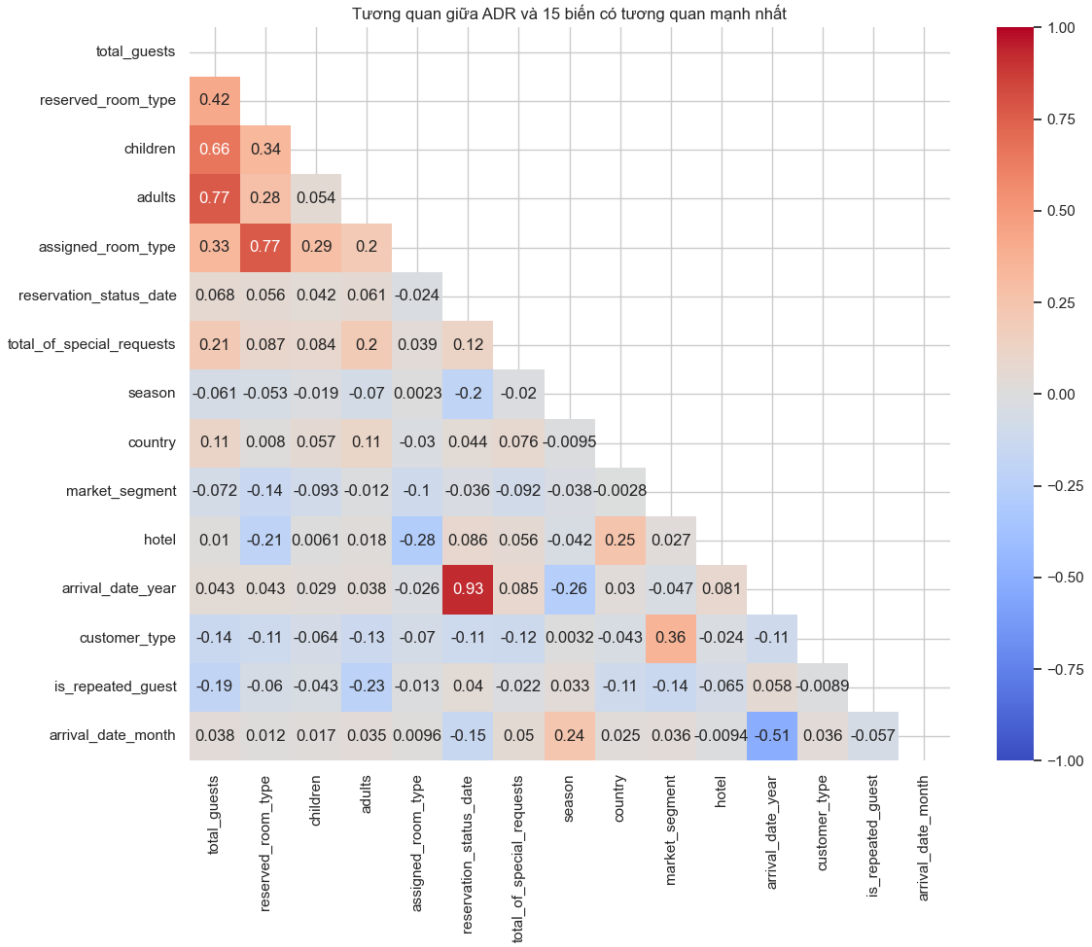


Hình 4. 3 Tương quan deposit_type vs adr

Loại No Deposit có nhiều mức adr cao, trong khi các loại yêu cầu đặt cọc (Non Refund, Refundable) lại chủ yếu tập trung ở mức giá thấp hơn. Có thể vì các khách sạn ưu đãi giá thấp để bù lại tính bắt buộc của đặt cọc.

4.1.3. Phân tích tương quan

Vẽ heatmap để trực quan hóa ma trận tương quan, với các tham số tùy chỉnh để làm nổi bật mối quan hệ giữa các biến.



Hình 4. 4 Tương quan giữa ADR và 15 biến có tương quan mạnh nhất

Xác định và sắp xếp 15 biến có tương quan mạnh nhất (cả dương và âm) với adr, giúp nhận diện các yếu tố ảnh hưởng lớn đến giá phòng

```
15 biến có tương quan mạnh và có ý nghĩa với ADR:
total_guests                 0.455202
reserved_room_type           0.407399
children                     0.339644
adults                       0.329857
assigned_room_type           0.263223
reservation_status_date      0.239341
total_of_special_requests    0.220022
season                      -0.218311
country                      0.193410
market_segment              -0.178430
hotel                        0.159687
arrival_date_year            0.158663
customer_type               -0.131083
is_repeated_guest           -0.130762
arrival_date_month           0.128710
Name: adr, dtype: float64
```

Qua hai biểu đồ trên đã cung cấp cái nhìn trực quan và định lượng về mối tương quan giữa ADR và 15 biến độc lập có độ tương quan cao nhất. Biểu đồ ma trận tam giác đầu tiên biểu diễn hệ số tương quan Pearson giữa các biến, trong đó màu sắc và độ đậm nhạt thể hiện mức độ và chiều hướng tương quan. Dễ dàng nhận thấy rằng total_guests có mối tương quan dương mạnh nhất với ADR (hệ số 0.455), tiếp theo là reserved_room_type (0.407) và children (0.34). Những biến này có thể phản ánh quy mô và mức độ yêu cầu của khách đặt phòng – yếu tố ảnh hưởng trực tiếp đến giá phòng trung bình hàng ngày.

Biểu đồ thứ hai củng cố và sắp xếp các biến theo thứ tự tương quan tuyệt đối với ADR, giúp định lượng rõ hơn mức độ ảnh hưởng. Đáng chú ý, một số biến như season (-0.218), market_segment (-0.178), và customer_type (-0.131) có hệ số tương quan âm, cho thấy các yếu tố theo mùa hoặc phân khúc thị trường cũng ảnh hưởng đến việc điều chỉnh giá phòng theo hướng giảm.

Tổng thể, kết quả phân tích tương quan cung cấp định hướng ban đầu quan trọng cho việc lựa chọn đặc trưng (feature selection) trong mô hình dự đoán ADR, giúp rút gọn tập biến và tập trung vào các yếu tố có ảnh hưởng thực sự, từ đó cải thiện hiệu quả và độ chính xác của mô hình học máy.

4.1.4 Chuẩn hóa dữ liệu và tách tập huấn luyện – kiểm tra

- Tập huấn luyện (train set): Dùng để huấn luyện mô hình.

- Tập kiểm tra (test set): Dùng để kiểm tra hiệu suất của mô hình trên dữ liệu chưa được nhìn thấy.

Code:

```
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split


# Chuẩn hóa dữ liệu số
numeric_features = df_encoded.select_dtypes(include=['int64',
'float64']).columns
scaler = StandardScaler()
df_encoded[numeric_features] =
scaler.fit_transform(df_encoded[numeric_features])
```

```
# Tách tập dữ liệu
X = df_encoded.drop(['adr', 'reservation_status_date'], axis=1)
y = df_encoded['adr']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Dữ liệu đã được chia thành hai phần:

- Tập huấn luyện (train set): 80% tổng số mẫu, dùng để huấn luyện mô hình.
- Tập kiểm tra (test set): 20% tổng số mẫu, dùng để đánh giá hiệu quả dự báo của mô hình trên dữ liệu chưa từng thấy.

4.1.5 Lựa chọn và đánh giá đặc trưng quan trọng

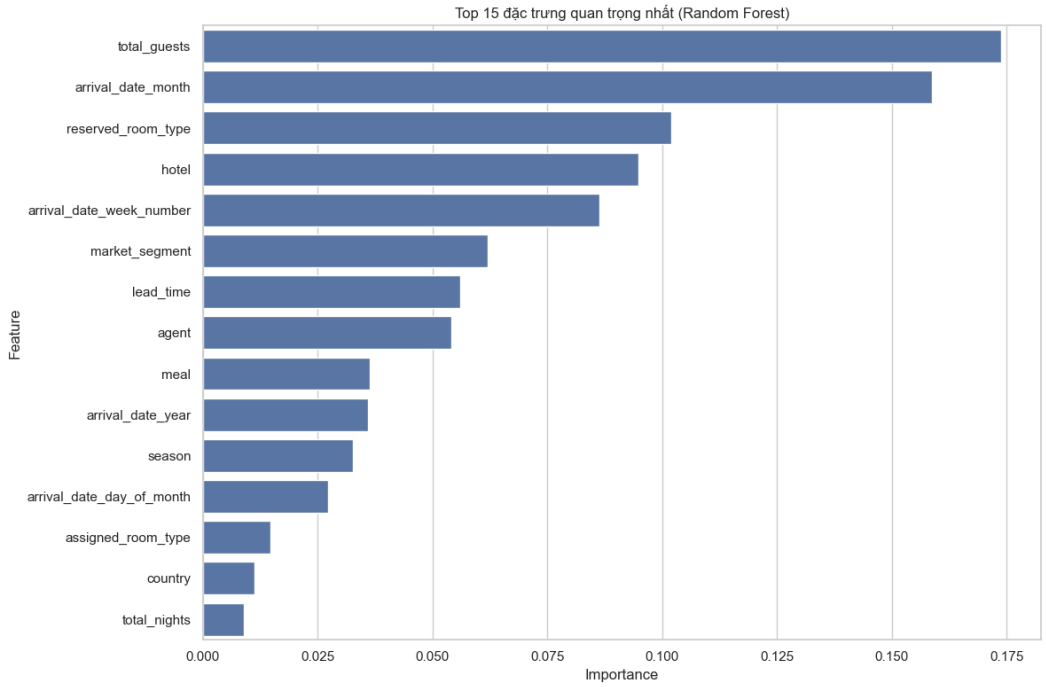
Quy trình lựa chọn và đánh giá đặc trưng quan trọng được thực hiện qua các bước:

Huấn luyện mô hình và tính toán độ quan trọng đặc trưng

Em sẽ dùng mô hình tree-based feature importance cho biết mức độ ảnh hưởng của từng đặc trưng đến kết quả dự báo của mô hình cây quyết định, giúp lựa chọn các biến đầu vào quan trọng, loại bỏ biến dư thừa và nâng cao hiệu quả, độ chính xác của mô hình học máy

Sử dụng Random Forest Regressor feature importance

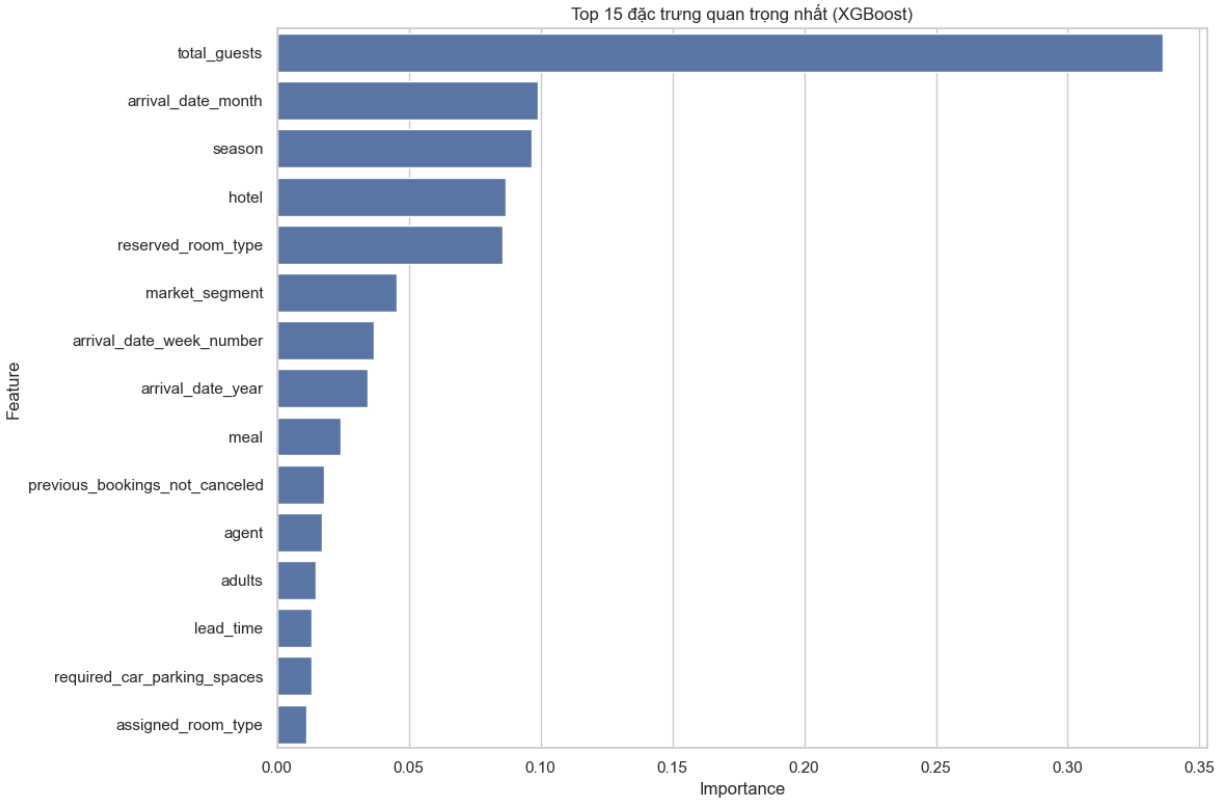
Kết quả



Hình 4. 5 Top 15 đặc trưng quan trong nhất (Random Forest)

Sử dụng XGBoost feature importance

Kết quả:



Hình 4. 6 Top 15 đặc trưng quan trong nhất (XGBoost)

Áp dụng phương pháp loại trừ đệ quy (RFE):

Sử dụng thuật toán RFE (Recursive Feature Elimination) kết hợp với Linear Regression để lựa chọn ra tập đặc trưng tối ưu, loại bỏ dần các biến ít quan trọng nhất cho đến khi còn lại số lượng đặc trưng mong muốn.

	Feature	Ranking	Selected
0	hotel	1	True
1	lead_time	1	True
2	arrival_date_year	1	True
3	arrival_date_month	1	True
4	arrival_date_week_number	1	True
5	arrival_date_day_of_month	1	True
9	children	1	True
8	adults	1	True
13	market_segment	1	True
12	country	1	True
11	meal	1	True
18	reserved_room_type	1	True
26	total_of_special_requests	1	True
28	season	1	True
22	agent	1	True
25	required_car_parking_spaces	2	False
15	is_repeated_guest	3	False
20	booking_changes	4	False
29	total_guests	5	False
19	assigned_room_type	6	False
24	customer_type	7	False
17	previous_bookings_not_canceled	8	False
10	babies	9	False
16	previous_cancellations	10	False
23	days_in_waiting_list	11	False
6	stays_in_weekend_nights	12	False
7	stays_in_week_nights	13	False

Tổng hợp xếp hạng từ nhiều phương pháp và chọn ra 25 đặc trưng để đưa vào huấn luyện

Code:

```
# Tạo bảng tổng hợp xếp hạng từ nhiều phương pháp
feature_rankings = pd.DataFrame({'Feature': X.columns})
feature_rankings['RF Rank'] =
feature_rankings['Feature'].map(feature_importance.set_index(
'Feature')['Importance'].rank(ascending=False))
feature_rankings['XGB Rank'] =
feature_rankings['Feature'].map(xgb_importance.set_index('Fea
ture')['Importance'].rank(ascending=False))
```

```
feature_rankings['RFE Rank'] =
feature_rankings['Feature'].map(feature_ranking.set_index('Fe
ature')['Ranking'])


# Tính điểm trung bình từ các phương pháp
feature_rankings['Average Rank'] = feature_rankings[['RF
Rank', 'XGB Rank', 'RFE Rank']].mean(axis=1)
feature_rankings = feature_rankings.sort_values('Average
Rank')


print("Top 25 đặc trưng quan trọng nhất:")
print(feature_rankings.head(25))
```

Output:

```
Top 25 đặc trưng quan trọng nhất:
                          Feature  RF Rank  XGB Rank  RFE Rank  Average Rank
3               arrival_date_month      2.0       2.0         1      1.666667
29                    total_guests      1.0       1.0         5      2.333333
18              reserved_room_type      3.0       5.0         1      3.000000
0                           hotel      4.0       4.0         1      3.000000
4         arrival_date_week_number      5.0       7.0         1      4.333333
13                 market_segment      6.0       6.0         1      4.333333
28                         season     11.0       3.0         1      5.000000
11                           meal      9.0       9.0         1      6.333333
2               arrival_date_year     10.0       8.0         1      6.333333
22                          agent      8.0      11.0         1      6.666667
1                       lead_time      7.0      13.0         1      7.000000
12                        country     14.0      16.0         1     10.333333
5       arrival_date_day_of_month     12.0      19.0         1     10.666667
8                          adults     19.0      12.0         1     10.666667
19             assigned_room_type     13.0      15.0         6     11.333333
25    required_car_parking_spaces     22.0      14.0         2     12.666667
26      total_of_special_requests     17.0      20.0         1     12.666667
17  previous_bookings_not_canceled     24.0      10.0         8     14.000000
24                  customer_type     23.0      21.0         7     17.000000
14           distribution_channel     20.0      17.0        14     17.000000
20                booking_changes     18.0      30.0         4     17.333333
27                   total_nights     15.0      22.0        15     17.333333
15              is_repeated_guest     26.0      24.0         3     17.666667
10                         babies     27.0      18.0         9     18.000000
9                        children     25.0      29.0         1     18.333333
```

Bảng trên thể hiện 25 đặc trưng quan trọng nhất được lựa chọn dựa trên tổng hợp thứ hạng từ ba phương pháp: Random Forest, XGBoost và RFE. Các đặc trưng này bao gồm cả

thông tin về thời gian, thông tin về khách và đặt phòng, cũng như các yếu tố liên quan đến hành vi khách hàng.

Các đặc trưng về thời gian như tháng, tuần, năm đến có thứ hạng rất cao, cho thấy yếu tố mùa vụ và thời điểm đặt phòng ảnh hưởng lớn đến giá phòng trung bình. Đặc trưng tổng số khách (total_guests) và loại phòng đặt trước (reserved_room_type) cũng nằm trong nhóm đầu, phản ánh mối liên hệ giữa quy mô đoàn khách, loại phòng với giá trị đặt phòng. Ngoài ra, các biến về thị trường, kênh phân phối, loại khách hàng, số lượng yêu cầu đặc biệt, số lần thay đổi đặt phòng… cũng góp phần quan trọng vào khả năng dự báo.

Việc lựa chọn các đặc trưng này giúp mô hình tận dụng tối đa các yếu tố có ý nghĩa thực tiễn, đồng thời loại bỏ các biến dư thừa hoặc ít liên quan, từ đó nâng cao hiệu quả và độ chính xác của mô hình dự báo giá phòng.

4.2 Danh sách và cách sử dụng các mô hình học máy

Danh sách các mô hình học máy sẽ được áp dụng để xây dựng và so sánh hiệu suất trong dự án này. Các mô hình bao gồm cả phương pháp truyền thống và hiện đại và tối ưu hóa.

Danh sách mô hình

- Linear Regression: Một mô hình hồi quy tuyến tính cơ bản, phù hợp để dự đoán các giá trị liên tục dựa trên mối quan hệ tuyến tính giữa các đặc trưng.

- Random Forest: Một thuật toán dựa trên cây quyết định, sử dụng nhiều cây quyết định để tăng độ chính xác và giảm overfitting.

- XGBoost: Một triển khai hiệu quả của Gradient Boosting, tối ưu hóa tốc độ và hiệu suất với các kỹ thuật như regularization.

- CatBoost: Một biến thể của Gradient Boosting, tối ưu cho dữ liệu phân loại (categorical) và xử lý tốt các giá trị thiếu.

- LightGBM: Một framework Gradient Boosting nhẹ và nhanh, sử dụng thuật toán histogram-based để cải thiện hiệu suất.

Cách sử dụng:

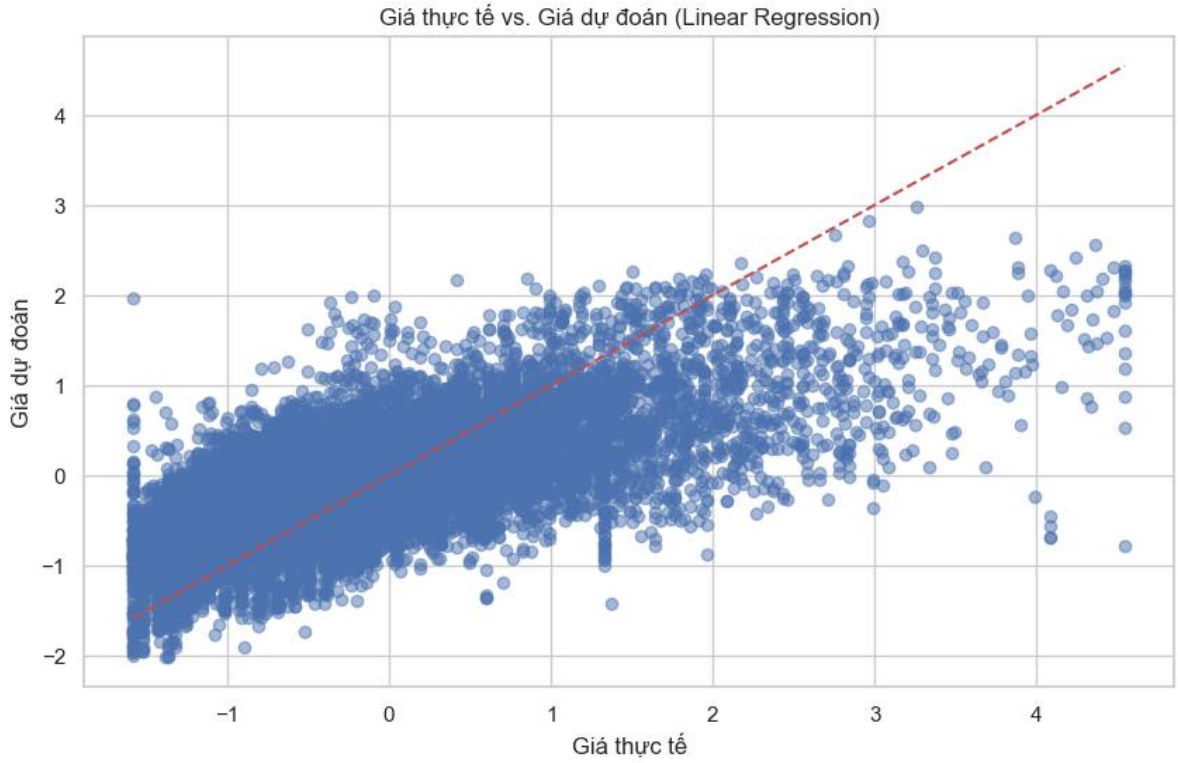
- Các mô hình sẽ được huấn luyện trên tập dữ liệu đã được chia (train set và test set).

- So sánh hiệu suất thông qua các chỉ số như MSE, RMSE hoặc R^2 để chọn mô hình tốt nhất.

- Tối ưu hóa siêu tham số (nếu cần) bằng các phương pháp như Grid Search hoặc Random Search.

4.2.1 Mô hình hồi quy tuyến tính (Linear Regression)

```
Kết quả mô hình hồi quy tuyến tính:
MSE: 0.54
RMSE: 0.74
MAE: 0.55
R²: 0.4560
```



Hình 4. 7 Kết quả mô hình Linear Regression

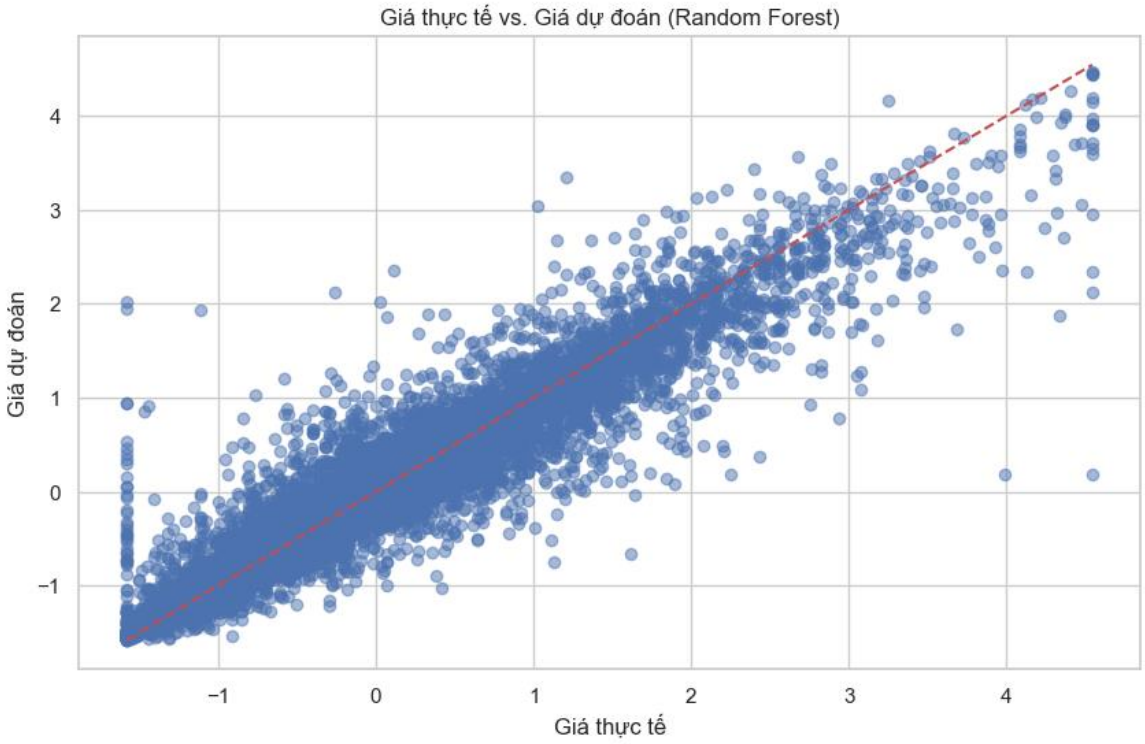
Kết luận về mô hình hồi quy tuyến tính

Kết quả đánh giá mô hình hồi quy tuyến tính cho thấy các chỉ số như MSE = 0.54, RMSE = 0.74, MAE = 0.55 và hệ số xác định R^2 = 0.4560. Điều này cho thấy mô hình có khả năng dự đoán ở mức trung bình, giải thích được khoảng 45.6% phương sai của dữ liệu thực tế. Biểu đồ so sánh giữa giá trị thực tế và giá trị dự đoán cho thấy các điểm dữ liệu phân tán khá rộng quanh đường chéo lý tưởng, cho thấy mô hình còn tồn tại sai số dự đoán đáng kể.

Nguyên nhân có thể đến từ việc dữ liệu có tính phi tuyến hoặc tồn tại các yếu tố ảnh hưởng mà mô hình tuyến tính chưa nắm bắt được.

4.2.2 Mô hình Random Forest

```
Kết quả mô hình Random Forest:
MSE: 0.10
RMSE: 0.32
MAE: 0.19
R²: 0.8954
```



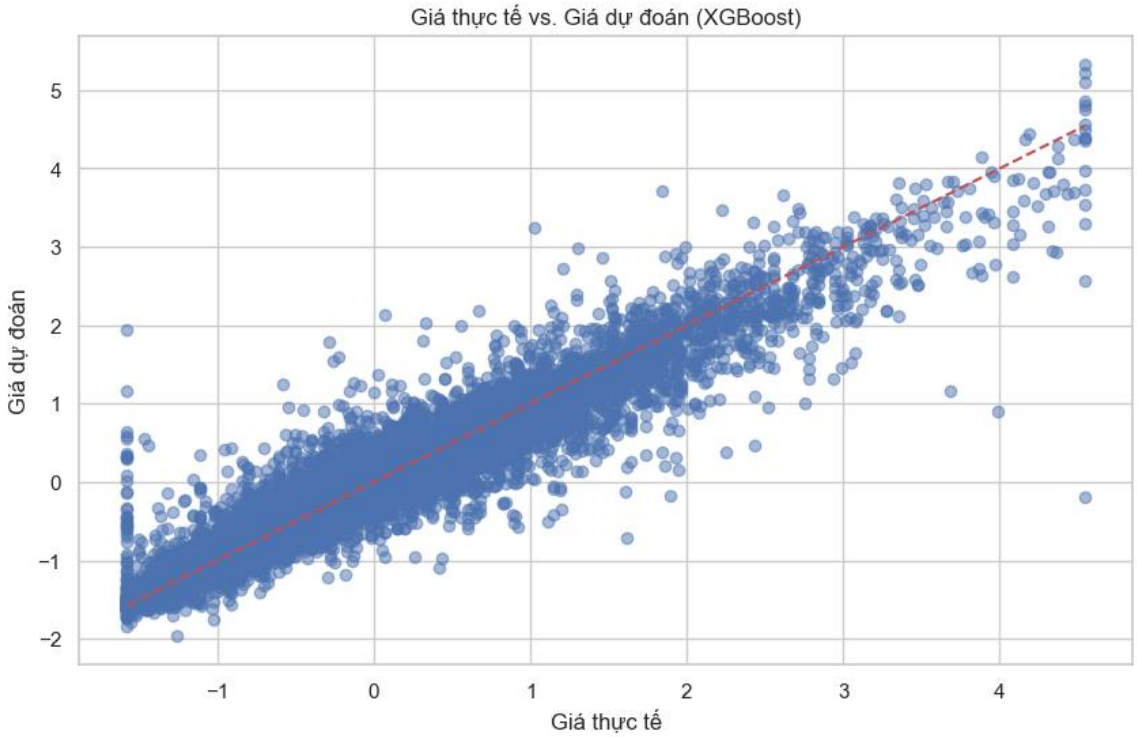
Hình 4. 8 Kết quả mô hình Random Forest

Kết luận cho phần đánh giá mô hình Random Forest:

Kết quả đánh giá mô hình Random Forest cho thấy mô hình dự đoán khá tốt với các chỉ số sai số thấp (MSE = 0.10, RMSE = 0.32, MAE = 0.19) và hệ số xác định R^2 đạt 0.8954. Điều này chứng tỏ mô hình có khả năng giải thích được khoảng 89.54% phương sai của dữ liệu thực tế. Biểu đồ so sánh giữa giá trị thực tế và giá trị dự đoán cho thấy các điểm dữ liệu phân bố gần đường chéo, thể hiện sự tương quan mạnh giữa giá trị dự đoán và giá trị thực tế. Tuy nhiên, vẫn còn một số điểm ngoại lệ và sai số nhất định, cho thấy mô hình có thể được cải thiện thêm bằng cách tối ưu tham số hoặc thử nghiệm các thuật toán khác.

4.2.3 Mô hình XGBoost

```
Kết quả mô hình XGBoost:
MSE: 0.10
RMSE: 0.32
MAE: 0.22
R²: 0.8969
```



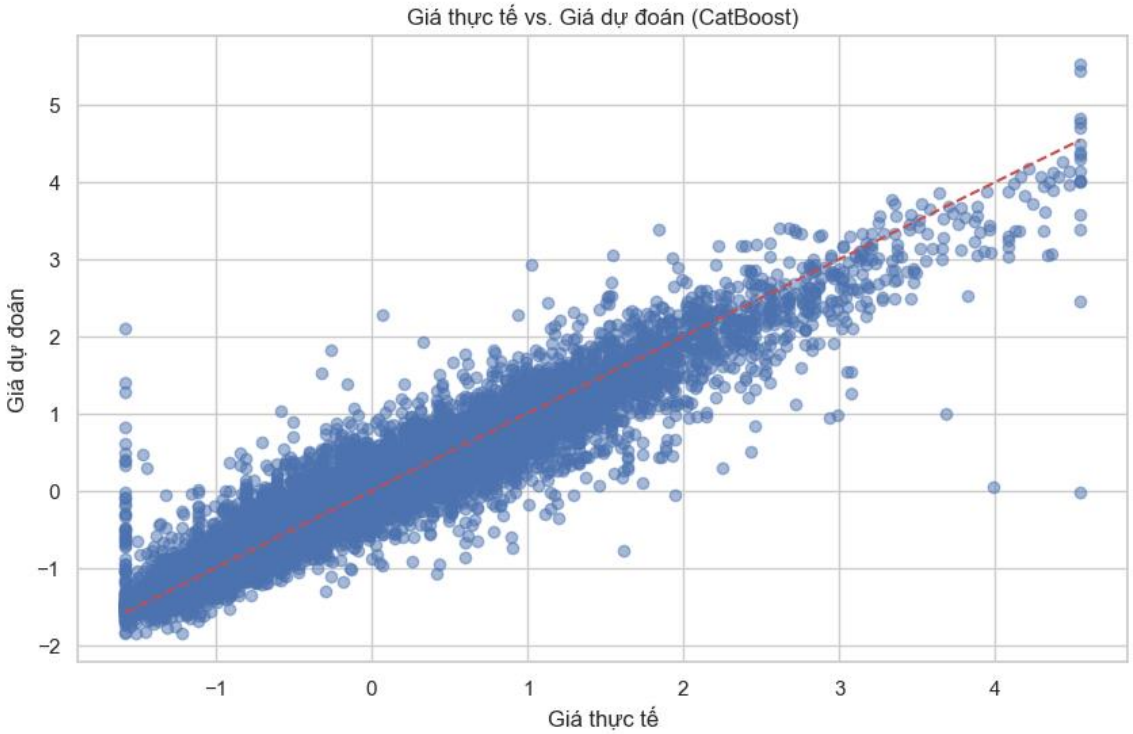
Hình 4. 9 Kết quả mô hình XGBoost

Kết luận cho phần đánh giá mô hình XGBoost:

Mô hình XGBoost cho kết quả dự báo rất tốt với các chỉ số đánh giá như MSE = 0.10, RMSE = 0.32, MAE = 0.22 và hệ số xác định R^2 đạt 0.8969. Điều này cho thấy mô hình giải thích được khoảng 89.69% phương sai của dữ liệu thực tế, thể hiện khả năng dự báo mạnh mẽ và ổn định. Biểu đồ so sánh giữa giá trị thực tế và giá trị dự đoán cho thấy các điểm dữ liệu tập trung gần đường chéo, chứng tỏ mô hình dự đoán sát với thực tế. Tuy vẫn còn một số điểm ngoại lệ, nhưng nhìn chung sai số nhỏ và phân bố hợp lý.

4.2.4 Mô hình CatBoost

```
Kết quả mô hình CatBoost:
MSE: 0.11
RMSE: 0.33
MAE: 0.22
R²: 0.8937
```



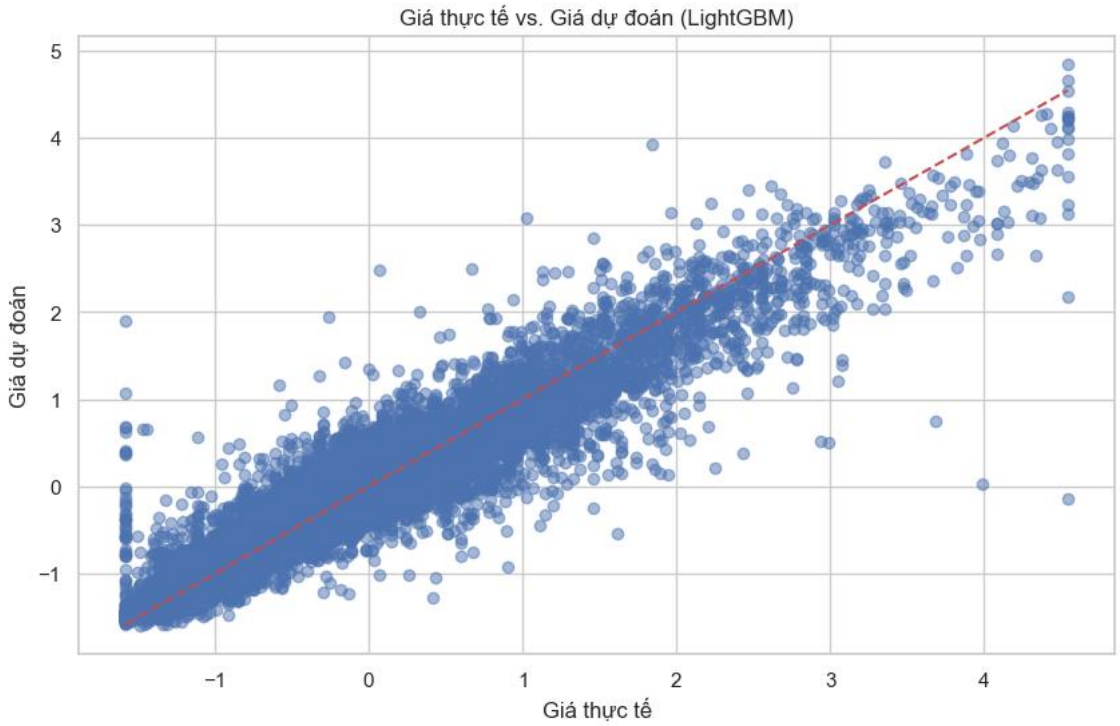
Hình 4. 10 Kết quả mô hình CatBoost

Kết luận cho phần đánh giá mô hình CatBoost:

Mô hình CatBoost đạt kết quả dự báo tốt với các chỉ số MSE = 0.11, RMSE = 0.33, MAE = 0.22 và hệ số xác định $R^2 = 0.8937$. Điều này cho thấy mô hình giải thích được khoảng 89.37% phương sai của dữ liệu thực tế, thể hiện năng lực dự báo ổn định và đáng tin cậy. Biểu đồ so sánh giữa giá trị thực tế và giá trị dự đoán cho thấy các điểm dữ liệu phân bố gần đường chéo, chứng tỏ mô hình dự đoán sát với thực tế. Tuy vẫn còn một số điểm ngoại lệ, nhưng sai số tổng thể nhỏ và phân bố hợp lý

4.2.5 Mô hình LightGBM

```
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.002749 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 836
[LightGBM] [Info] Number of data points in the train set: 49416, number of used features: 25
[LightGBM] [Info] Start training from score -0.000476
Kết quả mô hình LightGBM:
MSE: 0.12
RMSE: 0.34
MAE: 0.24
R²: 0.8816
```



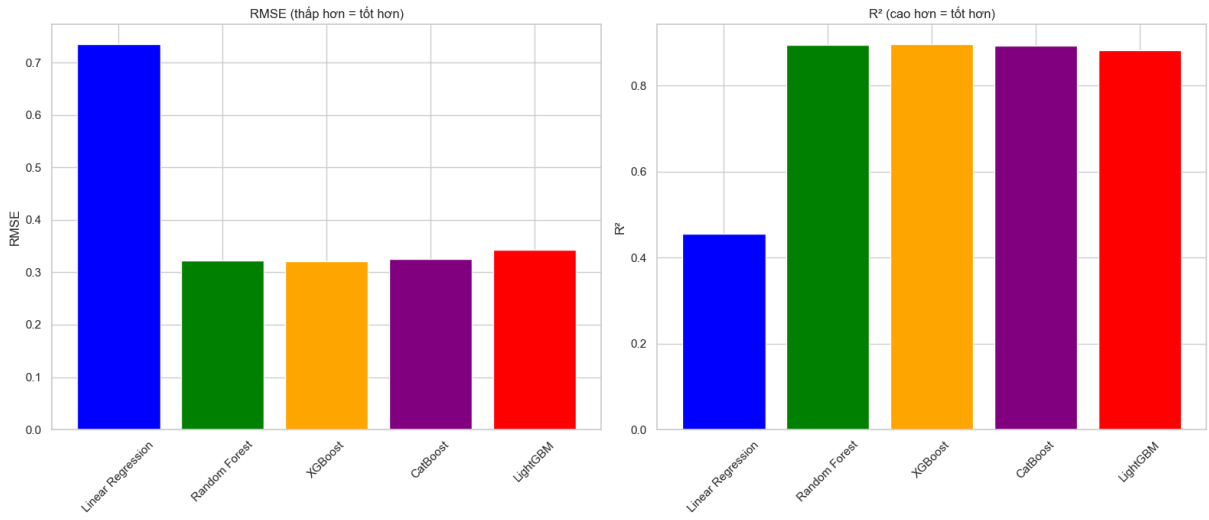
Hình 4. 11 Kết quả mô hình LightGBM

Kết luận cho phần đánh giá mô hình LightGBM:

Mô hình LightGBM cho kết quả dự báo khá tốt với các chỉ số MSE = 0.12, RMSE = 0.34, MAE = 0.24 và hệ số xác định $R^2 = 0.8816$. Điều này cho thấy mô hình giải thích được khoảng 88.16% phương sai của dữ liệu thực tế, thể hiện năng lực dự báo ổn định. Biểu đồ so sánh giữa giá trị thực tế và giá trị dự đoán cho thấy các điểm dữ liệu tập trung gần đường chéo, chứng tỏ mô hình dự đoán sát với thực tế. Tuy nhiên, so với các mô hình khác, LightGBM có sai số lớn hơn một chút và vẫn còn một số điểm ngoại lệ

4.3. So sánh các mô hình

Thống kê lại kết quả các mô hình, so sánh và đánh giá hiệu quả



Hình 4. 12 Thống kê lại kết quả các mô hình

```
Bảng so sánh các mô hình:
              Model      RMSE       MAE        R²
0  Linear Regression  0.735871  0.553226  0.455973
1      Random Forest  0.322725  0.192956  0.895364
2            XGBoost  0.320349  0.215220  0.896899
3           CatBoost  0.325275  0.223374  0.893704
4           LightGBM  0.343336  0.235154  0.881571
```

Ý nghĩa các thông số

- RMSE (Root Mean Squared Error): nhạy với lỗi lớn, thể hiện "trung bình bình phương" sai số. Muốn thấp để hạn chế outlier dự đoán quá lệch.

- MAE (Mean Absolute Error): sai số tuyệt đối bình quân, dễ diễn giải (ví dụ MAE = 9.8 nghĩa là trung bình sai lệch ~9.8 đơn vị tiền tệ).

- R² (Coefficient of Determination): tỷ lệ phương sai được mô hình "giải thích". Gần 1 càng tốt.

Chúng ta sẽ chọn chỉ số RMSE để đánh giá lựa chọn ra mô hình tối ưu cho tài toán nhất vì RMSE phạt nặng lỗi lớn (outliers) bằng cách bình phương sai số trước khi tính trung bình, tương đương độ lệch chuẩn của lỗi dự báo và thân thiện với tối ưu hóa gradient (như XGBoost, LightGBM, NN…), giúp kiểm soát rủi ro tài chính khi giá phòng biến động mạnh. Vì cùng đơn vị với ADR (VND), RMSE dễ diễn giải ("sai trung bình ±X đồng, outlier lên đến ±Y đồng") và là chuẩn so sánh quốc tế trong các benchmark regression.

```
Mô hình có hiệu suất tốt nhất là XGBoost với RMSE = 0.3203
```

4.4. Phân tích mô hình tốt nhất

4.4.1 Tối ưu hóa siêu tham số cho mô hình XGBoost với Optuna [6]

Thực hiện việc tối ưu hóa siêu tham số cho mô hình XGBoost bằng thư viện **Optuna**, sử dụng phương pháp **Bayesian Optimization**. Mục tiêu là tìm ra bộ siêu tham số tối ưu để giảm thiểu giá trị RMSE (Root Mean Squared Error) thông qua kiểm định chéo **5-Fold (K-Fold Cross-Validation)**. Sau khi tìm được tham số tốt nhất, mô hình **XGBoost** được huấn luyện lại và đánh giá hiệu suất trên tập kiểm tra bằng các chỉ số RMSE, MAE và R^2.

Việc tối ưu được thực hiện như sau:

Định nghĩa hàm objective cho Optuna:

các siêu tham số (hyperparameters)

Code:

```
    params = {
        'n_estimators': trial.suggest_int('n_estimators', 50,
300),
        'max_depth': trial.suggest_int('max_depth', 3, 10),
        'learning_rate': trial.suggest_float('learning_rate',
0.01, 0.3),
        'subsample': trial.suggest_float('subsample', 0.7,
1.0),
        'colsample_bytree':
trial.suggest_float('colsample_bytree', 0.6, 1.0),
        'gamma': trial.suggest_float('gamma', 1e-8, 5),
        'reg_alpha': trial.suggest_float('reg_alpha', 1e-8,
5),
        'reg_lambda': trial.suggest_float('reg_lambda', 1e-8,
5),
        'min_child_weight':
trial.suggest_int('min_child_weight', 1, 10),
        'max_delta_step': trial.suggest_int('max_delta_step',
0, 10),
```

```
        'objective': 'reg:squarederror',
        'random_state': 42
    }
```

Ý nghĩa:

Optuna sẽ thử nhiều giá trị khác nhau cho các tham số trong params để tìm ra tổ hợp tốt nhất giúp mô hình có kết quả tốt nhất (RMSE thấp nhất).

Sử dụng KFold Cross-Validation

Code:

```
kf = KFold(n_splits=5, shuffle=True, random_state=42)
```

Ý nghĩa

Giúp đánh giá mô hình chính xác hơn bằng cách chia dữ liệu nhiều lần. Tránh overfitting khi đánh giá trên 1 tập validation duy nhất.

Huấn luyện và tính toán điểm số

Code:

```
    for train_idx, valid_idx in kf.split(X_train):
        X_fold_train, X_fold_valid = X_train.iloc[train_idx], 
X_train.iloc[valid_idx]
        y_fold_train, y_fold_valid = y_train.iloc[train_idx], 
y_train.iloc[valid_idx]
        model = XGBRegressor(**params)
        model.fit(
            X_fold_train, y_fold_train,
            eval_set=[(X_fold_valid, y_fold_valid)],
            verbose=False
        )
        preds = model.predict(X_fold_valid)
        score = np.sqrt(mean_squared_error(y_fold_valid, 
preds))
        scores.append(score)
```

Ý nghĩa:

Huấn luyện mô hình với các siêu tham số đang thử nghiệm và tính RMSE của fold đó

Cuối cùng:

Trả về RMSE trung bình của 5 fold → mục tiêu để Minimize.

Tối ưu hóa với Optuna

Code:

```
study = optuna.create_study(direction='minimize')
study.optimize(objective, n_trials=30)
```

Ý nghĩa:

Tạo một study với mục tiêu minimize (giảm thiểu RMSE).

Thực hiện nhiều lần thử nghiệm (n_trials=30), mỗi lần thử một bộ tham số khác nhau.

Huấn luyện mô hình với tham số tối ưu

Code:

```
best_xgb = xgb.XGBRegressor(objective='reg:squarederror',
random_state=42, **study.best_params)
best_xgb.fit(X_train, y_train)
best_params = study.best_params.copy()
```

Ý nghĩa:

Sau khi tìm được bộ tham số tốt nhất, huấn luyện lại mô hình XGBoost trên toàn bộ tập train. Đảm bảo mô hình cuối cùng sử dụng bộ tham số tối ưu nhất đã tìm được.

Đánh giá mô hình tối ưu trên tập kiểm tra

Code:

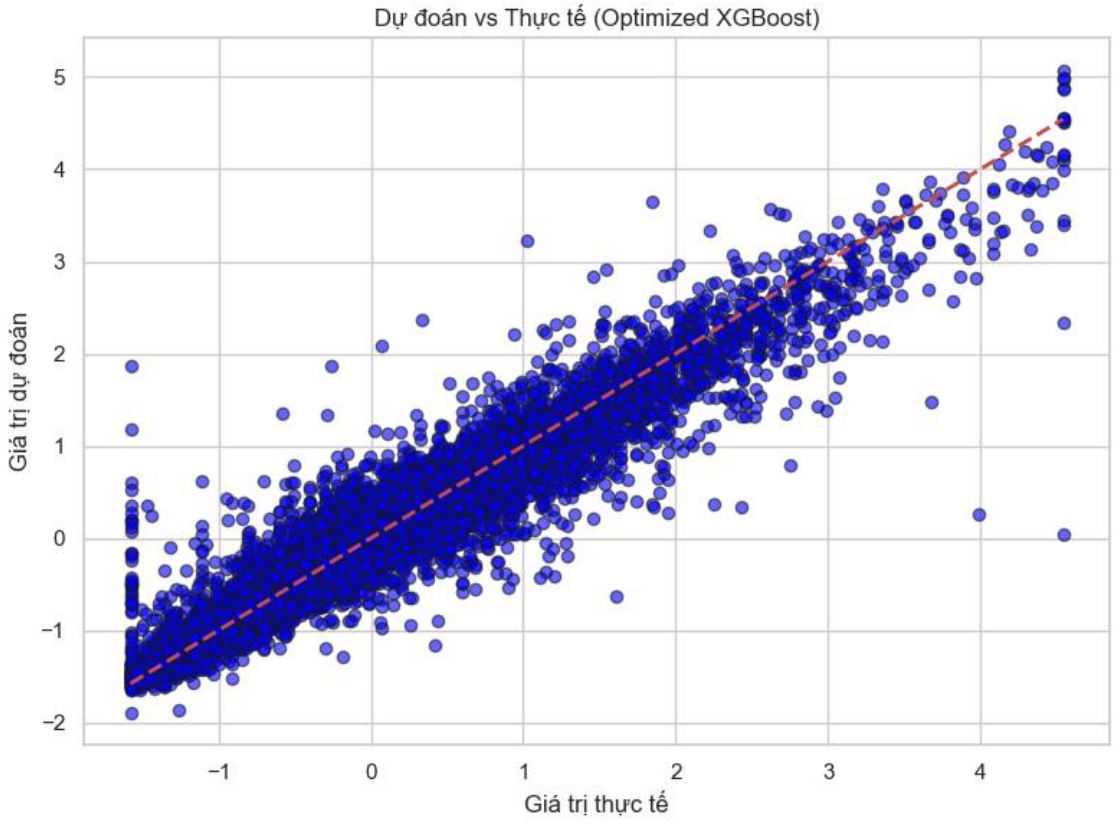
```
y_pred_best_xgb = best_xgb.predict(X_test)
rmse_best_xgb, mae_best_xgb, r2_best_xgb =
evaluate_model(y_test, y_pred_best_xgb, "Optimized XGBoost")
```

Ý nghĩa:

Dự đoán trên tập test với mô hình đã tối ưu. Tính toán các chỉ số RMSE, MAE, R2 Score để đánh giá hiệu quả thực tế. Kiểm tra xem mô hình tối ưu có thực sự cải thiện hiệu suất dự đoán trên dữ liệu chưa từng thấy.

Đánh giá kết quả sau quá trình tối ưu:

```
=== Đánh giá mô hình: Optimized XGBoost ===
RMSE: 0.2981
MAE: 0.1964
R2 Score: 0.9107
```



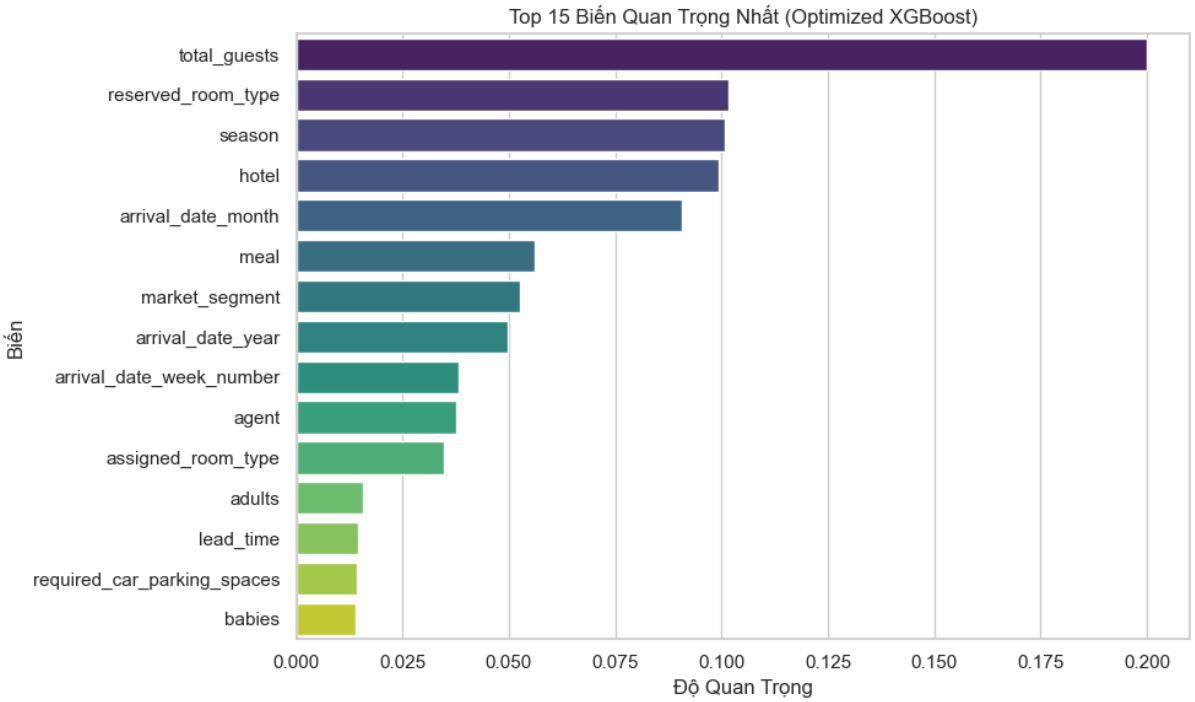
Hình 4. 13 Kết quả Optimized XGBoost

Sau khi huấn luyện mô hình XGBoost ban đầu, nhóm đã tiến hành tối ưu hóa mô hình thông qua điều chỉnh các siêu tham số. Kết quả cho thấy hiệu suất của mô hình đã được cải thiện rõ rệt. Cụ thể, chỉ số RMSE giảm từ 0.32 xuống còn 0.2981, tương đương mức giảm khoảng 6.84%, trong khi MAE giảm từ 0.22 xuống còn 0.1964, tức giảm khoảng 10.55%. Đồng thời, hệ số xác định R^2 cũng tăng từ 0.8969 lên 0.9107, phản ánh rằng mô hình sau tối ưu đã giải thích tốt hơn phương sai của dữ liệu. Những cải thiện này cho thấy việc tối ưu hóa mô hình không chỉ giúp giảm sai số dự đoán mà còn nâng cao khả năng tổng quát hóa của mô hình. Như vậy, quá trình điều chỉnh siêu tham số đóng vai trò quan trọng trong việc nâng cao hiệu suất và độ chính xác của mô hình học máy, từ đó góp phần tạo ra những dự báo đáng tin cậy hơn.

4.4.2 Phân tích và trực quan hóa đặc trưng quan trọng với XGBoost

Các biến quan trọng nhất:

	Feature	Importance
1	total_guests	0.200077
2	reserved_room_type	0.101540
6	season	0.100815
3	hotel	0.099246
0	arrival_date_month	0.090634
7	meal	0.055972
5	market_segment	0.052664
8	arrival_date_year	0.049604
4	arrival_date_week_number	0.038082
9	agent	0.037524
14	assigned_room_type	0.034818
13	adults	0.015576
10	lead_time	0.014608
15	required_car_parking_spaces	0.014237
23	babies	0.013950
17	previous_bookings_not_canceled	0.013572
11	country	0.011026
24	children	0.010039
19	distribution_channel	0.009320
22	is_repeated_guest	0.008931



Hình 4. 14 15 Biến quan trọng nhất (*Optimized XGBoost*)

Ý nghĩa:

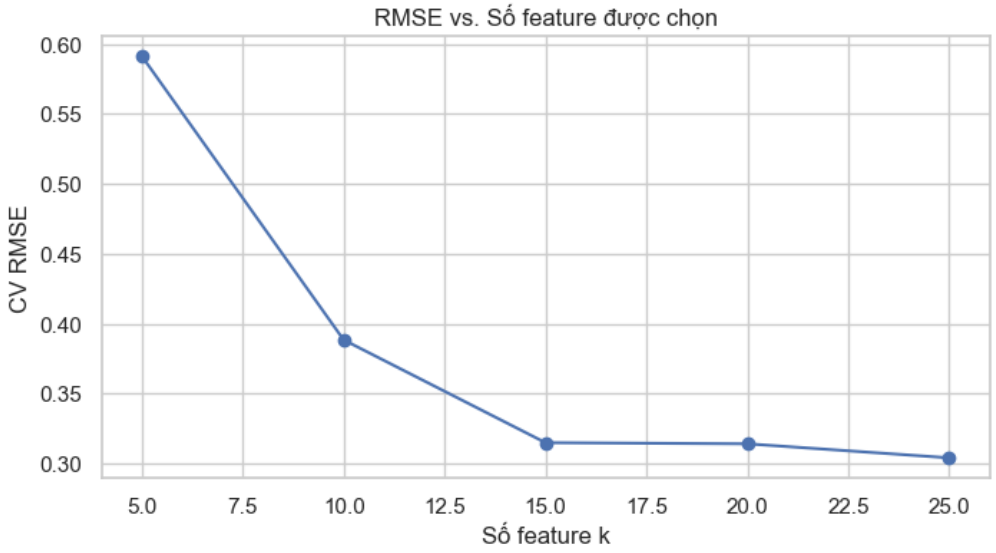
Qua hai hình trên, ta thấy rằng mô hình XGBoost đã xác định được một số biến có ảnh hưởng lớn nhất đến khả năng dự đoán kết quả. Cụ thể như total_guests, reserved_room_type, assigned_room_type, season, hotel,.. Việc xác định các biến quan trọng giúp doanh nghiệp khách sạn tập trung vào các yếu tố then chốt để tối ưu hóa dịch vụ, dự báo nhu cầu, và xây dựng các chính sách phù hợp nhằm giảm tỷ lệ hủy phòng hoặc nâng cao trải nghiệm khách hàng

4.4.3. Dùng RFECV tự động chọn số features

Bây giờ chúng ta sẽ xác định xem nên chọn bao nhiêu biến quan trọng nhất để vừa đảm bảo mô hình dự đoán tốt, vừa tránh dư thừa, giảm độ phức tạp và nguy cơ overfitting. Đây là một bước quan trọng trong quy trình chọn lọc đặc trưng (feature selection) trong học máy.

```
k =  5 → CV RMSE = 0.5912
k = 10 → CV RMSE = 0.3884
k = 15 → CV RMSE = 0.3150
k = 20 → CV RMSE = 0.3142
k = 25 → CV RMSE = 0.3041

Tổng hợp RMSE theo số feature:
     k      rmse
0   5  0.591219
1  10  0.388436
2  15  0.314976
3  20  0.314180
4  25  0.304144
```



Hình 4. 15 RMSE vs Số feature được chọn

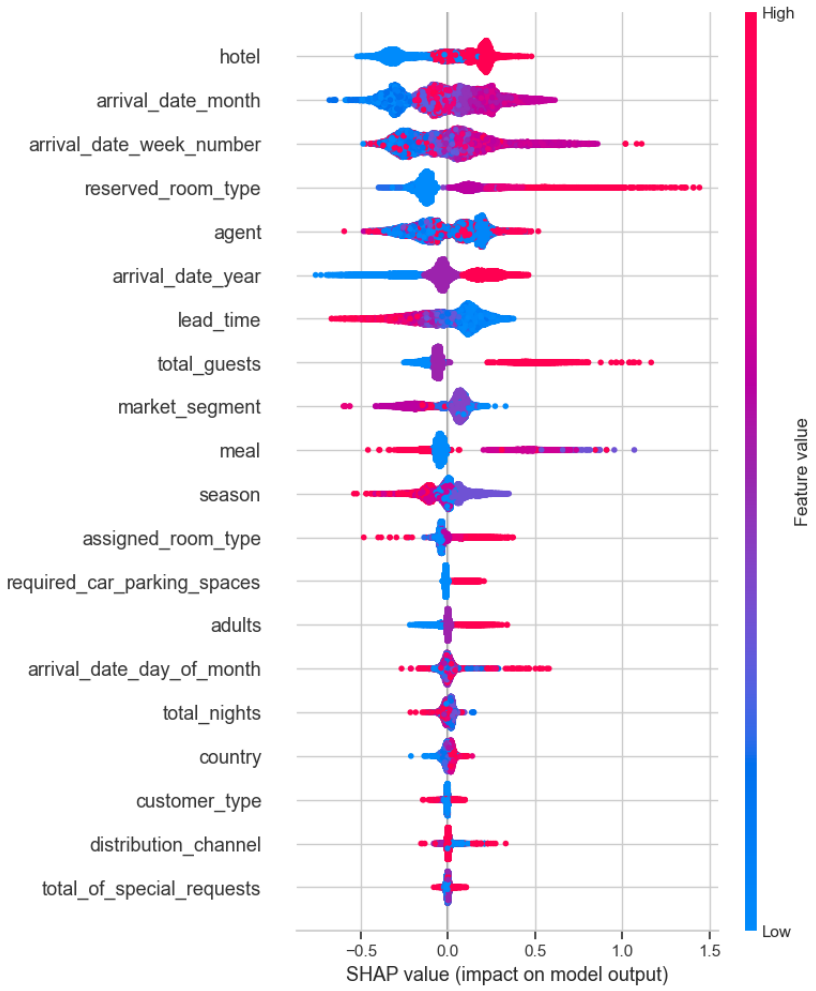
Thí nghiệm lựa chọn số lượng đặc trưng (k) cho thấy RMSE giảm dần khi tăng số feature từ 5 đến 25. Cụ thể, RMSE giảm từ 0.5912 (k=5) xuống còn 0.3041 (k=25). Mức giảm rõ rệt nhất xảy ra trong khoảng từ k=5 đến k=15, sau đó ổn định dần ở các giá trị k cao hơn. Biểu đồ RMSE theo k cho thấy hiệu suất mô hình cải thiện khi bổ sung thêm đặc trưng, tuy nhiên lợi ích biên giảm dần sau k=15. Dựa trên kết quả này, k=25 được chọn là cấu hình tối ưu, cho RMSE thấp nhất mà không gây quá khớp (overfitting) trong đánh giá chéo.

Bây giờ em sẽ sử dụng mô hình XGBoost được huấn luyện lại chỉ với 15 đặc trưng này trên tập huấn luyện, và đánh giá hiệu suất trên tập kiểm tra thông qua các chỉ số RMSE, MAE và R^2.

```
=== Đánh giá mô hình: Optimized XGBoost - 15 features ===
RMSE: 0.3060
MAE: 0.2004
R2 Score: 0.9059
```

Việc chỉ sử dụng 15 đặc trưng quan trọng nhất vẫn mang lại hiệu quả dự đoán cao cho mô hình XGBoost. Điều này cho thấy rằng lựa chọn đặc trưng hợp lý không chỉ giúp đơn giản hóa mô hình mà còn có thể duy trì, thậm chí cải thiện hiệu suất dự đoán. Nhờ đó, mô hình có thể được triển khai hiệu quả trong thực tiễn mà không cần sử dụng toàn bộ các biến đầu vào, góp phần tiết kiệm tài nguyên tính toán và nâng cao khả năng giải thích của kết quả dự báo.

4.4.4. Vẽ biểu đồ Shap

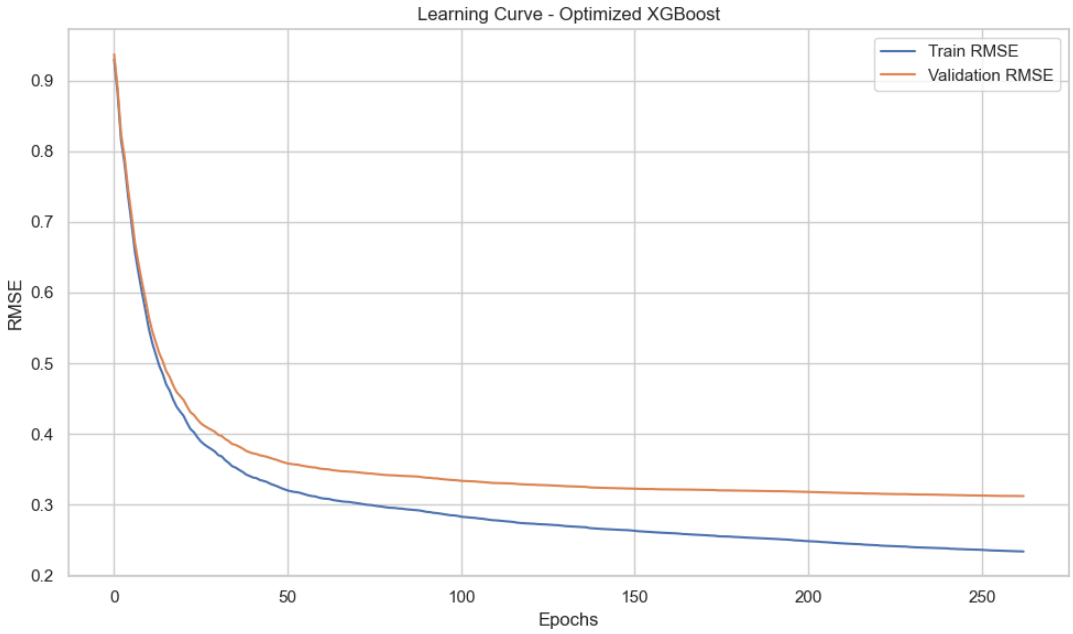


Hình 4. 16 Biểu đồ SHAP

Biểu đồ SHAP summary trên cho thấy mức độ quan trọng và ảnh hưởng của từng đặc trưng (feature) đến dự đoán của mô hình. Các đặc trưng như hotel, arrival_date_month, arrival_date_week_number, reserved_room_type, và agent có tác động lớn nhất đến đầu ra của mô hình, thể hiện qua phạm vi giá trị SHAP rộng và mật độ điểm cao. Màu sắc từ xanh (giá trị đặc trưng thấp) đến đỏ (giá trị đặc trưng cao) cho thấy mối liên hệ giữa giá trị đặc trưng và tác động của nó: ví dụ, giá trị cao của một số đặc trưng có thể làm tăng hoặc giảm xác suất dự đoán tuỳ vào hướng của giá trị SHAP.

Các đặc trưng như total_of_special_requests, distribution_channel, và customer_type có ảnh hưởng nhỏ hơn, thể hiện qua phạm vi giá trị SHAP hẹp và ít điểm dữ liệu hơn. Điều này gợi ý rằng mô hình chủ yếu dựa vào các đặc trưng liên quan đến thông tin đặt phòng, thời gian và loại phòng để đưa ra dự đoán.

4.4.5. Vẽ đường cong học (Learning Curve) cho mô hình XGBoost tối ưu



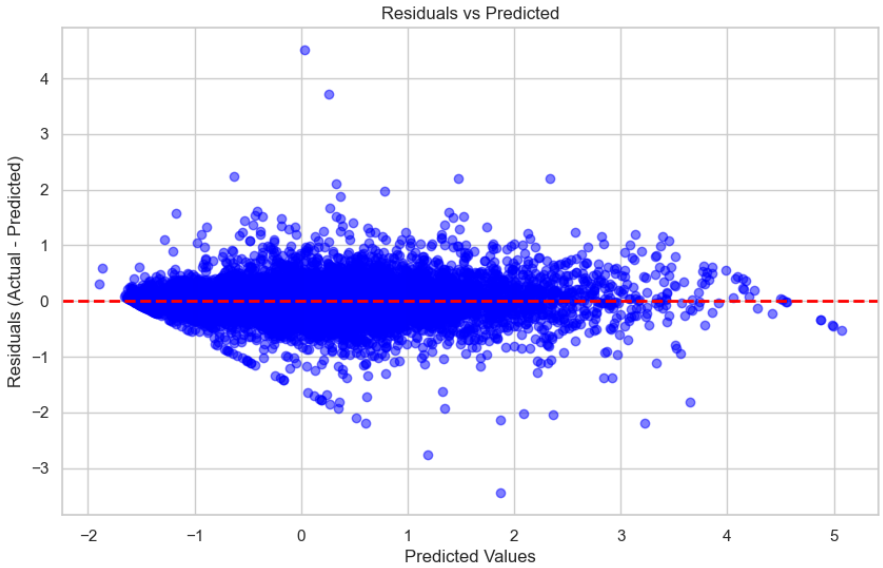
Hình 4. 17 Đường cong học (Learning Curve)

Mô hình XGBoost học tốt trên cả tập huấn luyện và validation, thể hiện qua việc RMSE giảm đều và ổn định. Sự chênh lệch nhẹ giữa train và validation RMSE sau ~200 epochs cho thấy overfitting ở mức thấp và có kiểm soát. Hiệu suất tổng quát hóa tốt, không có dấu hiệu học quá mức nghiêm trọng.

4.4.6. Phân tích và trực quan hóa lỗi dự đoán của mô hình XGBoost

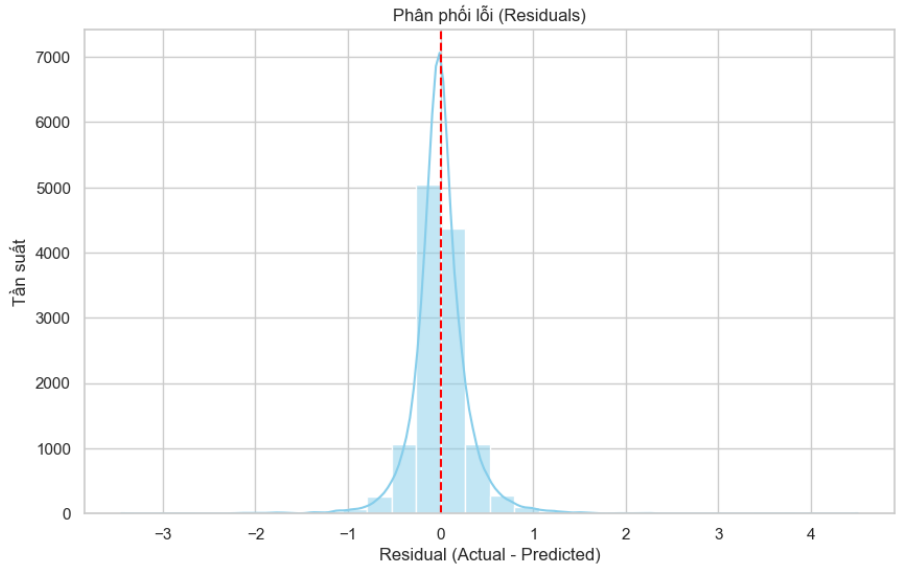
```
Mean of residuals: -0.0007420825194606207
Std of residuals: 0.2981425168660253
Min residual: -3.446682894814192
Max residual: 4.508613292890277
Lỗi tuyệt đối trung bình: 0.20
Lỗi phần trăm trung bình: -10.74%
```

Nhận xét: Phần dư của mô hình có trung bình gần bằng 0 (−0.0007), cho thấy sai số không bị lệch về một phía, đồng thời xác nhận tính khách quan trong dự đoán. Độ lệch chuẩn của phần dư là 0.2981, phản ánh mức độ biến động sai số ở mức thấp. Khoảng giá trị phần dư dao động từ −3.45 đến 4.51, với một vài trường hợp ngoại lệ cần được xem xét kỹ. Lỗi tuyệt đối trung bình đạt 0.20, cho thấy mô hình dự đoán chính xác trong đa số trường hợp. Tuy nhiên, lỗi phần trăm trung bình là −10.74%, cho thấy mô hình có xu hướng đánh giá cao hơn thực tế trong một số tình huống.



Hình 4. 18 Biểu đồ Residuals vs Predicted

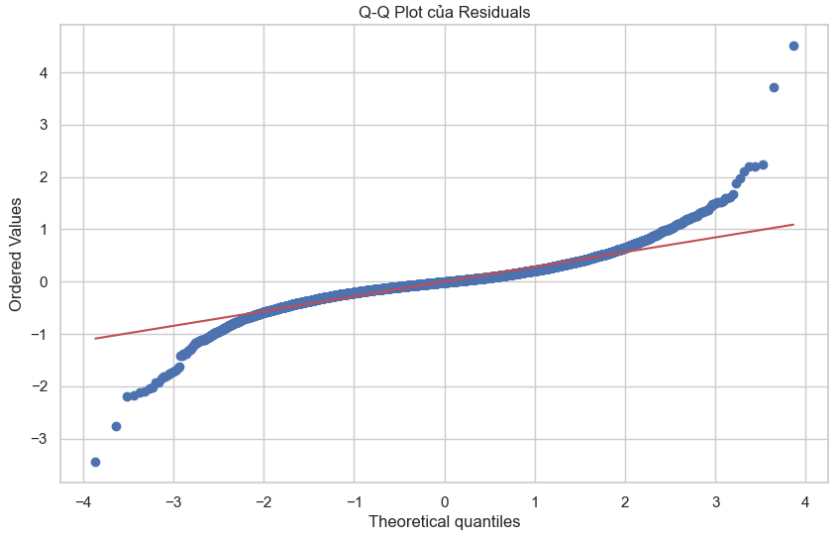
Nhận xét: Biểu đồ Residuals vs Predicted minh họa mối quan hệ giữa giá trị dự đoán và phần dư (sai số dự đoán). Các điểm dữ liệu phân bố tương đối đều quanh trục hoành (residual = 0), cho thấy mô hình không có dấu hiệu sai lệch hệ thống (bias). Tuy nhiên, phần dư có xu hướng phân tán rộng hơn ở hai đầu trục dự đoán, cho thấy mô hình hoạt động kém hiệu quả với các giá trị ngoại biên – một đặc điểm thường gặp khi xử lý dữ liệu phi tuyến hoặc phân bố không đồng đều



Hình 4. 19 Phân phối lỗi (Residuals)

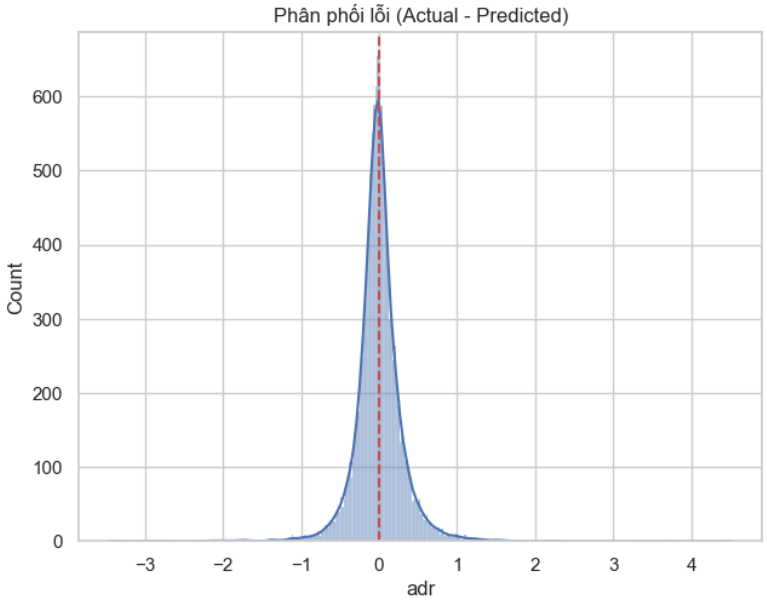
Nhận xét: Biểu đồ phân phối phần dư cho thấy sai số dự đoán của mô hình XGBoost tập trung chủ yếu quanh giá trị 0 và có dạng gần đối xứng, chứng tỏ mô hình không có sai lệch hệ thống (bias). Đường phân phối mịn và đỉnh cao tại trung tâm cho thấy phần lớn dự đoán

gần với giá trị thực tế. Tuy nhiên, phân phối hơi lệch trái nhẹ, cho thấy mô hình có xu hướng đánh giá cao hơn thực tế trong một số trường hợp. Tổng thể, phân phối phần dư gần với phân phối chuẩn, cho thấy giả định về phần dư độc lập và phân phối đều được đáp ứng tương đối tốt



Hình 4. 20 Q-Q Plot của Residuals

Nhận xét: Biểu đồ Q-Q plot cho thấy phần lớn các điểm dữ liệu nằm gần đường chéo, cho thấy phần dư gần tuân theo phân phối chuẩn. Tuy nhiên, tại hai đầu phân phối (đuôi trái và phải), các điểm lệch rõ rệt khỏi đường lý thuyết, biểu hiện hiện tượng lệch đuôi (heavy tails). Điều này cho thấy mô hình có thể chưa xử lý tốt các giá trị ngoại biên hoặc tồn tại sai số lớn trong một số trường hợp cực trị.

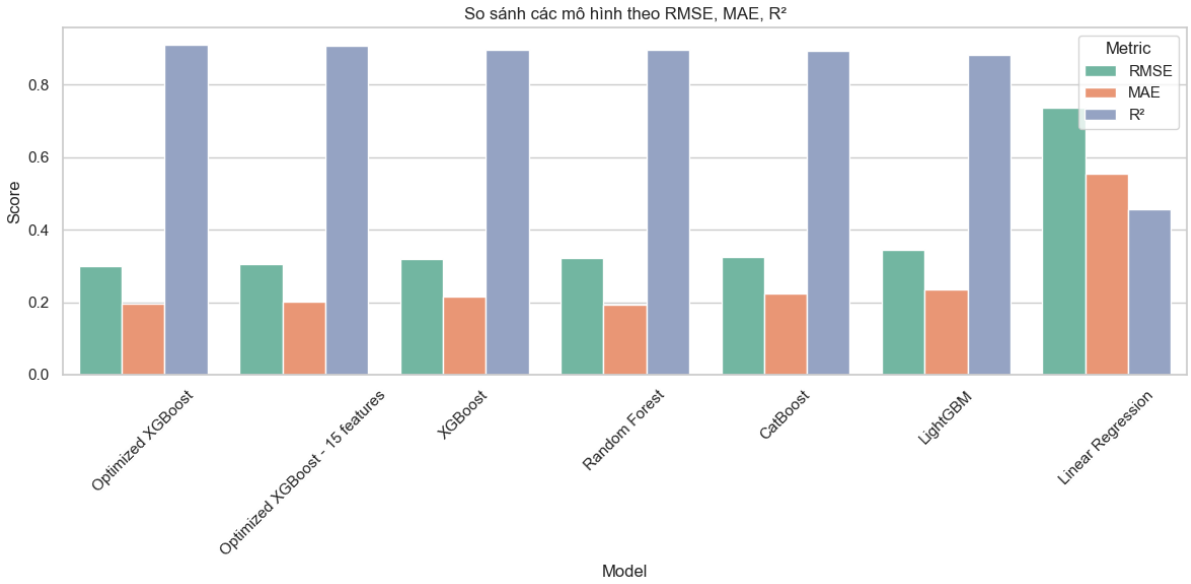


Hình 4. 21 Phân phối lỗi (Actual – Predicted)

Nhận xét: Biểu đồ phân phối phần dư cho thấy các sai số tập trung chủ yếu quanh giá trị 0 và có dạng đối xứng, chỉ ra rằng mô hình không có xu hướng dự đoán thiên lệch về một phía. Tuy nhiên, đỉnh phân phối nhọn và hai đuôi kéo dài cho thấy phần dư có phân phối lệch chuẩn (leptokurtic), thể hiện khả năng tồn tại một số điểm sai số lớn. Nhìn chung, mô hình có độ chính xác cao đối với phần lớn quan sát.

4.5. So sánh tất cả mô hình được sử dụng.

So sánh hiệu suất của nhiều mô hình học máy (Linear Regression, Random Forest, XGBoost, CatBoost, LightGBM, và Optimized XGBoost) dựa trên các chỉ số RMSE, MAE, và R^2.



Hình 4. 22 Tổng kết các mô hình đã dùng

Bảng so sánh các mô hình (sắp xếp theo RMSE):

Model	RMSE	MAE	R^2
Optimized XGBoost	0.298143	0.196363	0.910697
Optimized XGBoost - 15 features	0.305972	0.200363	0.905945
XGBoost	0.320349	0.21522	0.896899
Random Forest	0.322725	0.192956	0.895364
CatBoost	0.325275	0.223374	0.893704
LightGBM	0.343336	0.235154	0.881571
Linear Regression	0.735871	0.553226	0.455973

So sánh:

Dựa trên ba chỉ số đánh giá RMSE, MAE và R^2, mô hình Optimized XGBoost đạt hiệu suất tốt nhất với RMSE = 0.2981, MAE = 0.1964 và R^2 = 0.9107. Phiên bản tối ưu sử dụng 15 đặc trưng quan trọng nhất vẫn duy trì hiệu suất cao (RMSE = 0.3060, R^2 = 0.9060), chỉ giảm nhẹ so với mô hình đầy đủ, đồng thời giúp đơn giản hóa mô hình và tăng tính giải thích.

Các mô hình XGBoost thường, Random Forest và CatBoost có hiệu suất tương đương, với chênh lệch nhỏ về sai số và R^2. LightGBM xếp sau với R^2 = 0.8816. Linear Regression cho kết quả kém nhất với sai số cao (RMSE = 0.7359) và độ phù hợp thấp (R^2 = 0.4560), cho thấy không phù hợp với tính phi tuyến của dữ liệu

- Optimized XGBoost là mô hình phù hợp nhất cho bài toán này.
- Việc giảm số lượng đặc trưng còn 15 không ảnh hưởng nhiều đến hiệu quả dự đoán, góp phần giảm độ phức tạp mô hình.
- Các mô hình ensemble khác (Random Forest, CatBoost) cũng là lựa chọn tốt nếu cần đa dạng hóa mô hình.
- Linear Regression không đáp ứng yêu cầu dự đoán do không xử lý tốt quan hệ phi tuyến.

PHẦN KẾT LUẬN

1. Kết luận tổng quát.

Trong dự án này, em đã tiến hành phân tích và xây dựng các mô hình học máy nhằm dự đoán chỉ số ADR trong ngành khách sạn. Dữ liệu đã được tiền xử lý cẩn thận, các đặc trưng quan trọng được lựa chọn và chuyển đổi phù hợp để phục vụ cho việc huấn luyện mô hình. Nhiều thuật toán học máy khác nhau như Linear Regression, Random Forest, XGBoost, CatBoost, LightGBM, và phiên bản XGBoost tối ưu đã được triển khai và so sánh dựa trên các chỉ số đánh giá hiệu suất như RMSE, MAE và R^2. Trong số đó, mô hình XGBoost sau khi được tinh chỉnh siêu tham số đã cho kết quả tốt nhất, thể hiện qua sai số thấp và khả năng dự đoán chính xác cao. Kết quả này cho thấy rằng việc kết hợp giữa xử lý dữ liệu chất lượng, lựa chọn mô hình phù hợp và tối ưu hóa tham số có thể giúp giải quyết hiệu quả bài toán dự đoán ADR. Mô hình thu được hoàn toàn có thể được ứng dụng trong thực tiễn để hỗ trợ các quyết định định giá và tối ưu hóa doanh thu cho khách sạn.

2. Hạn chế

Mặc dù dự án đã thực hiện nhiều bước tiền xử lý và xây dựng mô hình dự đoán adr, nhưng bài vẫn có một số hạn chế sau:

Chất lượng và tính đầy đủ của dữ liệu gốc: Một số biến có tỷ lệ giá trị thiếu khá cao (ví dụ như company với hơn 94% giá trị bị thiếu, agent với hơn 13%) buộc phải loại bỏ hoặc xử lý bằng các kỹ thuật thay thế. Điều này có thể dẫn đến mất mát thông tin có giá trị tiềm năng trong việc dự đoán ADR.

Dữ liệu có thể không phản ánh đầy đủ xu hướng thực tế: Dữ liệu thu thập chủ yếu trong giai đoạn từ 2015 đến 2017. Điều này có thể không phản ánh đầy đủ các thay đổi thị trường gần đây, đặc biệt là sau đại dịch COVID-19 – khi hành vi du lịch và giá phòng có nhiều biến động lớn.

Giả định trong xử lý dữ liệu: Một số thao tác xử lý dữ liệu dựa trên giả định có thể không hoàn toàn chính xác. Ví dụ, giả định rằng các phòng có ADR = 0 là do miễn phí hoặc lỗi dữ liệu và cần loại bỏ, hoặc giả định đặt adults = 1 cho các phòng không có người lớn để khắc phục lỗi logic. Những giả định này có thể gây thiên lệch nếu không đúng trong thực tế.

Biến mục tiêu bị giới hạn: Dự án chỉ tập trung vào một chỉ số duy nhất là adr, trong khi hiệu quả kinh doanh khách sạn còn phụ thuộc vào nhiều yếu tố khác như tỷ lệ lấp đầy phòng (occupancy rate), doanh thu trên mỗi phòng trống (RevPAR), hoặc chi phí vận hành – những yếu tố này không được xem xét trong phạm vi bài toán hiện tại.

Chưa khai thác dữ liệu thời gian đầy đủ: Mặc dù đã có bước tạo biến season và phân tích theo arrival_month, nhưng dữ liệu thời gian như xu hướng hàng năm, ảnh hưởng của các sự kiện đặc biệt (lễ hội, ngày lễ, mùa cao điểm) vẫn chưa được khai thác triệt để, có thể làm giảm độ chính xác của mô hình trong các thời điểm đặc thù.

3. Hướng mở rộng

Mặc dù mô hình XGBoost đã cho kết quả dự đoán ADR khả quan, vẫn còn nhiều hướng mở rộng để nâng cao hiệu quả và ứng dụng thực tiễn:

Mở rộng và Cập nhật Dữ liệu: Bổ sung các đặt phòng gần đây, đặc biệt sau đại dịch để phản ánh hành vi khách hàng thay đổi. Tích hợp các yếu tố như sự kiện, thời tiết, tỷ giá, đánh giá khách hàng hay giá đối thủ để tăng độ chính xác.

Khai thác Yếu tố Thời gian: Áp dụng mô hình chuỗi thời gian (ARIMA, Prophet) hoặc học sâu (RNN, LSTM) để nhận diện xu hướng, mùa vụ. Tạo đặc trưng cho ngày lễ/sự kiện có ảnh hưởng đến giá phòng.

Nâng cao Kỹ thuật Mô hình hóa: Thử nghiệm các phương pháp ensemble nâng cao như stacking/blending. Khai thác deep learning, đặc biệt với dữ liệu văn bản hoặc hình ảnh. Mở rộng kỹ thuật XAI ngoài SHAP để hiểu rõ hơn về quyết định mô hình.

Mở Rộng Bài Toán: Dự đoán thêm các chỉ số như tỷ lệ lấp đầy, RevPAR, rủi ro hủy phòng. Phát triển hệ thống định giá động. Phân khúc khách hàng sâu hơn để cá nhân hóa chiến lược.

Ứng Dụng Thực Tế: Xây dựng API tích hợp với PMS khách sạn. Thử nghiệm A/B để đo lường hiệu quả mô hình trong môi trường thực tế.

Những hướng này không chỉ giúp cải thiện dự đoán ADR mà còn hỗ trợ nâng cao hiệu quả kinh doanh toàn diện cho khách sạn.

TÀI LIỆU THAM KHẢO

[1] J. Mostipak, “Hotel booking demand dataset,” *Kaggle*, 2018. [Online]. Available: **https://www.kaggle.com/datasets/jessemostipak/hotel-booking-demand/data**

[2] G. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[3] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, San Francisco, CA, USA, 2016, pp. 785–794.

[4] Microsoft, “LightGBM Documentation,” [Online]. Available: https://lightgbm.readthedocs.io

[5] Yandex, “CatBoost Documentation,” [Online]. Available: https://catboost.ai

[6] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, “Optuna: A Next-generation Hyperparameter Optimization Framework,” in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2019, pp. 2623–2631.